

بجي أبو عبد الواحد الأمين بوكليخة

الخوارزمية

و

لغة باسكال

Algorithmique

et

Langage Pascal

بسم الله الرحمن الرحيم

المقدمة :

الحمد لله الذي جعل العلم نورا و الجهل ظلام و الصلاة و السلام على سيدنا محمد و آله العلماء المصطفين الأخيار و أصحابه السادة الأطهار و من تبعهم من عباد الله الصالحين و بعد :

إلى أبناء الطلبة و المتعلمين أقدم هذا الكتاب حول البرمجة بلغة باسكال . و ما عزمي لنشره إلا لما لاحظت من نقص كبير للكتب في تقنيات البرمجة بالطرق الخوارزمية و البنيوية algorithmique et structurée باللغة العربية. لذلك فإنني أحاول هنا تقديم هذه المادة للطلبة العربيين مع تقديم المصطلح الفرنسي و الإنجليزي . نعرض في هذا الجزء لغة باسكال القياسية Le langage Pascal Standard وإضافات Borland Pascal for Windows. أما في التطبيقات فقد استعملنا عدة ترجمات compilers : الأول كان الـ : Ms Pascal و هو ترجمان جيد يوفر كافة أنواع البيانات و إضافات مهمة مثل قراءة و كتابة الأنواع الاختيارية من طرف المبرمج type déclare ou énumère. الثاني الـ : Turbo Pascal 5 و 7 المسمى : TP7 و كذا النسخة المطورة له BPW7 : Borland Pascal for Windows v.7. ثم الترجمان Free Pascal له خصائص Turbo Pascal و Delphi و ترجمان آخر يسمى IriePascal يشحن مجانا كذلك من موقعه و ترجمان Alice و أدوات أخرى تجد مراجعها في آخر الكتاب. تطرقنا إلى جميع أنواع البيانات الساكنة Type de donnée statique و البيانات القياسية و البنيوية و الجذاذات : type de donnée standard et structurée et les fichiers و بنية البيانات الحركية structure dynamique. فهو كتاب تعليمي خلاصة تجربة أكثر من خمسة عشر سنة تدريس نحاول فيها تقديم المادة بأسلوب تدريجي تربوي , فقد تجد بعض التكرار في مواضع كثيرة و برامج نقصد من ورائها تذليل هذا المدخل إلى علوم البرمجة للطالب و تمكينه من هضم مبادئ البرمجة البنيوية ليتفرغ بعدها لتعلم البرمجة بالأشياء programmation objet التي أصبحت أهم أدوات البرمجة

تحت أنظمة التشغيل : windows و solaris و Unix و هو مستقبل علوم الحاسبات. و الله
نسأل السداد و التوفيق و صلى الله على سيدنا محمد و على آله و صحبه .

الباب الأول : نظرة عامة عن لغة باسكال

Chapitre 1:Vue générale sur le langage Pascal

1.لماذا نريد تعلم هذه اللغة

1.Apprendre Pascal ?

ما هي الحاجة التي دفعت إلى إحداث و كتابة لغة باسكال من طرف الدكتور

NIKLAUS WIRTH من جامعة زوريخ السويسرية للمعلوماتية Institut fur Informatik
ETH Zurich

يقول المؤلف إن الحاجة للغة جديدة نبعت من الحرص على تعليم البرمجة the purpose of
Teaching Programming بطريقة منطقية و عقلانية و هذا ما لا توفره اللغات الحالية حيث
أنك تجد صعوبة و تناقضا مع البناء المنطقي السليم ... كما أنني مقتنع بأن اللغة التي ندرس
للطالب ليبر فيها او بها عن أفكاره تمسك و تأثر على طرق بنائه للحلول و الأفكار والإنتاج
المعلوماتي و أن الفوضى التي بها أنشئت هذه اللغات تنعكس على أساليب و تقنيات برامج
الطلبة . إن بناء لغة باسكال يعتمد على قاعدة أساسية وهي توفير لغة لتدريس البرمجة
كاختصاص معرفي تربوي ينطلق من مبادئ واضحة تترجم بسهولة و ببساطة إلى لغة البرمجة .
فهو إذا لغة تربوية تعتمد لإيصال المعرفة و علم البرمجة للطالب لذلك نقول إنها لغة الخوارزمية
Langage Algorithmique و هذا لا يعني بتاتا أننا لا نستعملها لتصميم و تطوير برامج
متقدمة و تطبيقات فعلية. بالعكس فالذي يتمكن منها سيجدها توفر له طرق معالجة جميع
أنواع البيانات all data structure و خاصة بعدما تطورت إلى التصريح بالبيانات على شكل
أشياء و أقسام Classe and object و أصبحت تعتمد البرمجة بالأشياء programmation
Objet كما تجدها معتمدة في لغة delphi.

خلاصة القول إن الهدف من تدريس لغة باسكال هو تعليم المنهجية البنيوية أي البرمجة البنيوية Structured Programming و التي تعتمد على بناء حل المسألة بتقسيمه إلى لبنات صغيرة تجمع وترتب لتصميم الحل النهائي. فهي إذا تقنية و منهجية تعلمك كيف تواجه أي مسألة لتصبح لديك عادة و بداهة ... فلحل أي إشكال يجب أن تبدأ بتجزئته و تبويبه, لكل جزء تضع حلا صغيرا و في الأخير ما عليك إلا ترتيب اللبنات و تنظيمها لتكون الحل الكلي المتناسك.

Problem solving technique of breaking down a problem into manageable components solving each of these components , thus providing resources and combining these resources to solve the PROBLEM.

و أن أهم شيء يجب التركيز عليه للتمكن من هذه التقنية هو فهمك كيفية بناء الحل على شكل إجراءات و دوال فقط : Procédures et Fonctions
هذا في بداية مشوارك ثم تنتقل إلى الوحدات Units و مكتبات الإجراءات و الوحدات الخارجية. Bibliothèque externe.

لذلك سنتطرق إلى لغة باسكال كلغة برمجة و لغة كتابة الخوارزمية في نفس الوقت , فنحن نفضل أن يتعلم الطالب الخوارزمية باللغة العربية . نشجع الطالب على التعبير في لغته التي يتقنها و يفهم فيها المسألة ليخطط الحل فيها ثم ما عليه إلا ترجمة تلك الخطوات إلى لغة باسكال التي في جملها و تركيبها تقترب من تلك الخطوات فما عليه إلا كتابتها بالإنجليزية و هكذا يتدرج من اللغة العربية إلى الإنجليزية و بالتجربة يصبح قادرا على التعبير عن حله مباشرة في الإنجليزية بدون اللف و الدوران على لغة فرنسا التي أصبحت لا تسمن و لا تغني من علم ... فمن الغريب أن تجد بعض الكتب بالفرنسية تسمى أنفسها: دروس في الخوارزمية ما هي إلا ترجمة حرفية للغة باسكال و للأسف أن بعض إخواننا الأساتذة الغير قادرين على تعلم و التعليم بالإنجليزية يصرون على تلقين هذا العلم بالفرنسية و خاصة الخوارزمية بدون إدخال أي تعليمية من أي لغة و بالفرنسية وهم يحسبون أنهم يحسنون صنعا و لو أنهم اطلعوا

على هذا العلم من عند أهله لوجدوا أنهم يقلدون و لا يعلمون ... فلا الطالب يصل إلى علم مفيد و لا الأستاذ طور معلوماته . و إذا أردت التأكد من ضعف لغة فرنسا في هذا العلم إبحث في الـ Internet عن أي معلومة بالفرنسية و بالإنجليزية لترى العجب و كم هو الفرق في عدد الصفحات : فالأولى بالوحدات و العشرات و الثانية بالآلاف فيا ترى : ألا يحسن بنا أن نجتهد قليلا فنفك أسرنا من لغة فرنسا و لا نخون أمانة أبنائنا فنكون قد ربيناهم لزمانهم لا لزماننا و لا نكون أنانيين .

2. ملخص عن اللغة

2.Summary of the Language

ككل لغة للغة باسكال حروف و كلمات و جمل و قواعد لتكوين الجمل المفيدة و النصوص او ما نسميه في علم البرمجة بالإجراءات و البرامج و التطبيقات , و هي: حروف اللغة , الأرقام , الحركات الخاصة , الكلمات الخاصة و قواعد اللغة النحوية التي سنتطرق إليها بالتدقيق في الأبواب المتتالية .

1. الحروف:

<letter> ::=

A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|

digit ::= 0|1|2|3|4|5|6|7|8|9 الأرقام:

3. الحركات الخاصة : Special symbol > ::= +|-|*|/|=|<|>|

<|>|<=|>|=|(|)| { | }|:|=|.|,|;|'|'^|

الكلمات الخاصة: reserved word:

الثوابت .. constants : false , true , maxint

الأنواع .. types : Integer , Real , Boolean , Char , Text

الدوال .. Functions : Abs , Arctan , Chr , cos , Eof , Eoln , Exp , Ln , Odd ,

Ord, Pred, Round , Sin , Sqr , Sqrt , Succ , trunc ,

الإجراءات .. Procedures : Dispose, Get, New , Pack , Page , Put , Read , Readln

, Reset , Rewrite , Unpack , Write , Writeln
الكلمات الخاصة على المستوى القياسي .

Reserved word at the Standard Level :

And , Array , Begin , Case , Const , Div , Do , downto , else , end , File , For ,
Function , goto , If , In , label , Mod , Nil , Not , Of , Or , Packed , Procedure ,
Program , record , Repeat , Set , Then , to , Type , Until , Var , While , With

ملاحظة مهمة : في البداية نتطرق إلى لغة باسكال القياسية standard المحددة في القياس

ISO STANDARD و عند التطبيقات سنستعمل المترجمان compilateur الموفر من طرف
Borland المسمى BPW فهناك عدة مترجمات COMPILERS للغة باسكال تختلف عن بعضها
بالإضافات التي أدخلتها على باسكال القياسي حسب المقياس ISO او المقياس الآخر
ISO/ANSI/IEEE فسنحاول ذكر هذه الاختلافات كلما دعت الحاجة إلى ذلك مثل : MS-
Pascal و هو من الأوائل و Free-Pascal و Irie Pascal و Alice Pascal و غيرها ... تجد
المراجع و القائمة في آخر الكتاب .

الباب الثاني : قواعد تكوين الكلمات في باسكال.

Chapitre 2: Règle de formation des mots en Pascal.

1. الهدف . 1. But

يتطرق هذا الباب إلى القواعد النحوية syntaxe لتكوين التعاريف

identificateur و الأعداد les nombres و سلاسل الحركات chaîne de caractères .

2. التعاريف .

2. Les identificateurs / Identifiers .

ما معنى التعريف ؟

إن التعريف هو كل اسم يختاره المبرمج لثابت constante أو متغير variable أو نوع بيان
data type أو دالة function أو إجراء procédure أو لبرنامج programme أو وحدة unit

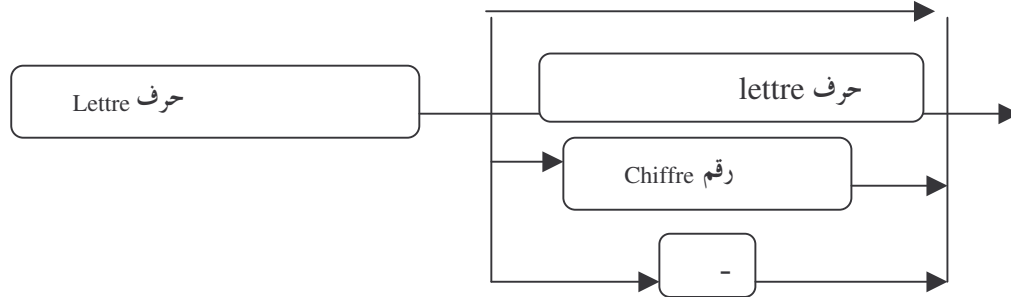
أو أي عنصر آخر من برامج باسكال .

يصاغ التعريف كسلسلة حركات comme chaîne de caractère أولها يجب أن يكون حرفاً أبجدياً يتبع بحرف أو رقم أو كلاهما . يجب أن لا يتعدى طول التعريف 8 حركات caractères في باسكال القياسي standard و لكن في ترجمان BPW7 يسمح بإدخال حركة الوصل : le tiret _ و بطول للتعريف يصل إلى 63 حركة .

مثال: هذه بعض التعاريف الصحيحة : A, Pascal , X1 ,
Mon_Premier_Programme, Nb_Premier, Eq_2_Degre
القاعدة :

باستعمال رسوم Conway نلخص قاعدة كتابة التعاريف :
رسوم Conway طريقة تستعمل في علم الحاسبات computer science لكتابة القواعد النحوية كما تستعمل طريقة أخرى تسمى Backus-Naur form where Syntactic constructs are denoted by English words

enclosed between the angular brackets < and > .



التصريح بالتعريف:

يصرح بالتعريف في قسم التصريحات déclaration أكان علم label أو ثابت constante أو نوع بيان type de donnée أو متغير variable أو اسم دالة nom de fonction أو اسم إجراء nom de procédure و كذلك يكون اسماً للبرنامج أو للوحدة أو للمقياس module أو للمكتبة Library.

أمثلة : هذه مجموعة تعاريف مصرح بها في برنامج ..

التعريف هو : Program Essai

التعريف هو : Label A10

التعريف هو : Const Pi = 3.14 ; Pi

التعريف هو : Type jour = (lundi, mardi, samedi); jour

التعريف هو : Var age : real ; Age

التعريف هو : Function Tang (...) : real ; Tang

التعريف هو : Procedure Swap (var x, y : real) ; Swap

مجال التعريف . Scope of Id ; Portée de l'identificateur

التعريف إسم لا يصبح حقيقة إلا في المجال المصرح به و نعني بمجال التعريف البرنامج الذي عين بداخله أو بالدالة أو بالإجراء الذي تم تعيينه داخله فنقول إنه شامل globale بجميع البرنامج أو محلي locale بالإجراء أو الدالة.

ملاحظة : الكلمات الخاصة باللغة لا تقبل كتعاريف خارج إختصاصها مثل كلمة var أو

do أو begin ...

أمثلة و تمارين :

1. جد التعاريف الصحيحة و الخاطئة منها :

Haut, bas, input, sin, 4ans, and, x, y , x+y, m.pascal

2. جد الخطأ في البرنامج التالي :

program ex;

uses wincrt;

Const Π = 3.14;

Var s, r : real ;

Begin

Read (Π); readln (r); S := $\Pi * r * r / 2$;

Writeln (' la surface est ', s);

end.

3. الأعداد

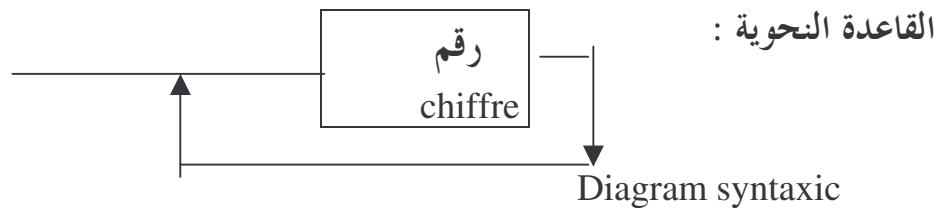
3.les nombres

إن مفهوم الأعداد في باسكال واضح , فالعدد يقصد به بيان data ما و لا يستعمل كاسم لثابت أو متغير إلا في العلامات labels و التي هي نادرة الاستعمال.
الأعداد نوعين : صحيح integer و حقيقي real .

1.3. الأعداد الصحيحة .

3.1 Les nombres entiers/Integer numbers

الأعداد الصحيحة موفرة و يشار إليها بـ: Integer و مجالها
من -32768 إلى +32767 ولها ثابت $\text{maxint} = 32767$.



تعيين قيمة متغير صحيح : يكتب العدد الصحيح من رقم أو أكثر بدون الفصل بينهما
بفراغ أو أي حركة أخرى .

مثال: $x := 100 ; y := -147 ;$

قراءة قيمة المتغير الصحيح:

إجراء read و readln يستعمل بقراءة القيمة من الدخل أو من جذاذة ما file fichier .

مثال : $\text{read} (z); \text{readln} (g);$

عند التنفيذ يضرب المستعمل مثلا : $\leftarrow 123 \rightarrow 147$

عرض أو كتابة العدد الصحيح بتعليمة $\text{write} (...)$.

لكتابه عدد صحيح على الشاشة أو الطابعة نتبع القاعدة التالية :

$\text{Write} (x:m); \text{writeln} (x:m);$

أين x العدد الصحيح و m مجال x format of x أي عدد الأعمدة المطلوب كتابة العدد فيها
 . إن أهمل m يكتب العدد في 14 عمودا ضمنيا . m وسيط من نوع صحيح value
 parameter of type integer.
 مثال :

```
Program ecrit ;
Uses wincrt;
var x : integer ;
Begin
  Write ( ' give an integer number: '); Readln( x );
  Writeln('12345678901234567890'); Writeln( x );
  Writeln ( '12345678901234567890'); Writeln( x:1 );
  writeln('1234567890123456789'); Writeln( x:5 );
End.
```

عندما يترجم البرنامج compiling و ينفذ , مثلا كما يلي :

يدخل المستعمل 45 give an integer number: 45

تكتب 12345678901234567890

تكتب في 14 عمود 45

12345678901234567890

تكتب في مجال من 1 عمود أي من العمود الأول 45

12345678901234567890

تكتب داخل مجال من 5 أعمدة 45

2.3. الأعداد الحقيقية .

3.2 Real numbers/nombres réels

لكتاباة العدد الحقيقي كتابت أو متغير أو لقراءته يجب إتباع قواعد تمثيل الأعداد

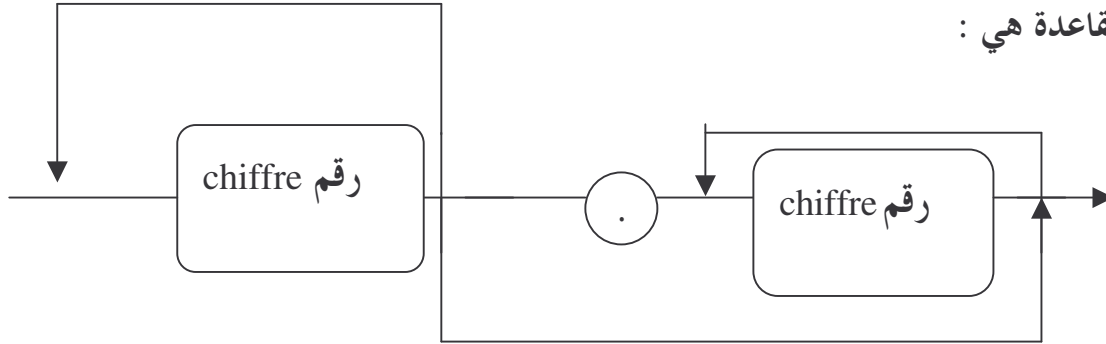
الحقيقية و هي أسلوب العلامة العشرية الثابتة fixed point representation أو العلامة

العشرية العائمة floating point representation

1.2.3 طريقة العلامة العشرية الثابتة .

3.2.1 La représentation en virgule fixe

هي الطريقة التي نستعملها عادة لكتابة العدد الحقيقي حيث نكتب الجزء العشري ثم النقطة العشرية ثم الجزء الصحيح منه .
و القاعدة هي :



أمثلة : 10.2 و 0.012 و 147 و 560.14

```
const pi= 3.14
Var s,x,y : real ;
...
x := 0.25; y := 147;
```

2.2.3 التمثيل بالعلامة العشرية العائمة .

3.2.2 Le format a virgule flottante

هذه الطريقة تكتب كل عدد حقيقي A على شكل صورة لعدد M مضروبا في إحدى قوى

$$10^n * M = A \text{ , حيث } 10$$

M تسمى أساس الصورة .

n عدد صحيح و هو أس العدد A

حسب كتابة الأساس M نجد أسلوبين للتمثيل بالعلامة العشرية العائمة : الصورة الأسية

القياسية للعدد و الصورة العلمية .

أ. الصورة الأسية القياسية. forme exponentielle standard.

في هذه الصورة العدد M يكون أصغر من 1 و أكبر من 0.1 : $0.1 < M < 1$ إن كان A موجبا

و يكون M أكبر من 1- و أصغر من -0.1 : $-1 < M < -0.1$ إن كان A سالبا
و تظهر العلامة العشرية مباشرة أمام أول رقم غير الصفر في M مضروبا في 10^n .
أمثلة : العدد العشري : 22.2 0.55 -0.065

الصورة الأسية القياسية: $(0.222 \cdot 10^2)$ $(0.55 \cdot 10^0)$ $(-0.65 \cdot 10^{-1})$

ب. الصورة العلمية. représentation Scientifique

في هذه الصورة تظهر العلامة العشرية بعد أول رقم غير الصفر في الأساس M .

أمثلة : العدد العشري : 22.33 0.75 0.006

الصورة العلمية : $2.233 \cdot 10^{+1}$ $7.5 \cdot 10^{-1}$ $6.0 \cdot 10^{-3}$

هذه الصورة الأخيرة هي التي اختارها باسكال لتمثيل الأعداد العشرية . و بالتالي فللمبرمج طريقتين لكتابة العدد العشري : بالعلامة الثابتة أو العائمة .

و يقصد بها هنا الصورة العلمية و ذلك عند كتابة قيمة ثابت حقيقي أو تعيين قيمة متغير في البرنامج أو عند قراءتها من جذاذة دخل fichier d'entrée.

ج. القاعدة النحوية .. diagramme syntaxique .

أول رقم يجب أن يكون الصفر ثم نقطة ثم الأعداد الباقية ثم الحرف E ثم الزائد أو الناقص ثم عدد أو أكثر .

أمثلة : 1.23E+5 9.25E+12 0.147E-01 0.314E+01

ك. كتابة الأعداد الحقيقية بتعليمة Write(x) .

كما سبق ذكره فإن باسكال اختار طريقة التمثيل العلمي للأعداد الحقيقية لذلك فإن تعليمة write (x) ستعرض العدد الحقيقي على الشكل العلمي فقط . فإن كنت تريد تمثيل العدد

بطريقة العلامة العشرية الثابتة أو القياسية , فما عليك إلا إتباع القاعدة التالية :

`write (x : m:n)` أين x هو العدد الحقيقي .

M مجال كتابة العدد و هو عدد الأعمدة المطلوب كتابة العدد فيها .

N عدد الأرقام بعد النقطة العشرية .

مثال : إن كانت $x = 123.456$ فـ: `write(x)` تكتبها كما يلي :

في 16 عمود: العلامة + تحمل . `1.234560000E+02`

لو كان العدد سالبا يكتب الناقص لو كتبت `write (x :20)` تنفذ كالتالي :

`1.23456000000000E+02`

أما لو طلبت `write(x:2);` تنفذ كتالي : `1.2E+02` . لاحظ أن العدد تم بتره لأنك طلبت

مجالا من عمودين (2) فقط .

أما لو طلبت `write (x::3);` تنفذ كما يلي: `123.456`

16 colonnes عمود

أما لو طلبنا `write(x:2:2);` تكون النتيجة : `123.45` .

الخلاصة : تكتب الأعداد الحقيقية بالعلامة العشرية الثابتة و العائمة المعروفة بالصورة العلمية

فعلى المبرمج أن يختار كيفية تعيين و عرض القيم الحقيقية .

4. سلاسل الحركات .

4.Les chaînes de caractères .

يقصد بسلسلة الحركات كل مجموعة حركات من حروف و أرقام و حركات خاصة تكتب

بين هلالين في تعليمة `write` مثل :

`Write ('ceci est une chaîne de caractères ');`

`Writeln('هذه سلسلة حركات');`

كل الحركات في جدول ASCII و الحركات العربية إن كانت موفرة في نظام التشغيل .

5. تمارين و أمثلة :

5.Exemple et exercices

1 . في ما يلي مجموعة تعاريف جد الخطأ فيها :

End ; alif.pas ; n°12 ; program ; السلام ; ax + b ; 'bonjour' ; Π ; Σ ; varx ;
typenouveau ; ali_baba ; good ; bad .

2 . أكتب برنامجا يعرض قيمة maxint بالشكل القياسي و في مجال من 20 عمود و قيمة

Π (pi) بأربعة اشكال ..

الباب الثالث : قواعد البرمجة في باسكال.

Chapitre 3: Règles de programmation en

Pascal.

1. الهدف 1.BUT

يعرض هذا الفصل المبادئ الأساسية لكتابة برامج باسكال و هي القواعد النحوية
للتصريح بالبيانات و إعطاء التعليمات تبني بها خوارزمية حل المسألة التي طرحت عليك و التي
تصبح في الأخير البرنامج .

2. رأس البرنامج و بنيته العامة .

2.En tête du programme et structure générale.

كل برنامج في باسكال يبدأ بالكلمة الخاصة Program تتبع بتعريفه أي اسم البرنامج و هو

رأس البرنامج

مثال : (* رأس البرنامج en tête du programme *)

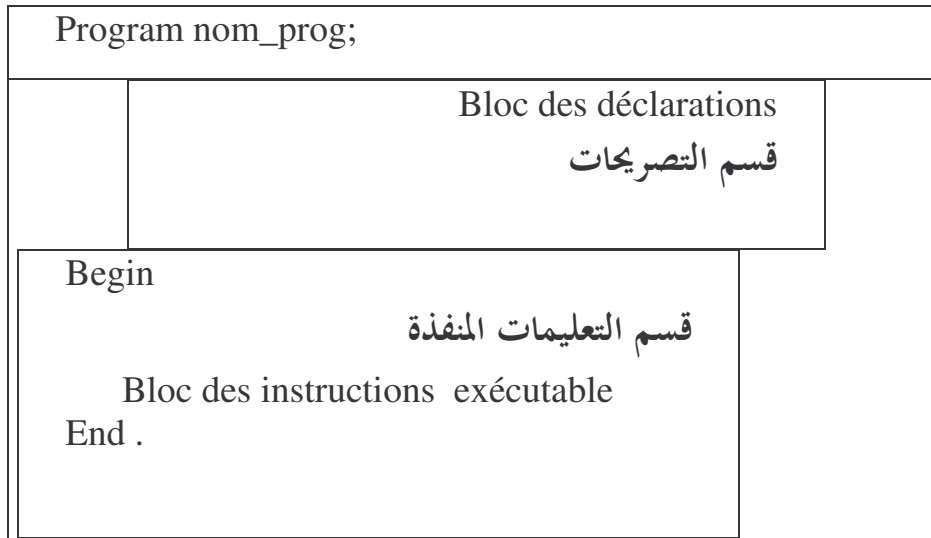
Program Exercice ;

بنية البرنامج:

كل برنامج ينقسم إلى رأس و جسم البرنامج :

فالرأس هو الذي عرفناه يتبع بالجسم : ينقسم إلى قسمين : قسم التصريحات bloc de déclaration و قسم التعليمات bloc des instructions .

و القاعد العامة لبنية برامج باسكال هي :



3.التصريحات .

3.Declarations

في هذا القسم سنقدم للترجمان compiler التعاريف التي سنستخدمها و البيانات و الأدوات التي سنستخدمها في البرنامج . فالترجمان لا يعرف إلا الكلمات الخاصة و التعاريف الخاصة باللغة فكل كلمة جديدة أو نوع بيان جديد أو إجراء أو دالة جديدة عليه يجب أن نصرح بها قبل أن يسمح لنا باستعمالها في قالب التعليمات المنفذة .

تقسم التصريحات إلى ستة أقسام :

قسم التصريحات بالعلامة déclaration des étiquettes

و ذلك باستعمال الكلمة الخاصة ; label

قسم التصريح بالثابت déclaration de constantes

const..... بالكلمة الخاصة

قسم التصريح بأنواع البيانات الجديدة `déclaration de Type`

بالكلمة الخاصة `Type ...`

قسم التصريح بالمتغير `déclaration des variables`

بالكلمة الخاصة `var ...`

قسم التصريح بالدالة أو الإجراء

`déclaration de fonction ou procédure`

`Function end;`

`Procedure end;`

1.3 التصريح بالعلامة .

3.1 Déclaration d'étiquette

العلامة طريقة كانت تستعمل كثيرا في البرمجة باللغات الغير بنيوية كلغة `basic` و `fortran` وهي عنوان يوضع في سطر ما يمكن القفز إليه أماما أو خلفا لتكرار أو مواصلة تنفيذ جزء من البرنامج بتعليمية `goto` و لكن حاليا تحلت تقنيات البرمجة عن هذا الأسلوب الفوضوي في كتابة البرامج . لذلك فإننا نمنع منعاً باتاً المتعلم من استعماله إلا للضرورة القصوى مثل البحث عن الأخطاء في برنامج طويل تريد تجربة أجزاء متفرقة فيه ..

العلامة يشار إليها بالكلمة الخاصة `label` ثم تتبع بالعلامات يفصل بينها بالفاصلة وهي في باسكال القياسي عدد صحيح لا يتعدى 4 أرقام `digits` و يسمح على المستوى الإضافي باستخدام تعاريف العلامات من حروف أبجدية .

مثال :

لاحظ أننا نعلن عن استعمال `wincrt:` وهي واجبة إذا أردنا أن نعمل تحت نظام

التشغيل `Windows` .

`Program loop;`

`Uses wincrt;`

`Label 1 , algers oran ;`


```

Begin
  Algers : goto 1
  Oran : writeln (' وهران هنا ');
          1: goto oran
end.

```

2.3. التصريح بالثابت

3.2 declaration de constante

تعريف الثابت définition de la constante

هو كل قيمة معروفة مسبقا قبل تنفيذ البرنامج و لا تتغير خلال التنفيذ , فهو إما عددا صحيحا أو حقيقيا أو سلسلة حركات .

أمثلة :

```

const g = 9.8 ;
      nom = 'pascal';
      max_entier = maxint ;
      min_entier = - maxint ;

```

و هذا يعني أننا عينا قيما لكل من g و nom و max_entier و min_entier و هي ذاكرات سوف لا يسمح المترجمان للمبرمج أن يقوم بتغيير قيمها ما دامت مصرحة كثوابت . و هي أول خطوة تقوم بها عند كتابة الخوارزمية : تحديد البيانات الثابتة و المتغيرة .

الثابت العددي les constantes numériques..

يمكن تعيين أي عدد صحيح أو حقيقي لثابت مع احترام قواعد كتابة الأعداد و مجالاتها .

أمثلة لأعداد ثابتة : 12.3 0.25 +14.36 00147 -147.25E-6 25.0E21 -15E8

الثابت كسلسلة حركات constantes chaînes de caractères تعتبر ثابت

كل حركة أو أكثر تكتب بين مزدوجين { '...' }

أمثلة : const reponse = '← Taper sur

; 'السلام عليكم ورحمة الله' = salut

كل سلسلة حركات تكون من 1 إلى 255 حركة . caractères

constantes prédeclarées. الثوابت المصرح بها

الثوابت التي سبق التصريح بها للترجمان معرفة له تستعمل مباشرة و هي : maxint

ثابت صحيح قيمته 32767 , أكبر قيمة في النوع الصحيح integer

و true و false ثابتين من نوع منطقي boolean .

مثال :

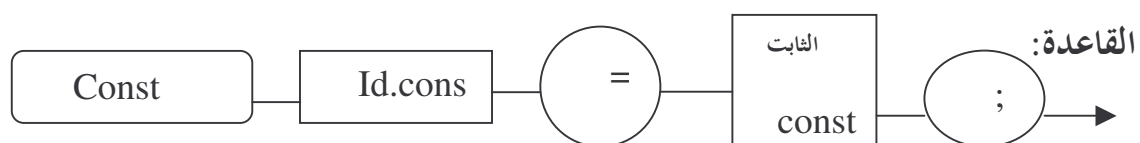
```
program constante;
  uses wincrt ;
  const max_entier = Maxint ;
  vraie = true;
  faux = false ;
begin
  writeln('max des entiers =', maxint);
  writeln('vraie=', vraie);
  writeln('faux=', faux);
end.
```

Max des entiers = 32767

التنفيذ يكون كالتالي:

Vraie=true

Faux=false



* تمرين : إليك بعض الثوابت الفيزيائية أكتب التصريحات اللازمة لاستعمالها في برنامج :

$g = 9.8$; $1\text{km} = 10^3$; $1\text{kg} = 9.8\text{j}$

مثال : Const rayon_terre = 1E7 ;

الخلاصة : الثابت قيمة يصرح بها في قسم خاص يبدأ بالكلمة const لا نستطيع التعيين إليه

في البرنامج.

3.3. التصريح بالنوع .

3.3 Declaring data type/ declaration de type

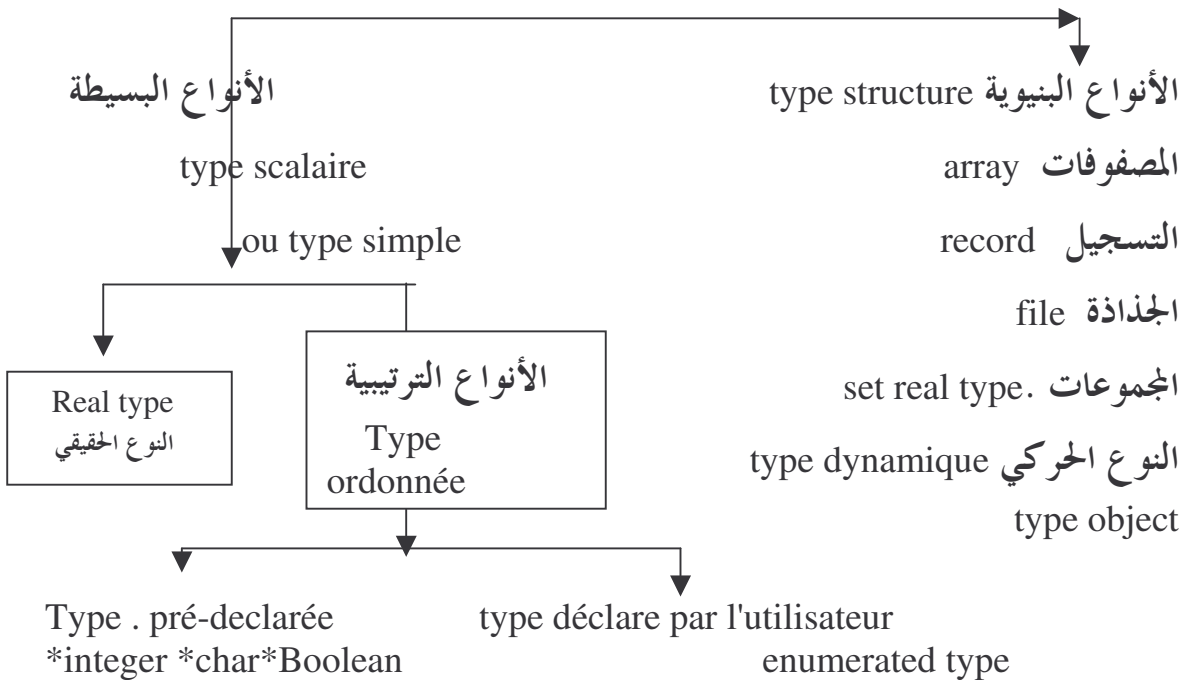
أنواع البيانات في باسكال تقسم إلى مجموعتين : الأنواع العددية أو البسيطة type scalaire simple type ; و الأنواع البنيوية type structure ; structured type .

سميت الأولى بالأنواع البسيطة لأنها لا تقبل القسمة إلى أنواع أخرى فهي أحادية النوع أما الأنواع البنيوية فهي متكونة من أنواع أخرى .

الأنواع القياسية standard المصرح بها في باسكال هي : real; integer char ; boolean ; الأنواع التعدادية أو الترتيبية énumère ou ordonne هي القياسية و التي يسمح للمبرمج بالتصريح بها .

بالإضافة إلى النوع الحقيقي نسمي هذه الأنواع بالبسيطة لأنها مجموعات من صنف واحد فقط .

أنواع البيانات في باسكال. type de données.



1.3.3. النوع الصحيح .

3.3.1 Integer type/ Type entier

مجموعة الأعداد الصحيحة في باسكال محدودة المجال مابين : -32768 إلى +32767 في BPW وله ثابت Maxint .

يصرح بها بالكلمة الخاصة integer .

مثال : var i , j : integer ;

2.3.3. النوع الحقيقي

3.3.2 Type Reel /Real Type

هي مجموعة الأعداد الحقيقية الموفرة من مجال $1.9E-39$ إلى $1.9E+38$, و تجدها على المستوى الإضافي من مجالات أكبر مثل $E-4932$ إلى $E+4932$ أو أكبر حسب قوة الترجمان و المشغل المرافق co-processor الخاص بالحساب .

مثال : var x , y : real ;

z, w : double ;

3.3.3. النوع حركات .

3.3.3.Char type/character

في باسكال char هي حركات من 8 bit في جدول ascii الموفر في كل الحاسبات وهي 256 حركة caractères .

مثال : var reponse : char ; و هي قيمة متغير من حركة واحدة فقط

4.3.3. النوع المنطقي .

3.3.4 Type Boolean /Boolean Type

و هو أصغر نوع يتكون من ثابتين فقط true و false .

رتبة false صفر (0) و رتبة true 1

مثال : var a, f : boolean ;

5.3.3. النوع المصرح به من طرف المبرمج .

3.3.5 Enumerated type/type declare

هذا النوع يعرف من طرف المبرمج فهو نوع محدث يصرح به و يحدد عناصره , كل عنصر يعد ثابت ضمن المجموعة , فهي مرتبة كذلك , الثابت الأول له الرتبة الصفر (0) ثم 1 , 2 ... مثال :

```
Type couleur = ( rouge , blanc, vert , bleu );
mois = ( janvier , fevrier, mars ,avril, mai , juin);
jours = ( samedi, dimanche, lundi,mardi,mercredi);
Var photo : couleur;
vacance : mois ;
travail : jours ;
```

في هذا المثال حددنا ثلاثة مجموعات جديدة و صرحنا بها في قسم التصريح بالنوع. ثم اخترنا لكل نوع متغير نود أن يأخذ إحدى القيم من المجموعة التي ينتمي إليها. فمثلا المتغير vacance سيأخذ قيمة واحدة من النوع mois و هي janvier أو fevrier أو ... إلى آخره . كما تلاحظ بهذه الطريقة يستطيع المبرمج إحداث أي نوع جديد من البيانات شريطة ذكر عناصره بين قوسين مرتبة كثواب .

يحدد باسكال استعمال هذا النوع داخل البرنامج فقط أي لا نستطيع قراءة قيمة المتغير من جذاذة دخل بتعليمة read مباشرة أو كتابته مباشرة بتعليمة write على جذاذة خرج .

6.3.3. النوع المجال .

3.3.6 Subrange type/type intervalle

هذا النوع يأخذ قيمه من الأنواع التعدادية : integer, char , boolean أو النوع الاختياري الذي يصرح به المستعمل . يصرح بنوع المجال في قسم التصريح بالنوع أو عند

التصريح بالمتغير .

أمثلة : Type long_mois = 1..31 ;

var bit : 0..1;

notes : 'A'..'F' ;

فائدة استعمال المجال هي إمكانية مراقبة تجاوزات استعمال المتغير في غير المجال المحدد له

الخلاصة : التصريح بالنوع ضروري و ذو فائدة كبيرة لبناء برامج كبيرة و جيدة و هي

الخطوة الأولى في تعلمك التخطيط لحل مسائل كبيرة و معقدة و المدخل إلى لغة الأشياء

. langage orienté objet

**** في الباب السادس تجد معلومات أكثر حول أنواع البيانات .

4.3. التصريح بالمتغير

3.4 Déclaration de variable

ما معنى المتغير ؟

المتغير هو قيمة ينتظر أن تتغير خلال تنفيذ البرنامج .

و في الحقيقة فهي ذاكرة يحضرها المترجم compiler معرفة باسم كعنوان لها و بقيمة غير

محددة و من النوع الذي طلبه المبرمج .

فهي ذاكرة نطلب تحضيرها لنعين إليها قيمة أو نخزن فيها ما نقرأه من الدخل أو من جذاذة

ما أو لنحجز فيها مؤقتا نتيجة قبل كتابتها على خرج ما : شاشة أو طابعة أو قرص . فهي

المكان الذي نحتاجه كثيرا للقيام بالعمليات الحسابية و تخزين جميع بياناتنا data المطلوب

معالجتها في الذاكرة المركزية .

أمثلة : Var a : integer ;

b : real ;

c : char ;

d : boolean ;

begin

a := 10 ;

```
readln ( b ) ;    c := 'z' ; d := true ;
end.
```

A	? 10
B	?0.5
C	?z
D	?true

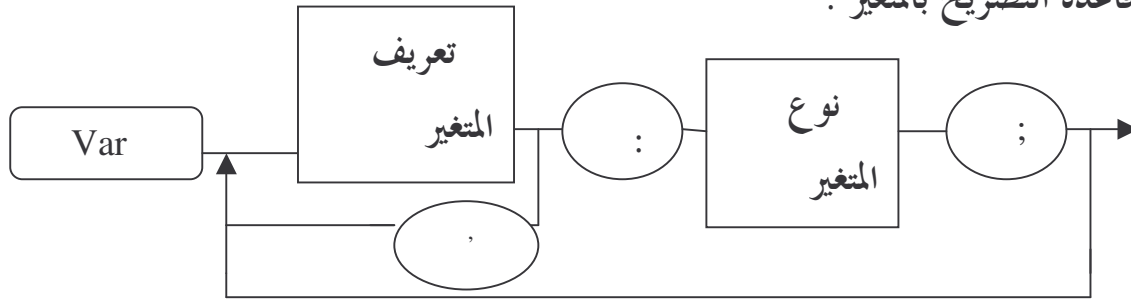
تنفيذ هذا الجزء من البرنامج يكون كالتالي :

في البداية تكون قيم المتغيرات مجهولة :

ثم عندما عينا لها قيما تخزن فيها ,

أو تقرأ من الدخل مثل (b) readln ضرب المستعمل قيمة 0.5 التي تم وضعها بالذاكرة b

قاعدة التصريح بالمتغير .



تبدأ بالكلمة الخاصة var ثم التعريف ثم نقطتين ثم نوع المتغير .

إسم المتغير و نوعه لا يتغيران و لا يمكن تعيين قيمة إليه من غير نوعه .

يجوز التصريح بنفس النوع أكثر من مرة و في أكثر من سطر .

مفهوم مجال المتغير . notion de portée de variable .

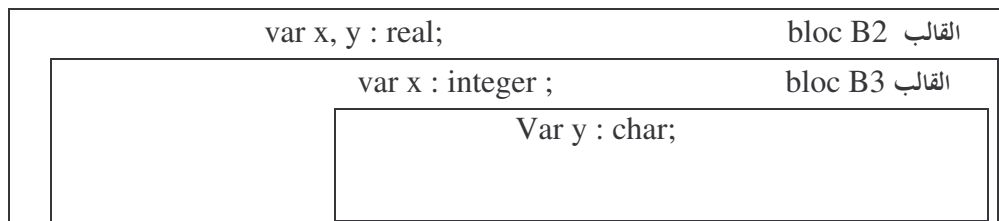
التصريح بالمتغير يعني تحضير ذاكرة في البرنامج إلا أن البرنامج في باسكال يبني على شكل

قوالب bloc من إجراءات procédures أو دوال functions أو وحدات units و بالتالي فالمتغير

يأخذ معناه في المحيط المخصص له أي داخل المجال الذي صرح فيه . فقد يكون مجاله شاملا

بكل البرنامج global أو محلي بجزء منه local .

القالب B1 bloc



من هذا المثال يتضح لنا مجال تصريح المتغير , في القالب B1 صرح بـ: x,y من نوع حقيقي
 real فهو تصريح يشمل القالب B2 و B3 لذلك نقول أن مجال المتغيرين x,y شامل global
 بـ: B2 و B3 و محلي بـ: B1. أما في القالب B2 فقد صرح بـ: x من نوع صحيح integer
 فهو شامل بـ: B2 و B3 أو محلي بـ: B2 و لكن يستثنى من مجاله B1. أما في القالب
 B3 فقد صرح بـ: Y من نوع حركات char فهو محلي فقط local بـ: B3 لا يتعدى B3.

5.3 تمارين :

3.5 Exercices

1. إليك التصريحات التالية : صحح الخطأ فيها .

```
Const  $\Pi$  = 3.14 ;
Var begin : char ;
    'a' : char ;
    bonjour : 'السلام عليكم' ;
    reel : real ;
```

2. إليك جزءاً من برنامج ما و القيم أدناه , إلى أي متغير تعين كل قيمة:

```
Var entier : integer ;
    Reel : real ;
    Haraka : char ;
    .....
```

القيم هي : Y, 32767, 0.12, 0.0, -45, 2.5, -4E-15, maxint

3. إليك التصريحات التالية أتم كتابتها بناء على محتوى البرنامج :

```
Type jour = ( mardi , mercredi , jeudi , vendredi ) ;
    Indice = -15 .. 45 ;
    Var I : integer ;
    .....
begin
    J := mardi ;
    I := 0 ;
    K := I + 10 ;
```


04 التعليمات المنفذة .

04: Statements / Les instructions exécutable .

1.4 هدف الفقرة :

4.1 BUT

التعليمات هي الأوامر التي تصدرها في جسم البرنامج و الدالة و الإجراء للقيام بالخدمات المرتقبة حل مسألة ما . فهي التنفيذ الفعلي للبرنامج. statements denotes actions that the program can execute .

التعليمات المنفذة قسمين : البسيطة simple و البنيوية structured .
التعليمات البسيطة هي التي لا تتكون من تعليمات أخرى كجزء منها .
و التعليمات البنيوية تتكون من تعليمتين أو أكثر .

البسيطة simple	البنيوية structured
التعيين assignment الإجراء procedure goto التعليمة الفارغة	Compound : begin . end If then else / Case of ... For/ While /Repeat / With

2.4 القاعدة النحوية

4.2 Règle syntaxique

يجب التفريق بين التعليمات بالنقطة فاصلة (;) و أن تكون داخل قالب begin و end .

3.4 التعيين .

4.3 Assignment statement .l'affectation

إذا أردنا تعريف التعيين بالطريقة الخوارزمية algorithmique نقول : ضع قيمة العبارة التي

على يمين الرمز =: في الذاكرة المعرفة بالاسم الذي على اليسار . و نعبر عنها في الخوارزمية بالسهم مثلا: 10 في الذاكرة x

و القاعدة : عبارة = expression : تعريف المتغير identificateur

بلغة باسكال

أمثلة : بالطريقة الخوارزمية

x := 100

x ← 100

y := x+1

y ← X+1

يجب أن تكون نتيجة تقييم العبارة على يمين التعيين من نفس نوع المتغير الذي تعين له .

4.4 التعليمات الحسابية .

4.4 Les instructions arithmétiques

التعليمة الحسابية هي كل تعليمة تعين عبارة حسابية expression arithmétique إلى

متغير من نوع عددي numérique .

1.4.4 العبارات الحسابية البسيطة .

4.4.1. Les expressions arithmétique simple

العبارة الحسابية البسيطة هي كل سلسلة من المتغيرات و الثوابت يفصل بينها بعامل

حسابي opérateur arithmétique .

العاملون للحساب ستة وهي les opérateurs arithmétique sont :

للجمع : + مثل x + y

للطرح : - مثل x - y

للضرب : * مثل x * y

للقسمة : / مثل x / y

للقسمة الصحيحة : div مثل x div y

متبقى القسمة الصحيحة : mod مثل x mod y

العاملون les opérateurs : + و - و * : يعملون على أنواع البيانات من نوع صحيح integer و حقيقي real و نتيجة التقييم تكون من نفس نوع المعاملون Opérandes : من نوع صحيح إن كان كلهم صحيح و حقيقي لو كان أحدهم حقيقي .
العامل / القسمة العامة للقيام بقسمة بين النوع الصحيح أو الحقيقي و النتيجة تكون دائما من نوع حقيقي .

العامل div : لقسمة عددين صحيحين و نتيجتها صحيحة .

العامل mod : متبقى القسمة الصحيحة .

أمثلة : $123 \bmod 5 \rightarrow 3$

$123 \text{ div } 5 \rightarrow 24$

2.4.4 تقييم العبارات الحسابية .

4.4.2 Évaluation des expressions arithmétiques

مثال : لو كانت العبارة التالية : $A * B + C * D$

كيف سيقومها ترجمان باسكال للتعين في متغير ما ؟

لحساب هذه العبارة يستعمل قواعد الأسبقية و ترتيبها règle de priorités ou de

précédence des opérateurs وهي قواعد تحدد من العامل الأول و الثاني و الذي ينفذ قبل الآخر .

القاعدة الأولى : كل عبارة حسابية تقييم من اليسار إلى اليمين بداية من عملية الضرب (*)

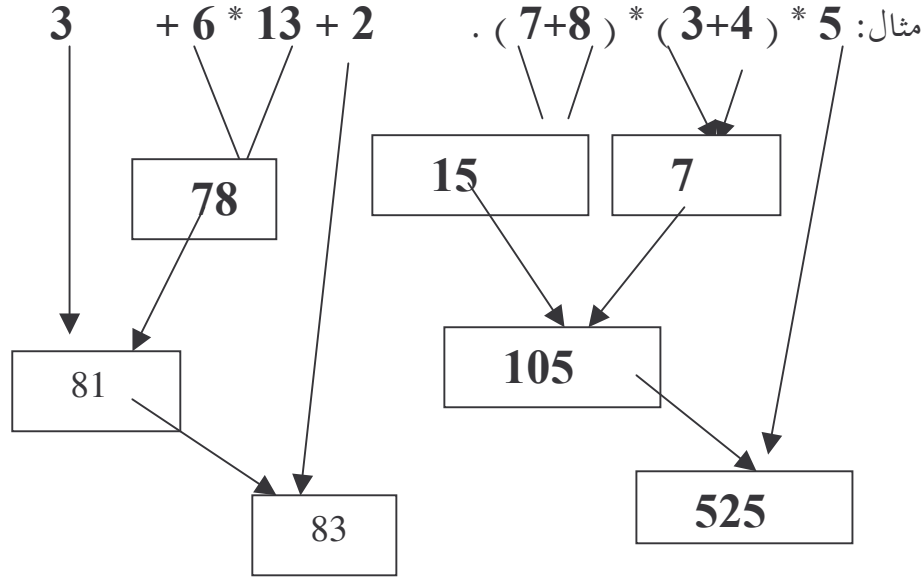
(/) , القسمة الصحيحة (div) أو المتبقى من القسمة الصحيحة (mod) . ثم

دائما من اليسار إلى اليمين عمليات الجمع

(+) و الطرح (-) .

القاعدة الثانية : كل ما كان بين قوسين له الأسبقية الأعلى .

من هذه المثال البسيط التالي يتضح كيف تعمل الأسبقية الأعلى إن وجدت أقواس و
الأسبقية بدون أقواس و هي دائما إن وجدت أقواس يبدأ بتنفيذ ما بداخلها بإتباع
القاعدة الثانية و الأولى دائما .



ملخص حول عاملون العبارات الحسابية
règles de priorités des opérateurs arithmétiques

الأسبقية Priorite	نوع النتيجة Type de resultat	نوع المعامل Type d'operande	العامل operateur	وصف العامل Description de l'operateur
2	Integer Real	Integer Real	+ و -	الجمع / الطرح
1	Integer real	Integer real	*	الضرب
1	Real	Integer real	/	القسمة
1	Integer	Integer	Div	القسمة الصحيحة
1	Integer	Integer	Mod	متبقى القسمة الصحيحة

5.4 التعليمات المنطقية .

4.5 Les instructions logiques

التعليمة المنطقية هي كل تعيين لعبارة منطقية إلى متغير منطقي .

ماهي العبارة المنطقية ؟ expression logique

كل عبارة نتيجة تقييمها منطقي تعد عبارة منطقية . فهي عبارة مكونة من ثوابت و

متغيرات من أي نوع فصل بينها بعامل المقارنة : (< > <= >= <> =) أو عبارة

مكونة من حدود facteur من نوع منطقي يفصل بينها بعامل منطقي { or and not }

أمثلة :

Not (juste) ; (grand > petit) or True

Z := (A = B) and (a = c) ;

Bas := (x + y >= Z) or (y-z < x) ;

Haut := (i <= max) and (j = min) ;

Bit := True and not Trouve ;

Jeu := (false and true) or not (false) ;

على سبيل المثال جد التصريحات اللازمة لكل عبارة ؟

كما تلاحظ عموما العبارة المنطقية تتكون من عبارات علاقية expression relationnelle

يربط بينها بعامل منطقي .

ما هي العبارة العلاقية ؟

كل عبارتين بسيطتين من أي نوع كان رُبطتا بعامل علاقة تسمى عبارة علاقية . عاملوا

العلاقة les opérateurs relationnels هم :

= المسوات بين الحدين facteurs أو العبارتين expressions .

< > الاختلاف différence

> الأكبر plus grand

< الأصغر plus petit

=> أكبر أو يساوي plus grand ou égal
 <= أصغر أو يساوي inférieur ou égal
 عاملوا المنطق . les opérateurs logiques هم :
 Not : النفي négation
 And : و المنطقي Et logique
 Or : أو المنطقي Ou logique
 مثال :

```

Program mantik;
Uses Wincrt;
Var x , y , z : real ;
    Egal , sup , inf : Boolean ;
Begin
    Write( ' أدخل ثلاث قيم '); Readln ( x , y , z ) ;
    Egal := x = y ;
    Sup := ( x > z ) and ( x > y ) ;
    Inf := ( y < z ) or ( x + y < z ) ;
    Writeln ( ' يساوي ' , egal , ' أكبر ' , sup ) ;
    Writeln ( ' أصغر ' , inf ) ;
End.

```

تقييم العبارات المنطقية :

تقيم العبارة المنطقية حسب قواعد الأسبقية التالية :

Not له الأسبقية الأعلى : الأولى .

* و / و div و mod و and : لهم الرتبة الثانية.

+ و - و **or** : لهم الأسبقية الثالثة .

= و < و > و <= و >= : لهم الرتبة الرابعة .

ملخص عاملوا باسكال

Résume des opérateurs

وصف العامل	العامل Operateur	العملية opération	نوع المعامل opérande	نوع النتيجة Résultat	رتبة الأسبقية priorité
عامل منطقي Opérateur booléen	Not	النفي المنطقي	منطقي	منطقي	1
عاملوا الضرب Opérateur Multiplic atif	*	الضرب	صحيح حققي	صحيح حققي	2
	/	قسمة حقيقية	صحيح حققي	حققي	
	Div	قسمة صحيحة	صحيح	صحيح	
	mod	متبقى القسمة الصحيحة	صحيح	صحيح	
	and	و المنطقي	منطقي	منطقي	
عاملوا الجمع Opérateurs additifs	+	الجمع	صحيح حققي	صحيح حققي	3
	—	الطرح	صحيح حققي	صحيح حققي	
	or	أو المنطقي	منطقي	منطقي	

6.4 إجراءات القراءة و الكتابة .

4.6 Procedure read , readln , write , writeln

1.6.4 تعليمة القراءة .

4.6.1 Read , readln

تعليمة القراءة تسمح للمبرمج إدخال البيانات (المعطيات) إلى البرنامج خلال التنفيذ . فهي إجراء مصرح به مسبقا في الترجمان .

C'est une procédure déclarée ..available procedure

و القاعدة لاستعمالها هي :

Read (nom de fichier , liste des variables) ;

قائمة المتغيرات اسم الجذاذة

Read (f , P1,P2, ..., P_n) ;

أين f اسم جذاذة الدخل fichier d'entrée المراد قراءة البيانات منها .

و P₁,P₂, ..P_n وسطاء الدخل parameters و هي أسماء المتغيرات التي نعين إليها القيم .و

تعني اقرأ من الجذاذة المعينة بالاسم القيمة أو القيم و وضعها في المتغير أو المتغيرات P₁ .. P_n .

إذا أهمل اسم جذاذة الدخل فهذا يعني بالنسبة للترجمان compiler أنك تقصد جذاذة الدخل

القياسية fichier input standard الذي هو لوحة المفاتيح clavier ; keyboard . أما إذا

عينت اسم جذاذة الدخل فهي غالبا جذاذة مخزنة في القرص fichier sur disque .

مثال :

Program lit ;

Uses winCrt ;

var a , b : Real ; c : char ;

begin

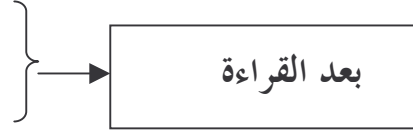
read (a , b) ; read (c) ;.....;

end.

عند التنفيذ في الذاكرة المركزية en mémoire centrale نجد أنه حضر لنا ذاكرات a,b,c

بدون قيم فيها ثم عندما نضرب القيم باللوحة و ندخلها بالضغط على ملمس الدخول ,
تقوم تعليمة القراءة بنقل القيم من اللوحة إلى الذاكرة و تضعها في الذاكرة المعنية لها .

A	?	10.5
B	?	1E+12
C	?	T



محتوى الذاكرة قبل القراءة مجهول

تعليمة اقرأ ثم ارجع إلى السطر. Readln.

قاعدة كتابة readln هي :

$\text{readln} (F , P_1, P_2, .. P_n); \longleftrightarrow \begin{cases} \text{Read}(F,P_1,P_2,..P_n); \\ \text{Readln} (F) ; \end{cases}$

فهي نفس تعليمة read يضاف إليها طلب الرجوع إلى السطر في الجذاذة التي تقرأ منها
. و دائما إن أهمل اسم جذاذة الدخول تكون جذاذة الدخول المفترضة par défaut هي لوحة
المفاتيح keyboard أي اقرأ ما يدخل من لوحة المفاتيح .

بعض خصائص تعليمة read , readln :

$\text{Read} (a) ; \text{Read} (b) ; \longleftrightarrow \boxed{\text{Read} (a,b);}$

$\begin{cases} \text{Read} (x) ; \\ \text{Read} (y) ; \\ \text{Readln} ; \end{cases} \longleftrightarrow \boxed{\text{Readln} (x,y);}$

2.4.6 تعليمات الكتابة .

4.6.2 Write ,writeln

لمعرفة نتيجة معالجة البيانات لا بد لنا من تعليمة تكتب لنا على الشاشة أو الطابعة أو على
وسيلة تخزين كالقرص أو الشرط فهي إجراء الكتابة write.

و هي إجراء مصرح به مسبقا في اللغة available procedure

القاعدة لاستعمالها هي : $\text{write} (\text{filename}, V_1,V_2, .. V_n);$

Filename تعني اسم الجذاذة التي يطلب الكتابة عليها . إن أهمل و لم يعين فهي الشاشة.

V_1, V_2, \dots, V_n الوسطاء parametres و هم : ثابت , متغير, سلسلة حركات , حقل تسجيل field of record .. أو تسجيل .

WriteLn تشبه write و لكنها تكتب إشارة نهاية السطر line marker في آخر السطر في الجذاذة و بالتالي ترجع إلى السطر الموالي .

Write (f , V_1, \dots, V_n) ;
WriteLn (f) ;

} WriteLn (f, V_1, \dots, V_n);

تشكيل الكتابة — write(p:m) .

وسيط P : parameter

m : عدد الأعمدة التي نود كتابة الوسيط فيها

و تعني أكتب p في على الأقل p عمود colonne .

5. الدوال القياسية.

5. Fonctions Standards

يوفر باسكال جملة برامج فرعية تسمى دوال functions , حيث يستطيع المبرمج إدخالها في تعليماته لربح الوقت مثل حساب جذر تربيع عدد ما ننادي الدالة sqrt (..) بدلا من كتابة برنامج فرعي داخل برنامجنا.

هذه الدوال القياسية وهي:

الدوال الموفرة في مكتبة باسكال القياسية *fonctions disponibles*

وصف الدالة Description de la fonction	نوع المتغير Type de variable	نوع النتيجة Type de résultat	مثال : Exemple
القيمة المطلقة $\text{abs}(x)$ valeur absolue de x	Integer Real	Integer Real	إن كانت $x = -15$ $z := \text{abs}(x)$ ترجع 15 في z

Odd(x); Impaire لاختيار هل x فردي	Integer	Boolean	X:=11; Z:= odd(x); قيمة z true
Trunc(x) تتر x إلى عدد صحيح	real	Integer	x:=10.5 إذا z:=trunc(x); قيمة z 10
Round(x) تقرب العدد الحقيقي إلى صحيح	real	Integer	X:=10.5 Z:= round(x) نتيجة z 11
Ord(x) ترجع رتبة x	Type énumère ou ordonne	Integer	X:='W'; Z:=ord(x); قيمة z 87
Chr(x) ترجع الحركة برتبة x	Ordonne	char	X:= 87 Z:=chr(x) ترجع الحرف w
Eoln(f) تعلن نهاية السطر في الجذاذة	fichier	Boolean	While not eoln(f) do read(f,v) ...
Eof(f) تعلن نهاية الجذاذة أم لا	fichier	Boolean	While not eof(f) do Read(f,E)...
Sin (x) جيب الزاوية x	صحيح حقيقي	real	Z:= sin(x) radian ب X
Cos (x) تمام جيب x	عدد	real	Z:= cos (x) بالراديان X
Sqr(x) تربيع x	عدد	عدد	X:=10 Z:=sqr(x)
Sqrt(x)	real	real	X:=9; Z:=sqrt(x)

جذر تربيع x			
Succ(x) التابع لـ: x	ترتيبي	ترتيبي	$X:=9$ $Z:=succ(x)$ قيمة Z 10
Pred(x) السابق لـ: x	ترتيبي	ترتيبي	$X:=9$ $Z:=pred(x)$ قيمة Z 8
Exp(x) الدالة الأسية لـ x	عدد	real	$Z:=exp(x)$
Ln(x) لوغاريتم x	real	real	$Z:=ln(x)$
arctan (x)	real	real	$arcsin(x)=arctan(x/sqrt(1-sqr(x)))$ $arccos(x)=arctan(sqrt(1-sqr(x))/x)$

6.تمارين :

6.Exercices

الهدف من هذه التمارين تدريبك على قواعد اللغة و تمكنك من كتابة التعليمات بدون
أخطاء نحوية ثم في الباب الموالي سنتطرق إلى الطرق الخوارزمية *méthodes algorithmique*
لحل أي مسألة .

1.جد التصريحات الضرورية لإتمام البرنامج التالي :

Program ex ;

.....

Begin

mois := Janvier ;

vacance := juin ;

travail := mois < vacance ;

writeln (' هل هو مريض ؟ ' ,not (travail));

end.

2.أكتب البرنامج الذي كان تنفيذه كالتالي :

إضرب عددين صحيحين : entrer 2 nb.entiers

10 145

ضرب المستعمل

تم التحويل x=145,y=10 échange effectué

3. أكتب برنامجا يجد حاصل القسمة لعددين صحيحين و متبقى القسمة منفردا بدون

استعمال div و mod ثم أكتب نفس البرنامج بهما .

ما الفرق بين الطرقتين .

4. أكتب البرنامج الذي يحول بعض القياسات من النظام الإنجليزي إلى النظام العالمي mksa

1 inch = 2.54 cm ; 1 foot = 30.480 cm ; 1 yard = 0.9144 m

1 mile = 1.609 km ; 1 gallon = 3.785 litres ;

1 pound = 0.373 kg ; 1 ounce = 31.103 g

5. أذكر العبارات الصحيحة و صحح أي خطأ فيها :

Y := 10.2 mod 4 + sqr (sin (x));

Sqrt (x) := trunc (sqr (x));

Z := round (10) + trunc (5.5) ;

W := abs (-12.5) * trunc (1E18) ;

15 div 4 := 3 mod 3.3

6. أكتب العبارة expression التالية في لغة باسكال :

$X_1 = (-b + \sqrt{b^2 - 4ac})$;

$2x^4 + 4x^2 + 6x + 10 = 0$

$\Delta = b^2 - 4ac$

surface = $\Pi * R^2$

7. أكتب برنامجا يحسب مساحة مثلث ما حسب القاعدة التالية :

T = (C1 + C2 + C3) / 2. أضلاع المثلث : C1, C2, C3

$$\text{surface} = \sqrt{T(T-C1)(T-C2)(T-C3)}$$

8. لحل نظام المعادلات الخطية مثل :

$$\begin{cases} aX + bY = c \\ dX + eY = f \end{cases}$$

$$X = af - cd / ae - bd$$

نستعمل القاعدة التالية :

$$Y = af - cd / ae - bd$$

أكتب البرنامج الذي يقرأ a, b, c, d, e, f و يعرض قيمة x, y

9. أجريت تجربة لمعرفة قوة الجاذبية في مدينة كندية كالتالي : تلقى كرة حرة من أعلى برج للسقوط و يسجل الارتفاع و الزمن الذي يمر إلى وصولها على سطح الأرض فكانت النتائج كما يلي :

الارتفاع y بالمتر	الزمن t بالثانية
73.548	3.74
121.5	4.84
230.04	6.64
137.052	5.13
151.192	6.11

أكتب البرنامج الذي يحسب قوة الجاذبية g باستعمال المعادلة :

$$y = 1/2gt^2 \quad g \text{ أحسب لكل تجربة ثم معدل النتيجة لـ } g \text{ و بشروح .}$$

10. طلب منك طبيب للأطفال أن تكتب له برنامجا يساعده على معرفة سن الأطفال

بالشهر بناء على معرفة تاريخ الازدياد و التاريخ الذي تقدم فيه إلى الطبيب مثلا :

أدخل الطبيب : ← 6 1978
← 3 1980

عرض البرنامج : né le :06 1978

Date: 03 1980

Age : 21 mois

الباب الرابع : الإجراءات و الدوال .

Chapitre 4:Les fonctions et procédures .

المصطلحات الجديدة :

المتغير المحلي : variable locale - local variable

المتغير الشامل : variable globale - global variable

مجال المتغير : portée des variables-scope of variable

تقديم الوسيط : transmission de paramètre;-passing parameter

الشكلي : paramètre formel-formal parameter

الوسيط الفعلي : paramètre effectif - actual parameter

1. الهدف

1.But

في هذا الباب سنتطرق إلى طرق بناء الإجراءات و الدوال في باسكال.

و هي أهم الخطوات لبناء خوارزمية algorithmه حل مسألة ما .

2. الإجراءات و الدوال وسيلة بنيوية :

2.Les procédures et fonctions = méthode

structurée

تهدف لغة باسكال إلى تعليمك طرح حلول بنيوية أي وضع حل لكل مسألة لبنة لبنة حتى

يكتمل الحال الشامل : و هي مبادئ تجدها مفصلة في كتاب solving problems of Polia

الرياضي الشهير نذكر هنا بعضها : لبناء حل مسألة ما ..أنظر ملخص لها في آخر الكتاب .

و قبل ذلك نذكر بتعريف الخوارزمية : سمي حل المسألة بالخوارزمية Algorithmه و ذلك

لأن الكلمة algorithmه مشتقة من اسم الرياضي العربي الشهير الخوارزمي الذي كان له

الفضل في طرح لأول مرة حلولاً للمعادلات المتعددة المجاهيل بطرق مبوبة و بتدرج فأصبحت

هذه المنهجية لطرح الحلول تسمى على اسمه : algorithmه الخوارزمية .

كيف تبني خوارزمية حل مسألة :

أولا : يجب أن تفهم المسألة = المجهل ؟ ما المعطيات ؟ ما الشروط ؟ ضع الرموز و

التعاريف المناسبة لكل من المجهل و المعطيات .

ثانيا : ابتكار الخطة :أوجد الربط بين المجهل و المعطيات .قسم الحل إلى حلول جزئية . جد

العلاقة بين الحلول الجزئية لتكوين الحل الشامل.

ثالثا : حسن الحل تدريجيا :جودة التعامل مع الدخل و الخرج

رابعا : ضع الشروح و التعاليق لفهم كل خطوة و استعمال كل جزء منه من طرف أي

مستعمل فالإجراءات و الدوال هي وسيلتك لكتابة الحلول الجزئية ثم التعبير عن الحل

الشامل بكل سهولة و بناء مكتبة اللبنت و الأشياء التي تحتاجها باستمرار.

3. الإجراءات .

3.Les procédures

1.3 القاعدة النحوية .

3.1 Règle syntaxique

Procédure name (قائمة الوسطاء parameter list) ;

Liste des paramètres

Label

Const

Type

Var

Procedure ... Function ...

Begin { of procedure: name }

....

End;

قسم التصريحات و هو يشبه

قسم تصريحات البرنامج

قسم التعليمات

كما تلاحظ الإجراءات برنامج فرعي sous programme تجد فيه جميع خواص البرنامج :

لكل إجراء تعريف و هو اسمه ثم بين قوسين قائمة الوسائط parametres و هي مجموعة متغيرات . إن سبق الوسيط بالكلمة الخاصة var يسمى وسيط مقدم بالعنوان call by reference أما إن كان مصرح به بدون var يسمى وسيط مقدم بالقيمة call by value أما إن كانت قائمة الوسائط فارغة فهو إجراء بدون وسيط .

2.3 الإجراء بدون وسيط .

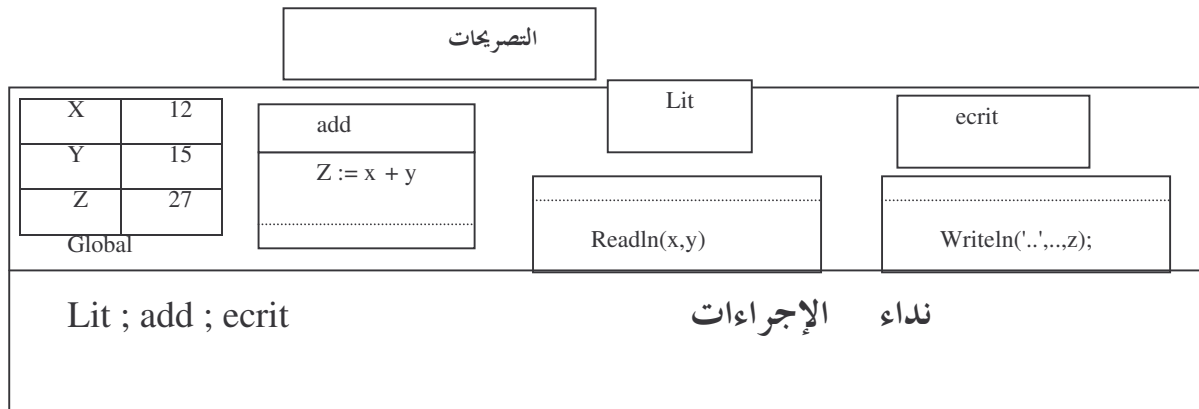
3.2 Procédure sans paramètre

الإجراء بدون وسيط هو كل إجراء لا يوفر نافذة لتبادل الدخل و الخرج منه و إليه فهو يستعمل المتغيرات الشاملة و المحلية و ينادى باسمه فقط .
مثال رقم 1:

```

program global ;
uses wincrt;
  Var x , y , z : real ;
  Procedure add ;
    Begin
      Z := x + y ;
    End;
  Procedure lit ;
    Begin
      Write('entrer 2 Nb.:'); Readln ( x, y ) ;
    End;
  Procedure ecrit ;
    Begin
      Writeln ( 'la somme de',x,'+',y,'=',z);
    End;
  Begin { programme principal }
    Lit ;
    Add;      } نداء الإجراء appel de procedure
    Ecrit ;
  End.{ fin du prg. Princ.}

```



نبدأ بنداء الإجراء lit و هو قراءة قيمة x و y حيث يكتب السؤال entrer 2 Nb. فيضرب المستعمل 12 و 15 ثم نداء الإجراء add الذي يجمع x و y ثم نداء للإجراء écrit الذي يكتب النتيجة

Entrer 2 Nb.: 12 15 ↵

La somme de 12 + 15 = 27

في هذا المثال بينا كيف نستعمل المتغير الشامل بالإجراءات حيث نرى أنه بداخل قسم التصريحات العام بالبرنامج فهو يشمل جميع ما بداخل هذا القسم .
مثال رقم 2 :

```

program local;
uses wincrt;
var a , b , c : integer ;
  procedure lit ;
    var a, b, c : integer ;
    begin
      readln ( a , b ) ;    c := 10 ;
    end;
  procedure écrit ;
    begin
      writeln ( a , b , c ) ;
    end;
begin { local }
  lit ;  a := 5 ; b := 50 ; c := a + b ; écrit;
end.{ local }

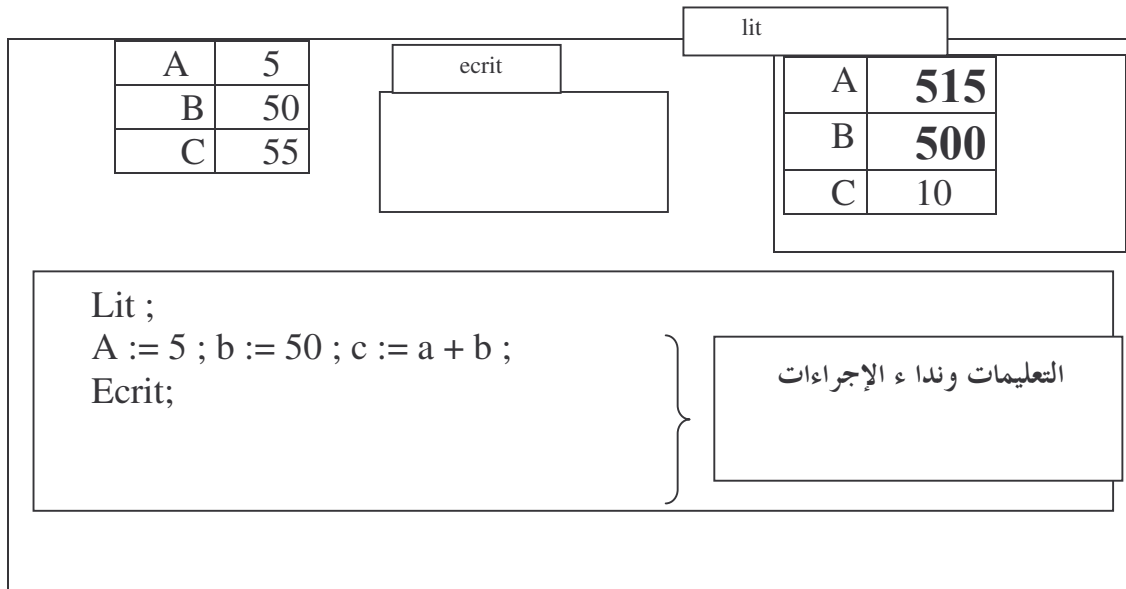
```

لنتابع التنفيذ بالذاكرة المركزية:

نبدأ ببدء الإجراء lit : يقرأ عددين من الدخل في a,b لنفرض ضرب المستعمل 515 و 500 ثم يضع 10 في c وهي المتغيرات المحلية بالإجراء lit. ثم 5 في a و 50 في b و a+b في c وهي المتغيرات الشاملة a,b,c بالبرنامج.

ثم تنفيذ إجراء الكتابة lit يعرض على الشاشة محتوى a,b,c الموجودة بالبرنامج أي المتغيرات الشاملة لا التي بداخل الإجراء lit فهي غير موفرة خارج إطارها و تصبح مهمة inconnnue,unknown مباشرة بعد نهاية الإجراء lit .

فإذا أردنا أن نتابع التنفيذ في الذاكرة المركزية mémoire centrale يكون كالتالي :



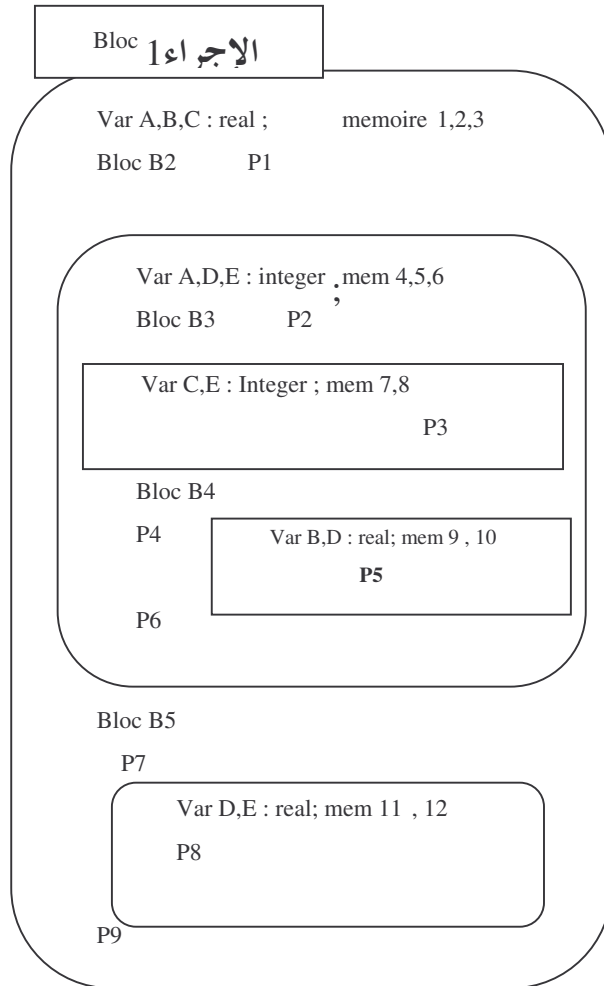
أما إجراء الكتابة écrit فسيعرض على الشاشة قيم a,b,c الشاملة أي القيم 5 و 50 و 55 لا القيم 515 و 500 و 10 التي بقيت داخل الإجراء lit.

قاعدة استعمال المتغير المحلي و الشامل :

في المثال الموالي نبين قواعد استعمال المتغير المحلي و الشامل :

أنظر الجدول و التصريحات في الذاكرة :

في المكان At position	يمكن استعمال المتغيرات التالية (with there memory locations)can be used
P1	A(1),B(2),C(3)
P2	A(4),B(2),C(3),D(5),E(6)
P3	A(4),B(2),C(7),D(5),E(8)
P4	A(4),B(2),C(3),D(5),E(6)
P5	A(4),B(9),C(3),D(10),E(6)
P6	A(4),B(2),C(3),D(5),E(6)
P7	A(1),B(2),C(3)
P8	A(1),B(2),C(3),D(11),E(12)
P9	A(1),B(2),C(3)



1. المتغير الشامل global هو كل متغير صرح به في البرنامج الرئيسي أو في إجراء يحتوي على إجراءات . يمتد مجاله إلى جميع أجزاء البرنامج المصرح فيه و تبقى قيمته موفرة للاستعمال .

2. المتغير المحلي local هو كل متغير يصرح به في قالب الإجراء . فهو محلي بالإجراء لا يستعمل إلا داخله و لا يتجاوزه .

3. عندما يكون اسم المتغير المحلي و الشامل متشابهين تكون أسبقية الاستعمال داخل الإجراء للمتغير المحلي .

نلخص هذه القواعد في صورة القوالب الممثلة للإجراءات :

في B1 عندنا 3 متغيرات محلية A,B,C و لا واحد شامل .
 في B2 المتغير A الذاكرة رقم 4 وD وE متغيرات محلية وB,C شاملة
 في B3 : C,E محلية . A(4),B(2),D(5) شاملة global
 في B4 : B,D محلية . A(4),C(3),E(6) شاملة.
 في B5 : D,E محلية . A(1),B(2),C(3) شاملة.

3.3 الإجراء بوسيط.

3.3 Procédure avec paramètre

الوسيط في باسكال يكون من ثلاث أنواع :

الوسيط المقدم بالقيمة value parameters/par valeur

الوسيط المقدم بالعنوان reference parameters/par adresse

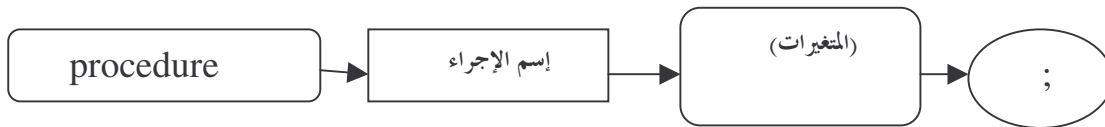
الوسيط كإجراء أو دالة procedural and functional parameters

أو النداء بالاسم call by name .

1.3.3 النداء بالقيمة

3.3.1 Transmission par valeur. call by value

القاعدة : كل إجراء بوسيط مقدم بالقيمة أو ينادى بالقيمة يكتب حسب قواعد الإجراءات بالإضافة إلى قائمة المتغيرات في رأس الإجراء بدون الكلمة الخاصة var .



المثال 1:

```

Program call_value;
  Uses wincrt;
  Var x, y, z : real ;
  
```

```

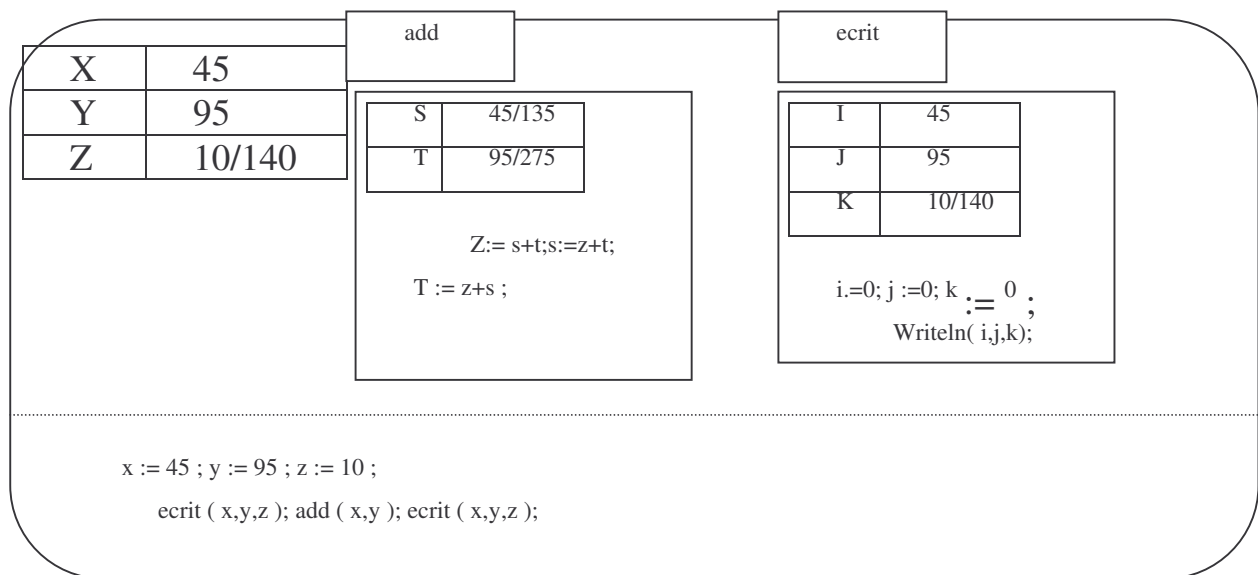
Procedure add ( s , t : real ) ;
  Begin
    Z := s + t ; s := z +t; t := s + z;
  End;
Procedure ecrit ( i ,j ,k : real ) ;
  Begin
    I := 0 ; j := 0; k := 0 ;
    Writeln ( i,j,k );
  End;
Begin { prog call_value }
  X := 45 ; y := 95 ; z := 10;
  Ecrit ( x,y,z) ;
  Add ( x,y);
  Ecrit ( x,y,z);
End.{ fin du prog }

```

نداء الإجراءات بالوسيط الفعلي

para.effectif x,y,z

لتابع التنفيذ :



لاحظ أن المتغيرات المصرح بها في رأس كل إجراء كوسطاء حضر لها ذاكرة في كل إجراء

خاص بها، فهي محلية local بالإجراء .

نداء الإجراء $ecrit(x,y,z)$ ينقل قيم x,y,z الوسطاء الفعليين ou paramètres effectifs

réels إلى الوسطاء الشكليين، ثم يكتب على الشاشة قيم x,y,z وهي 0 0 0 لأن داخل

الإجراء تم تغيير i, j, k و تعيين 0 في كل منها . نداء الإجراء $add(x, y)$ ينقل قيم x, y إلى s, t ثم تقوم بالجمع :

في z تصبح القيمة 140 و في s تصبح 95+140 و في t تصبح 135+140 و الخروج من الإجراء يهمل قيم s, t و لا تكون متوفرة خارجه لذلك نداء الإجراء $ecrit(x, y, z)$ يكتب القيم 0 0 0 و لو أنها نقلت محتوى x, y, z إلى المتغيرات الشكلية i, j, k و لكن قيمها داخل الإجراء غيرت فأصبحت 0.

الخلاصة : كل أنواع المتغيرات يمكن استعمالها في النداء بالقيمة إلا النوع $file$ أي الجذاذة لا يمكن تقديمها كوسيط بالقيمة . النداء بالقيمة يعني تقييم الوسيط الفعلي $actual$ $parameter$ من اليسار إلى اليمين و تعيينه في الذاكرة المخصصة له كوسيط شكلي و اعتباره متغير محلي بالإجراء .

و بطبيعة الحال يجب أن يكون نوع المتغير الفعلي و نتيجة تقييمه من نفس نوع المتغير الشكلي .

2.3.3 الإجراء بوسيط مقدم كعنوان .

3.3.2 Call by reference/paramètre passé par adresse

القاعدة : $procedure\ name\ (var : parameter\ list) ;$

$procedure\ body$

في هذا النوع من الإجراءات نتبع نفس قواعد كتابة الإجراء إلا أن الوسيط الشكلي $formal\ parameter$ يجب أن يصرح بالكلمة الخاصة var . و هذا يعني أن الوسيط الفعلي $actual\ parameter$ يجب أن يقدم كمتغير و لا يجوز أن يكون عبارة $expression$.

المتغير الشكلي هنا لا يحضر له ذاكرة فهو إذا ليس بمحلي بالإجراء , يقدم له عنوان ذاكرة الوسيط الفعلي ليصبح عنوان واحد باسمين : واحد شكلي فقط و الآخر فعلي , عليه تتم كل العمليات .

في هذا النوع من النداء يمكن إرجاع قيمة أو أكثر خارج الإجراء و يمكن إدخال
تغييرات على البرنامج أو الإجراء الرئيسي .

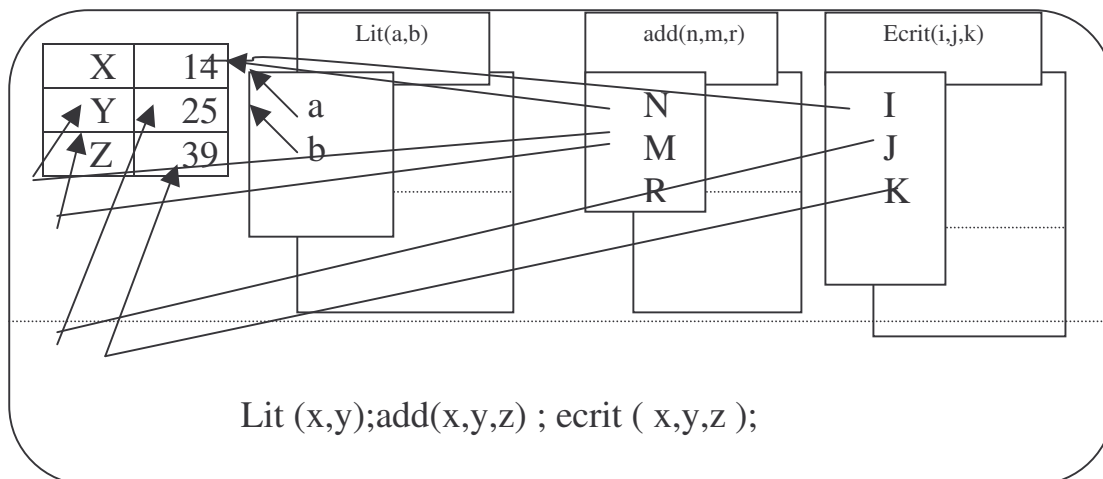
مثال رقم 1 :

```

program call_reference ;
uses wincrt;
  Var x,y, z : real ;
  Procedure lit ( var a,b : real );
    Begin
      Readln( a , b );
    End;
  Procedure add ( var n , m ,r : real );
    Begin
      r := n +m ;
    End;
  Procedure ecrit ( var i,j,k : real );
    Begin
      Writeln('la somme de ',i,j,'=',k);
    End;
  Begin { debut call_reference }
    Lit ( x,y );
    Add( x,y,z );
    Ecrit (x,y,z);
  End.{ fin call_reference}

```

ضرب المستعمل 14 25

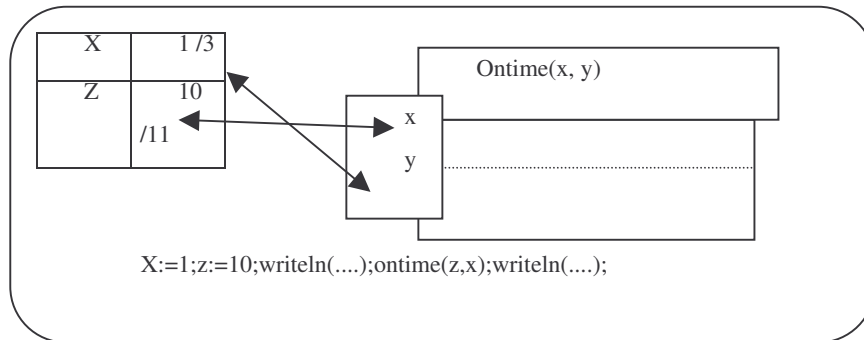


كما تلاحظ كل العمليات تمت على الوسيط الفعلي الذي حضر له ذاكرة أما الوسيط الشكلي فهو أسم يُربط باسم المتغير الفعلي . و هكذا يكون الإجراء برنامج فرعي يتدخل في أي مكان في البرنامج كلما نادينا به .
مثال رقم 2:

```

program time;
uses wincrt;
  Var x,z : integer ;
  Procedure onTime ( var x,y: integer );
    Begin
      X := x + 1 ; y := y + 2 ;
    End;
  Begin
    X := 1 ; z := 10 ;
    Writeln( ' early time ' , x,z );
    OnTime ( z , x );
    Writeln ( ' Late Time ' , x , z );
  End.

```



في هذا المثال عندنا الوسيط الشكلي x في الإجراء onTime نجد اسمه كمتغير فعلي و لكن عند النداء تم تقديم الوسيط الفعلي z للوسيط الشكلي x و الوسيط الفعلي x للوسيط الشكلي y . حيث علينا إتباع قاعدة التابع الأول بالتالي والتالي بالتالي ... عند النداء .

Early time 1 10
Late time 3 11

فيكون التنفيذ على الشاشة :

الخلاصة:

الربط بين الوسيط الشكلي و الفعلي يتم خلال التنفيذ أي عند النداء.
 نوع بيان type de donnée الوسيط يجب أن يكون مصرح به قبل استخدامه فهو إما من
 الأنواع الموفرة أو التي يصرح بها المبرمج في قسم التصريح بالنوع , و أن يكون من نفس
 النوع عند النداء و في نفس الترتيب حتى يتم الربط المناسب بينهما.

1.3.3. النداء بالاسم .

3.3.1 Call by name or procedural and functional parameters

هذا النوع من الوسيط المقدم كإجراء أو دالة يستعمل في الظروف التالية:

في التحليل العددي numerical analysis

في النداء للبرامج الفرعية المكتبة calling library routines

في التطبيقات المعقدة special applications .

في التطبيقات الخاصة بالتحليل العددي نحتاج مثلا إلى تقديم دالة كوسيط لإجراء

يبحث عن التفاضل بين حدين integral between limits أو القيمة العليا و الدنيا maximum
 and minimum value إلى آخره ...

و في خوارزميات الذكاء الاصطناعي artificial intelligence

مثال 1 : أكتب دالة تحسب الظل و ظل التمام لزاوية ما , على أن تقدم الدالة sin و cos

كوسيط في هذه الدالة .

في ترجمان BPW يقدم الوسيط كإجراء أو دالة بعدما تصرح به في قسم التصريحات بالنوع
 و هي إضافة مهمة حيث تجعل الدالة و الإجراءات نوع جديد.

حيث نصرح بـ Trig من نوع Function كنوع جديد .

```
program callbyFunction; {F+}
```

```
uses winCRT;
```

```
type trig = function ( x : real ) : real;
```

```

var t , x : Real;

    n , d : trig;{ n,d de type fonction déclaré}

function tang( n,d :trig ):real;
    begin
        tang := n( x )/d(x);
    end;

function s( x : real ):real;
    begin
        s := sin(x);
    end;

function k( x : real ):real;
    begin
        k := cos(x);
    end;

begin
    write('entrer la valeur de l"angle');

    readln( x);
    x := x * 3.14/180;
    writeln('le sin est ',s(x):8:2);
    writeln('le cos est ',k(x):8:2);
    t := tang (s, k);{parametres: fonction}
    writeln('la tangante de l"angle est=',t:8:2);

    writeln('la cotangante est = ',tang(k,s):8:2);

end.

```

مثال 2:

Exemple de passage de paramètre de type fonction dans une procédure
ou Fonction { \$F+ } { For programs that use procedural variables,
all such procedures must use the FAR code model. }

```

program ProcVar;
  uses wincrt;
type
  IntFuncType = function (x, y : integer) : integer;
  { No func. identifier }
var
  IntFuncVar : IntFuncType;
procedure DoSomething(Func : IntFuncType; x, y : integer);
begin
  Writeln(Func(x, y):5);    { call the function parameter }
end;
function AddEm(x, y : integer) : integer;
begin
  AddEm := x + y;
end;
function SubEm(x, y : integer) : integer;
begin
  SubEm := x - y;
end;
begin
  { Directly: }
  DoSomething(AddEm, 1, 2);
  DoSomething(SubEm, 1, 2);
  { Indirectly: }
  IntFuncVar := AddEm;      { an assignment, not a call }
  DoSomething(IntFuncVar, 3, 4); { a call }
  IntFuncVar := SubEm;      { an assignment, not a call }
  DoSomething(IntFuncVar, 3, 4); { a call }
end.

```

4.الدوال.

4.Functions.les fonctions

1.4 تعريف الدالة :

4.1 Définition d'une fonction

الدالة برنامج فرعي sous-programme, subroutine تطبق عليها جميع القواعد التي

ذكرت في حق الإجراء : مجال المتغير, تقديم الوسيط بالقيمة و العنوان . و لكن تختلف عن الإجراء في شيء واحد : الدالة ترجع قيمة واحدة في مكان واحد . قيمة قابلة للإستعمال في البرنامج أو الإجراء أو الدالة أين يتم ندائها . الدالة ليست تعليمة كالإجراء بل تكون جزءاً من عبارة يعين لها مكان ترجع فيه النتيجة .

تستعمل الدالة التي نصح بها كالدالة القياسية fonction standard مثل $\sin(..)$ أو
 eof(..)

2.4 القاعدة النحوية .

4.2 Syntax

Function name (parameter list) : data type ; نوع البيان الذي سترجعه

قسم التصريحات المحلي
 const..type..var..procedure..function
 begin

التعليمات actions

آخر تعليمة في الدالة : Expression := إسم الدالة Name

End; { نهاية التصريح بالدالة }

عند التصريح بالدالة يجب أن تكون آخر تعليمة فيها تعيين القيمة التي سترُجعها إلى إسم الدالة , حيث يعتبر إسمها كأنه ذاكرة توضع فيها النتيجة التي توصلت إليها , لذلك نقول أنها تُرجع قيمة . retourne une valeur . نوع هذا البيان يكون
 real,integer,char,boolean

على المستوى القياسي standard و لكن على المستوى الإضافي extend level يمكن للدالة أن ترجع أي نوع بيان بسيط simple أو بنيوي structured type .

أمثلة :

1. أكتب دالة تحسب ظل الزاوية. tangante d'un angle.

```
Function tang ( z : real ) : real ;
```

```
Begin
```

```
    Tang := sin( z ) / cos ( z ) ;
```

```
End;
```

2. أكتب دالة تقييم جواب المستعمل على سؤال تطرحه فترجع صحيح أم خطأ

false ..True

```
Function test ( choix : char ) : Boolean ;
```

```
Begin
```

```
    Write ( ' v.v une autre execution o(ui) ou n(on)?' );
```

```
    Readln ( choix ) ;
```

```
    Test := choix in [ 'n','N' ] ;
```

```
End;
```

لنستعمل الدالتين في برنامج يحسب ظل زاوية ما .

```
Program exemple_fonction;
```

```
Uses wincrt;
```

```
Var x : real ;
```

```
Function tang ( z : real ) : real ;
```

```
Begin
```

```
    Tang := sin ( z ) / cos ( z ) ;
```

```
End ;
```

```
Begin { exemple_fonction }
```

```
    Writeln( ' prog. de calcul de la tangente d'un angle ' );
```

```
    Write ( ' entrer la valeur de l'angle : ' );
```

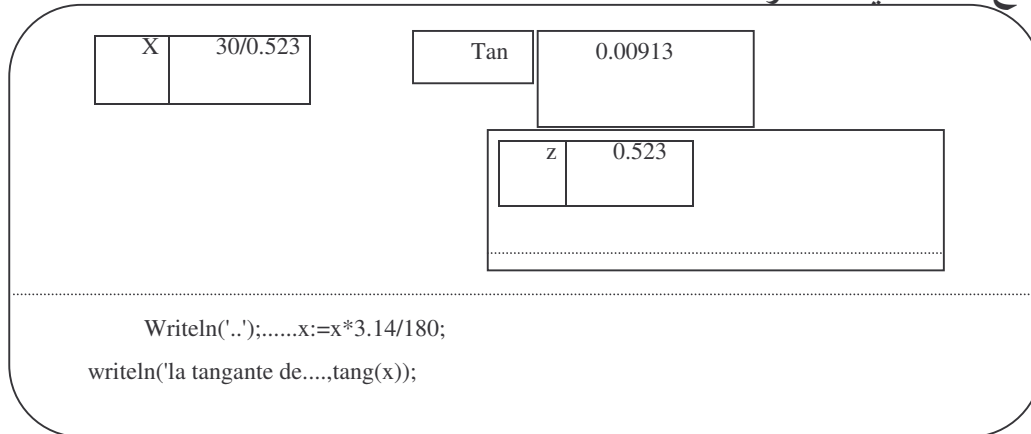
```
    Readln ( x ) ;
```

```
    X := x * 3.14 / 180 ; { التحويل إلى الراديان }
```

```
    Writeln( ' la tangente de ',x,' est = ',tang ( x ) );
```

```
End.
```

لنتابع التنفيذ في الذاكرة .



إذا ضرب المستعمل قيمة x : 30 تحول إلى الراديان ثم نداء الدالة $\text{tang}(x)$ بتقديم الوسيط بالقيمة par valeur فنقل قيمة x إلى z , تحسب قيمة الظل وترجعها في الذاكرة الخاصة بالدالة و التي عيناها أمام اسمها , لنبين بأنها ليست ذاكرة محلية بالدالة و لكن ذاكرة موفرة خارجة الدالة لتكون القيمة التي ترجعها قابلة للاستعمال من كل تعليمة تطلبها.

Prog.de calcul de la tangente ...
 Entrer la valeur de l'angle : 30
 La tangente de 30 est = 0.00913

5.الدوال و الإجراءات الأمامية .

5.Fonction et procédure Forward

لنظرب مثال نوضح به الحاجة إلى الإعلان الأمامي directive forward

```
Procedure P1 ( ...parameters ..) ;
Begin
    P2 ( .....) ; { نداء الإجراء P2 }
    .....
end ; { fin de P1 }
procedure P2 ( .... parameters ....) ;
begin
    . ... P1 ( .....);
end;{ fin de P2 }
```

في الإجراء P1 نداء للإجراء P2 قبل التصريح به و هو أمر غير مسموح به في باسكال حيث أن P2 لم تعرّف بعد للترجمان حتى يمكنه الربط بينها و بين P1 , أما في P2 فنداء P1 مقبول لأنه معرف قبل P2 . حل إشكال نداء P2 في P1 نقوم بإعلان مسبق لـ: P2 بالكلمة الخاصة Forward كالتالي :

```
Program .....;
  Procedure P2 ( ..... ) ; Forward
  Procedure P1 ( ..... ) ;
    Begin
      ....
      P2 ( .... ) ; { نداء P2 }
    End ;
  Procedure P2 ;
    { نكمل التصريح بـ: P2 بدون ذكر الوسطاء }
    Begin
      ....
      P1 ( ..... ) ;
    End;
```

و هكذا يكون P2 معرف مسبقا بالتصريح برأس الإجراء فقط و بالإعلان Forward نصح الخطأ و يمكن لنا إستعمال أي إجراء أو دالة في أي إجراء أو دالة أخرى قبل أن ننهي التصريحات كاملة.

6. الإجراءات والدوال التراجعية.

6.Fonctions et procédures récursives

الشيء التراجعي هو الشيء الذي يعرف كله من نفسه أو جزءاً منه يعرف من نفسه

كذلك .

an object is said to be recursive if it partially consists or is defined in terms of itself .

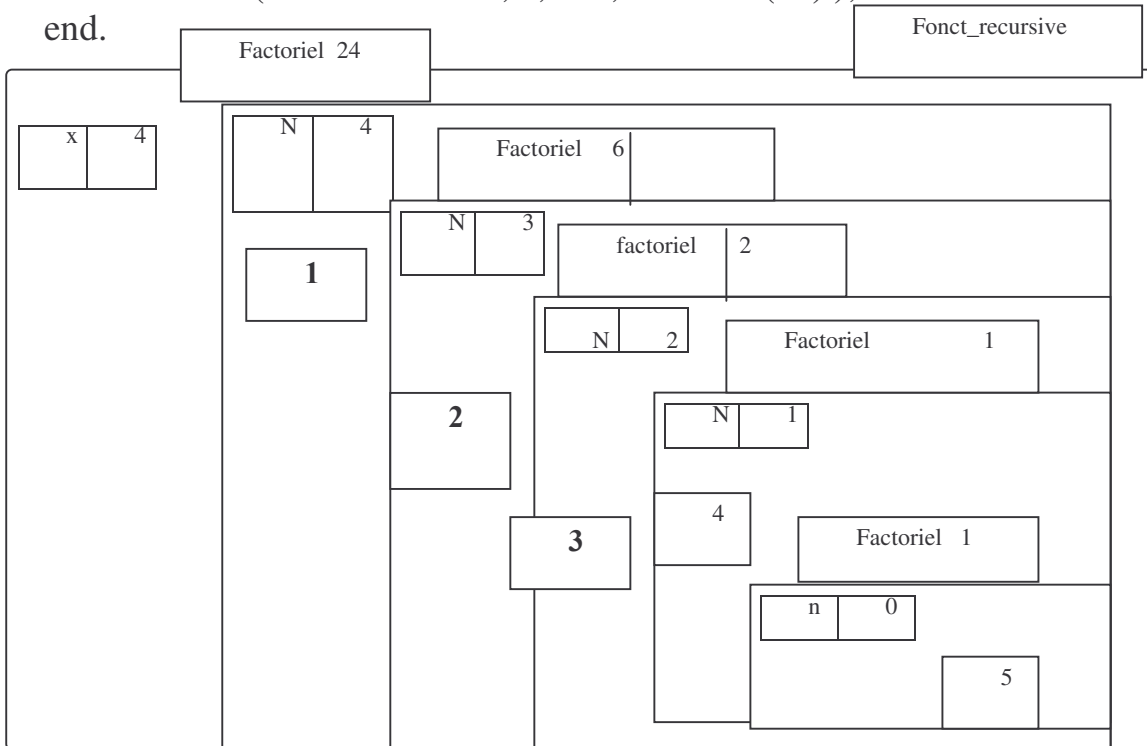
أكتب دالة تحسب عاملي N! . Factoriel de N!

معروف إن $N * (N-1)! = N!$. إذا تكون الدالة كما يلي :


```

program fonct_recursive;
  uses wincrt;
  var x : integer ;
  function factoriel ( N : integer ) : integer ;
    begin
      if n = 0 then  factoriel := 1
      else  factoriel := N * Factoriel ( N-1 );
        (* هنا تم النداء التراجعي للدالة *)
      end ;
    begin { Fonct_recursive }
      writeln(' ce prog. Calcul le factoriel d" un nombre N ? : ');
      readln ( x );
      writeln(' le factoriel de ',N,'est ', factoriel ( x ) );
    end.

```



لاحظ كيف تتم عمليات النداء التراجعي حيث يبدأ بنداء الدالة و تقديم الوسيط بالقيمة -تنقل قيمة x إلى N مثلا 4 ثم نداء الدالة مرة ثانية و تقديم الوسيط الفعلي 1-4 قيمة 3 =N-1 -تنقل إلى N.

ثم يبدأ تنفيذ الدالة إذا $N=0$ إذا ضاع في factoriel 1 و إلا اضرب N في ما ترجعه الدالة بالوسيط $N-1$ و بما أن N تساوي 2 يتم إذا نداء الدالة بـ $N=2$. و هكذا يتواصل نداء الدالة بـ $N=1$ ثم نداء آخر بـ $N=0$ و هو الأخير حيث ترجع الدالة قيمة 1. و يبدأ التراجع من ذلك النداء 5 فالـ 5 يرجع إلى القيمة 1 و الـ 4 يرجع إلى القيمة 3 و الـ 3 يرجع إلى القيمة 2 و الـ 2 يرجع إلى القيمة 1 و الـ 1 يرجع في factoriel 6 و هي القيمة التي ستكتب في الشاشة :

Le factoriel de 4 est 24

مثال 2 : أكتب دالة تحسب x^n :

علما بأن $s^n = s * s^{n-1}$. إذا تكون الدالة تراجعية :

```
Function puissance( x : real ; n : integer ):real ;
Begin
    If n = 0 then puissance := 1
    Else puissance := puissance(n-1,x)*x
End;
```

مثال 3 : لإجراء تراجع بسيط يكتب T عدد معين حسب تنفيذه في البرنامج .

```
Procedure print_star ( n : integer );
Begin
    If n = 0 then writeln ;
    Else
        begin
            Write ( 'T ' );
            Print_star ( n-1 );
        end
End;
```

7. الدوال و الإجراءات الخارجية .

7. Procédures et fonctions externes

الدوال و الإجراءات الخارجية هي التي تخزن في جذاذة fichier خارج البرنامج الرئيسي و تسمى المكتبة الخارجية غير المكتبة القياسية الخاصة باللغة standard library و

تربط بالبرنامج بالرابط linker ليتم إحداث الجذاذة المنفذة fichier exécutable . يمكن إحداث الدوال و الإجراءات الخارجية بنفس الكيفية التي صرح بها في البرنامج العادي إلا أنه تكتب بداخل جذاذة مستقلة تسمى الوحدة Unit. فالوحدة Unit وسيلة جيدة لإنشاء مكتبة الدوال و الإجراءات التي تود استعمالها في برامجك كأدوات عمومية .

تتكون الوحدة من ثلاثة قوالب :

قسم البيني Interface

قسم التراكم Implementation

و قسم القيم الابتدائية Initialisation و تنتهي بالكلمة الخاصة End. .

*مثال: إجراء خارجي في وحدة myunit يستعمل داخل برنامج بالإعلان uses myunit فقط .

```
unit myunit;
interface
    procedure pause;
implementation
    procedure pause;
    begin
        writeln(' taper entrer pour continuer');
    readln;
    end;
    BEGIN{ initialization }
END.

program testwithUnit;
uses wincrt, myunit;
var   choix : char;.....
begin
    .....           pause;
    end.
```

****ملاحظة :** عندما تنشأ وحدة يجب أن تخزنها تحت نفس الاسم الذي أعطيته لها في التصريح بها مثل myunit فهو اسم الجداذة التي ستخزن فيها الوحدة myunit save as : ثم تترجمها compile it و لا تنفذها لأنها غير قابلة للتنفيذ .

8.تمارين :

8.Exercices

1. لتكن لدينا الدالة التالية :

```
Function F ( var x, y : integer ) : integer ;
Begin
  Writeln( A,B,X,Y);
  Y := x + y ; A := x + y ;
  writeln ( A,B,X,Y );
  F := A + B ;
End;
```

و لتكن لدينا التعليمات التالية :

```
A := 2 ; B := 5 ; D := 2 ; E := 5 ;
Writeln ( D , E ) ;
C := F ( D , E ) ;
Writeln ( A, B, C, D, E ) ;
```

```
2 5
2 5 2 5
9 5 2 7
9 5 14 2 7
```

النتيجة تكون كما يلي :

A	2/9
B	5
C	14
D	2/2
E	5/7

F	14
X	
y	

2. لتكن لدينا التعليمات التالية:

```
A := 2 ; B := 5 ; writeln ( A , B ) ; C := F ( A+B,B);
Writeln ( A , B , C ) ;
```

إذا كانت الدالة صرحت بوسيط بدون Var أي بتقديم القيمة call by value not by variable لا بالعنوان . قم بالتنفيذ كما بينا لك ذلك في التمرين 1.

الباب الخامس : التعليمات البنوية .

Chapitre 5:Les instructions structurées

1. المقدمة :

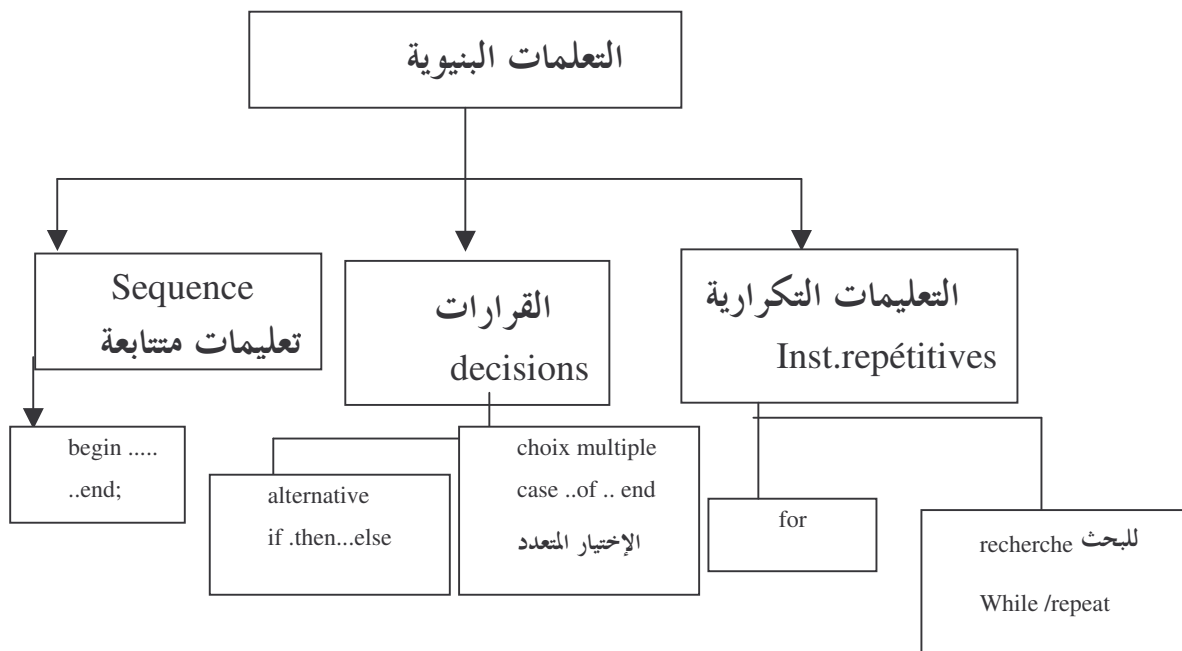
1.Introduction

في هذا الباب نبدأ باستعمال الطريقة الخوارزمية méthode algorithmique لكتابة التعليمات , الإجراءات و البرامج و بالتالي حل المسألة بصفة عامة و المعروفة بطريقة G.N.S : graph of Nassi-Shneidermn .
التعليمات البنوية أو المركبة هي :

التعليمات الاختيارية : les instructions de choix

التعليمات التكرارية: les instructions de répétitions

بالإضافة إلى قالب : bloc: begin.....end لاحتواء كل مجموعة تعليمات



2. التتابع .

2.Séquence d'instruction

كل مجموعة تعليمات متتابعة يجب كتابتها داخل قالب bloc محدد بين Begin و end . وهي قاعدة تتبع في التعليمات البنيوية : كلما كانت عندنا أكثر من تعليمة واحدة وجب احتواء التعليمات في قالب begin...end .

3.الإختيار ضمن حالتين على الأكثر.

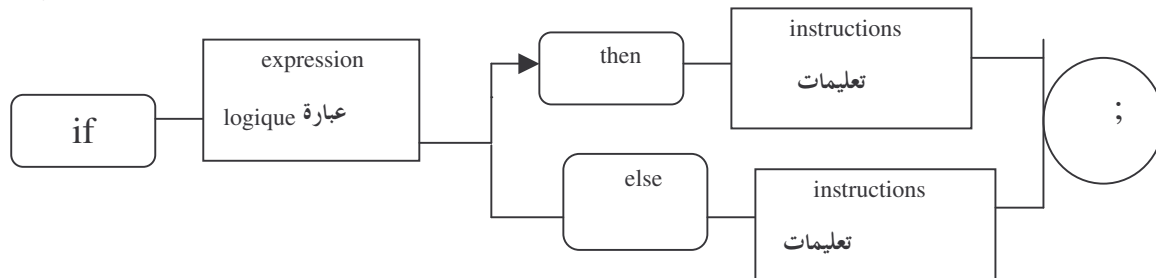
3.Choix parmi 2 cas au plus :If ..Then ..else .

يوفر باسكال تعليمة للإختيار من بين حالتين : ifthenelse و تعني : إذا الشرط صحيحاً إذا ..نفذ التعليمة أو التعليمات التي عينت بعد then و إلا أي كان الشرط خطأ نفذ التعليمات التي عينت بعد else .

1.3 قاعدة التعليمة الشرطية

3.1 Syntaxe

syntaxe de l'instruction conditionnelle.



If «condition» Then : ومعناها :

Instruction تعليمة

Else

Instruction تعليمة ;

If « condition » Then أو :

begin

Inst1;.....

```

end
Else
Begin
Inst.1;.....
end;

```

و القاعدة الثانية هي أن النقطة الفاصلة (;) ممنوعة قبل else . ولا تكتب إلا في النهاية أي في آخر تعليمة بعد else . و بالطريقة الخوارزمية G.N.S نعبر عن تعليمة if كما يلي :

If condition	إذا الشرط
false خطأ	true صحيح

و هي طريقة أسهل من الأسلوب القديم المسمى flow chart - organigramme الخريطة الإنسيابية.

في هذا القالب الخاص بـ if: نقرأ كما يلي : إذا نتيجة تقييم العبارة المنطقية صحيح true تنفذ التعليمات التي كتبت على اليمين أي تحت true و إذا كانت النتيجة خطأ false تنفذ التعليمات التي على اليسار أي تحت false .

ملاحظة: يمكن ترك قالب then أو قالب else فارغا بدون تعيين تعليمات فيه و لكن مع احترام القاعدة العامة.

2.3 أمثلة :

3.2 Exemples

1. أكتب إجراء يسأل عن إسم المستعمل إن كان إسمه عبد الله ردّ عليه بالتحية : السلام عليكم وإلا لا يجيبه تماما .

```

Procedure salute ;
  Var rep : char ;
  Begin

```

```

    Write(' هل إسمك عبد الله ن(عم) أم (لا) ');

```

```

    Readln ( rep );

```

```

    If rep = 'ن' then

```

```

        Writeln(' السلام عليكم و رحمة الله ');

```

```

    { في هذا المثال قالب else فارغ }

```

```

  End;

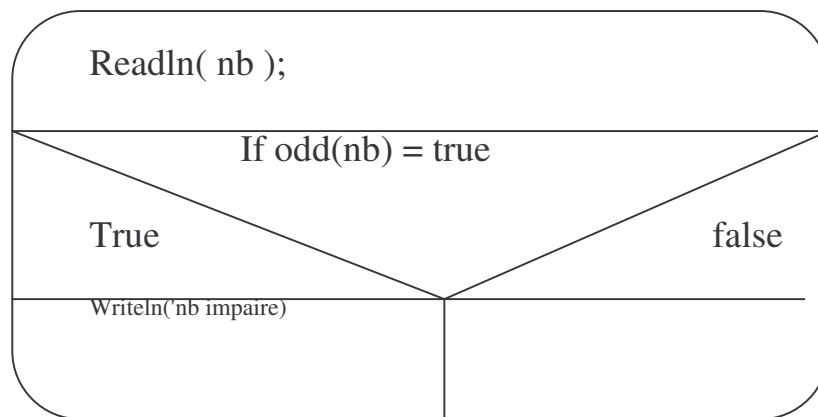
```

ملاحظة: يمكن ترك قالب else فارغا و بالتالي نضع النقطة فاصلة (;) لإنهاء التعليمة .

2. أكتب إجراء يقرأ عددا ما و يرجع الجواب عدد وتري أو فردي إن كان العدد فردي و

إلا لا يرجع جوابا .

نكتب الخوارزمية l'algorithmme بأسلوب G.N.S :



3. أكتب نفس الإجراء و لكن للإعلان هل العدد زوجي أم لا .

الحل : نكتب نفس الخوارزمية و لكن بقالب true فارغا و بقالب else به تعليمة

writeln(' nb paire ') . و يكون الإجراء كما يلي :

```

Procedure pair ( var nb : integer );

```

```

  Begin

```

```

    Write(' entre un Nb: ');      Readln ( nb );

```

```

    If odd (nb) = true then { vide }

```



```

Else
    Writeln(' nb. Paire ');
End;

```

بطبيعة الحال يجب أن تنادي الإجراء بوسيط يقدم كعنوان مثلا :

التصريح بالوسيط الفعلي

```

Var x : integer ;

```

.....

نداء الإجراء بالوسيط الفعلي

```

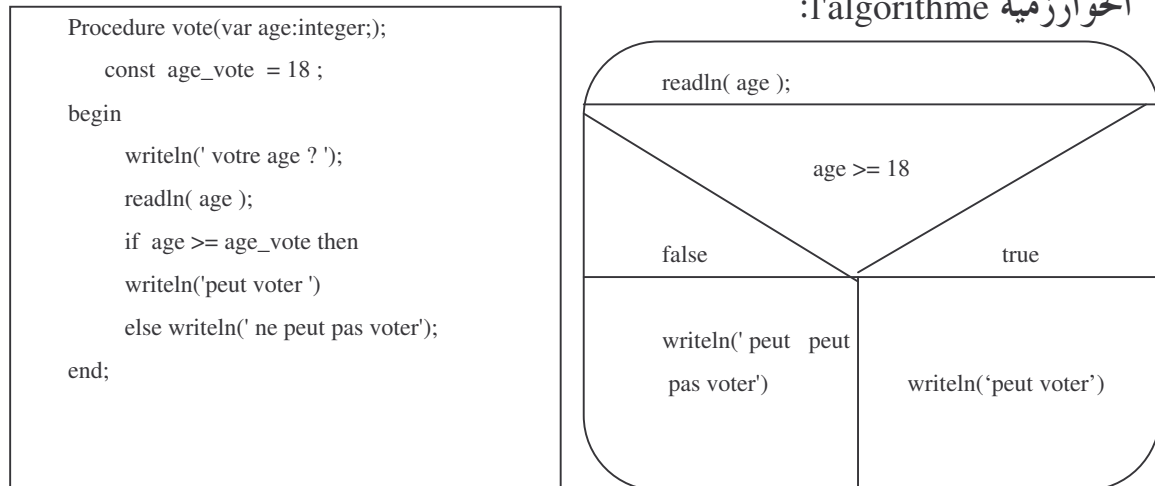
pair ( x ) ;

```

ملاحظة: هنا قالب then فارغ و لكن مازالت التعليمة لم تنتهي بعد و بالتالي لا نضع نقطة فاصلة بعد then و لكن نضعها في نهاية التعليمة بعد else و هي علامة نهاية if .

4. أكتب إجراء يختبر سن المواطن و يرجع هل ينتخب أم لا .

الخوارزمية: l'algorithme:



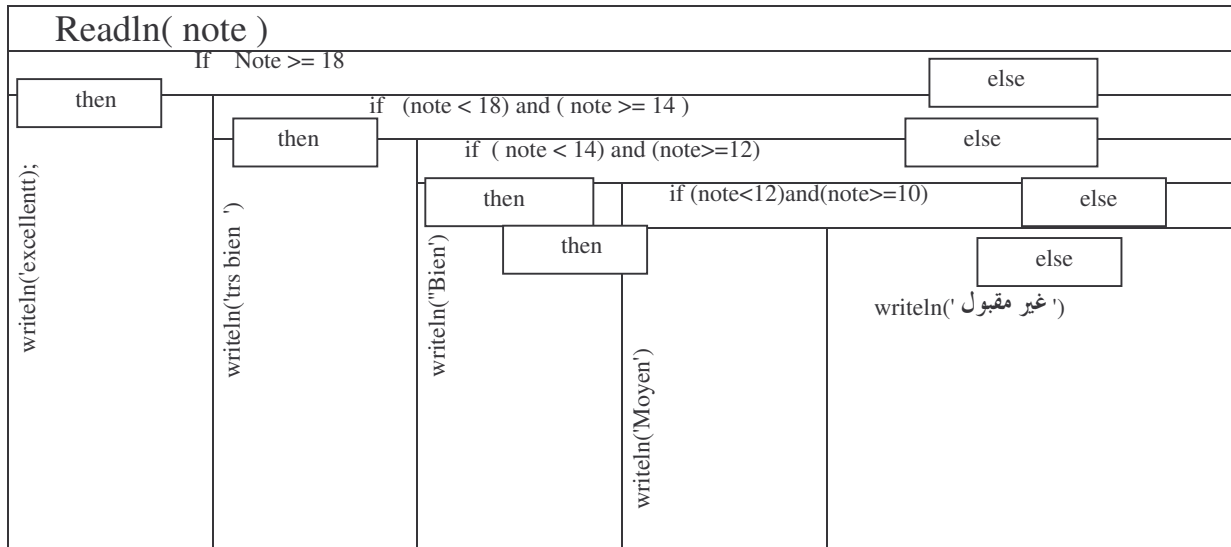
5. أكتب خوارزمية و إجراء يكتب ملاحظات الأستاذ لكل تلميذ حسب الجدول التالي :

إن كانت النقطة من 18 .. 20 : ممتاز 16 .. 18 : جيد جدا 14 .. 16 : جيد

12 .. 14 : حسن 10 .. 12 : متوسط أقل من 10 : غير مقبول .

الخوارزمية: l'algorithme:

أنظر إلى أسلوب قوالب G.N.S كيف تسمح لك كتابة الخوارزمية بشكل واضح و بنيوي و هو ما يتفق مع لغة باسكال التي وضعت لتعليم البرمجة البنيوية .



ملاحظات: في هذا المثال استعملت تعليمة if متداخلة الواحدة تلو الأخرى nested if
 ..then..else , و أدخل عامل الربط المنطقي opérateur logique و أي and وبالتالي نتيجة
 تقييم كل عبارة يكون منطقي logique : true أو false .
 ترجمة الخوارزمية إلى إجراء :

Procedure note_obs(var note : real);

Begin

Write ('أدخل نقطة الطالب');

Readln(note);

If (note >= 18) then

Writeln('ممتاز')

Else

If (note < 18) and (note >= 16) then

Writeln('جيد جدا')

Else

If (note < 16) and (note >= 14) then

Writeln('جيد')

Else

If (note < 14) and (note >= 12) then

Writeln('حسن')

```

Else
  If ( note < 12 ) and ( note >= 10 ) then
    Writeln('متوسط')
  Else
    Writeln (' غير مقبول ');
End; { fin note_obs }

```

4.الإختيار من مجموعة حالات.

4.Instruction de choix multiple:Case ..of..end

في المثال السابق كانت عندنا إختيارات من أكثر من 2 حالة و إستعملنا تعليمة if لحل الإشكال و لاحظنا كيف تداخلت التعليمات و هو أمر نستطيع إختصاره باستعمال تعليمة case لإختيار حالة ضمن مجموعة حالات أو إختيارات .

القاعدة: syntaxe:

```

Case expression      of
  Constant list : action ;
  Constant list : action ;
end ;

```

و المعنى semantics

قيم العبارة expression في سطر case إن كانت النتيجة تساوي أحد الثوابت الموجودة في قائمة الثوابت المكتوبة في الأسطر الموالية تنفذ التعليمة أو التعليمات الملزمة لذلك الثابت و هو ما نعبر عنه بـ action أي إفعل .

إذا كانت نتيجة التقييم لا توجد في القائمة يعلن عن خطأ إلا إذا إستعملت الإختيار Else و الذي تعين فيه ماذا عليك فعله في حالة عدم وجود ثابت يناسب تقييم العبارة .

يجب إختيار العبارة من نوع ترتيبي ordinal type : integer,char,boolean و أن تكون الثوابت المعينة من نفس النوع enumerated type .

الترتيبي المعين للعبارة.

أمثلة :

1. أعد كتابة الإجراء السابق بتعليمة case .

الخوارزمية l'algorithm:

إذا note =					
9..2	11..10	13..12	15..14	16..17	18..20
غير مقبول	متوسط	حسن	جيد	جيد جدا	ممتاز (writeln)

ملاحظة: هنا نوع من الثوابت مجال intervalle / range و هو مسموح به في بعض

الترجمات .

Procedure note (var note : integer) ;

Begin

Case note of

18..20 : writeln('ممتاز');

16,17 : writeln('جيد جدا');

14,15 : writeln('جيد');

12,13 : writeln('حسن');

10,11 : writeln('متوسط');

5..9 : writeln('غير مقبول');

else writeln('مستقبلك في المدرسة مهدد');

end;{ case}

end;{ note}

غيرنا نوع note من
real إلى integer لأن
real غير ترتيبى

2. أكتب إجراء يقوم بالعمليات الحسابية الأربع : + و - و * و / لعددين و يرجع النتيجة

للبرنامج الذي يناديه .

```

Procedure calcul ( var x,y,z : real ; var operateur : char );
Begin
  Case operateur of
    '+' : z := x + y ;
    '-' : z := x - y ;
    '*' : z := x * y ;
    '/' : z := x / y ;
    else writeln(' operateur inconnue ');
  end;{case}
end;{ calcul }

```

لإستعمال هذا الإجراء مثلاً نصحح بالمتغيرات الفعلية كما يلي :

```

Var a , b ,c : real ;
Op : char ;

```

و نناديه بعدما نقرأ قيم a,b و رمز العملية op كما يلي :

```

Write(' entrer 2 Nb: '); readln( a , b );
Write ( ' entrer le signe de l"operation +,-,*,/' );
Readln( op );
Calcul ( a , b , c , op );{ نداء الإجراء بالوسطاء الفعليين }

```

لكتابة النتيجة نكتب إجراء أو بتعليمة case مباشرة كالتالي :

```

Case op of
  '+' : writeln( a , '+',b,'=' , c );
  '-' : writeln( a , '-',b,'=' , c );
  '*' : writeln(a,'*' , b , '=' , c );
  '/' : writeln( a,'/' , b , '=' , c );
  else writeln(' operateur inconnu ');
end;

```

3. أكتب إجراء يتعرف على الحركات إن كانت صغيرة أو كبيرة majuscule أو

minuscule أو أرقام chiffre أو حركات خاصة caracteres spéciaux .

```

Procedure harakat ( var car : char ) ;
Begin
  Case car of
    'a'..'z' : writeln(' caractere alphanumérique minuscule');

```

```
'A'..'Z':writeln(' car. Alph. Majuscule ');
'0'..'9': writeln(' chiffre');
else writeln(' caractere special ');
end;{case}
end;{harakat}
```

5.التعليمات التكرارية .

5.Les instructions répétitives

التعليمات التكرارية ثلاث في باسكال :

تعليمية : أعد من ... إلىالفعل for ...todo

تعليمية مادام الشرط صحيحا إفعل whiledo.....

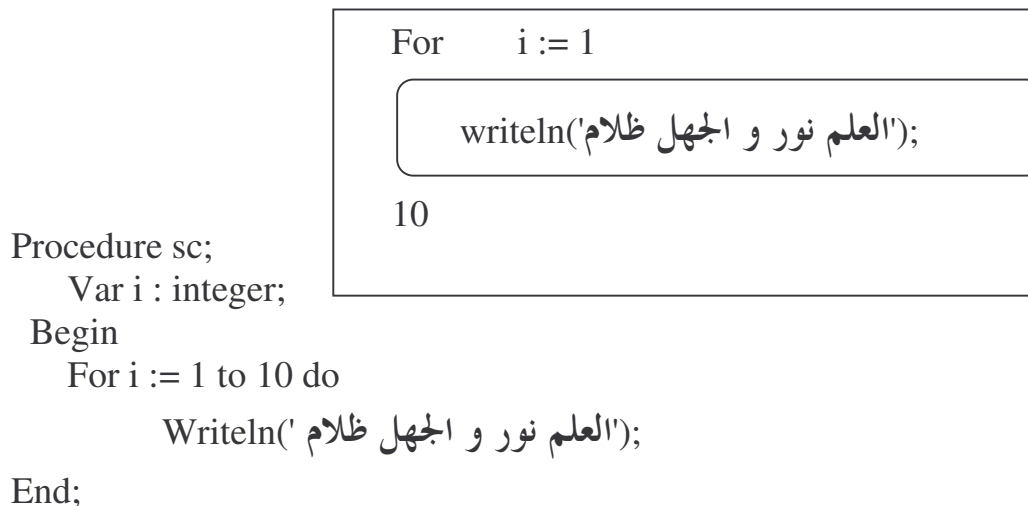
تعليمية : كرر إلى أن يصبح الشرط صحيحاuntil repeat

1.5 التعليمية : أعد من .. إلى ... الفعل

5.1 For ..todo

مثال 1.: أكتب 10 مرات الجملة: العلم نور و الجهل ظلام .

الخوارزمية بقالب G.N.S و الإجراء :



قالب G.N.S يسمح لنا تعيين قيمة العداد i الأولى و القيمة الأخيرة ليبدأ عد

التكرارات , و في داخل القالب الثاني تكتب التعليمات المراد تنفيذها. فيكون الإجراء نسخة

مترجمة للخوارزمية.

مثال 2.: أكتب إجراء يعرض على الشاشة رقم الحركة و الحركة et son caractère
numéro لجميع حركات ASCII .

الخوارزمية : l'algorithme

```
procedure ascii '
  var i : integer
begin
  for i := 0 to 255 do
    begin
      write ( i : 5 );
      writeln( chr (i):5 );
    end;{ for }
  end;{ ascii }
```

For i := 0

```
write( i );
writeln( chr ( i ) );
```

255

القاعدة النحوية : syntax

For variable := expression1 To أو downto expression2 DO
Instruction ;

أو

For variable := expression1 To أو downto expression2 DO
Begin
Instruction1;
.....
end;

المعنى semantic: تقييم العبارة الأولى و تضعها في المتغير العداد counter و تقييم العبارة الثانية لمعرفة القيمة النهائية للعداد إذا كان TO بينهم يكون العد تصاعديا أي يضيف العداد إلى نفسه واحد كل دورة بعدما ينفذ التعليمات أو التعليمات في القالب. أما إذا كانت DOWNTO بينهم فالعد تنازليا ينقص 1 من القيمة الأولى و يكرر حتى يصل إلى القيمة

النهائية .

****ملاحظات:** المتغير العداد خاص بتعليمة for لا يمكن تغيير قيمته داخل for فهو الذي يراقب التكرارات . و قيمته مهمة في نهاية التكرار إذا أردت إستعمالها فعليك تعيين قيمة جديدة لها .

إذا إستعملت for داخل إجراء أو دالة يجب أن يكون المتغير العداد بسيط محلي simple local variable بالإجراء لا و بسيط شكلي مقدم بالقيمة أو العنوان .
نوع المتغير العداد control variable يجب أن يكون من نوع ترتيبى: integer, char, boolean, enumere, إختياري .

قيمة العبارة الأولى في To يجب أن تكون أكبر من قيمة العبارة الثانية. و العكس في downto .

مثال 3. : لإستعمال downto :

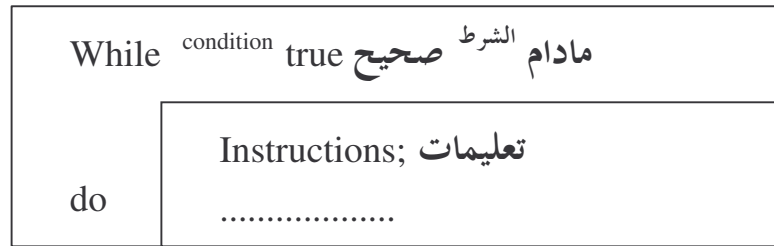
```
For car := 'Z' downto 'A' do
  Begin
    Write ( car );
    Writeln( ord ( car ) );
  End;
```

2.5 تعليمة ما دام الشرط صحيحا أفعل ...

5.2 While (condition: true) do

هذه التعليمة تكرارية شرطية : تعني قيم الشرط: إذا كانت النتيجة المنطقية صحيحة true نفذ التعليمات بعد do و كرر العملية بعد تقييم الشرط إذا ما زال صحيحا و إلا ينتهي التكرار .

فهي تعليمة تكرارية تنفذ من صفر مرة إلى مرات حسب الشرط أي لا تنفذ أي دورة أو تكرر إلى أن يصبح الشرط خطأ false .
تمثل بقالب G.N.S كما يلي :



مثال 1. : ليكن لدينا البرنامج التالي :

```

program exp_while ;
  uses wincrt;
  Var A : integer ;
Begin
  A := 100 ;
  While      A > 0 do
    Begin
      Write ( A:5);      A := A - 10 ;
    End ;
  Writeln ;
  While A > 0      do    Write (' rien ne s"ecrit ');
  Writeln(' la 2° boucle n"a pas été exécuté ');
End.

```

تنفيذ البرنامج يعرض ما يلي :

100 90 80 70 60 50 40 30 20 10

la 2° boucle n'a pas été exécuté

لاحظ أن الـ while الأول نفذ وكرر التنفيذ عدد معين حتى أصبح الشرط خطأ أي

قيمة A أصغر من 0 فتوقف : الشرط false. أما الـ while الثاني فلم ينفذ و لا مرة لأن قيمة

A أصبحت -10 بعد التكرارات الأولى

و الشرط إذا خطأ false فلا تكرر إذا .

القاعدة النحوية : syntax

While boolean expresion Do
Action

و تعني semantics : قيم العبارة المنطقية: إذا كانت النتيجة صحيحة true إفعال do

the action أي نفذ التعليمات ثم قيم الشرط مرة أخرى إذا كانت النتيجة صحيحة true

نفذ مرة أخرى و هكذا ما دام الشرط صحيحا كرر التنفيذ . إلى أن يصبح خطأ قف
الـ : while و واصل تنفيذ التعليمات بعده .

*ملاحظات : عدد التكرارات غير ثابت مثل for .

العبارة المنطقية هي التي تتحكم في عدد التكرارات وهي من : صفر إلى عدد معين
حسب دوران الشرط . هنا الشرط يشبه شرط تعليمة If و نفس القواعد تنطبق عليه .

مثال 2. : طرح سؤال على المستعمل قبل بدأ تنفيذ البرنامج : هل تريد بدأ

التنفيذ إن أجاب بنعم يبدأ و إلا ينهي العمل ثم يطرح سؤال ثاني هل تريد التنفيذ مرة
أخرى...

Program question ;

Uses wincrt;

Var reponse ;

.....

begin

write (' voulez - vous commencer ? '); readln (reponse) ;

while (reponse = 'o') or (reponse = 'O') do

begin

.....;

write (' v-v continuer '); readln (reponse);

end;{ while }

.....;

end.

مثال 3. : إجراء يحسب معدل بيانات تقرأ من الدخول و آخرها 0 أو أقل .

Procedure moyenne ;

Var data ,somme, moy : real ;

I : integer ;

Begin

Write (' give the data U want to average with last as
0 or negative one ');

Read (data) ; I := 0 ; somme := 0 ;

While data >= 0 do

Begin

I := I + 1 ;

```

Somme := somme + data ; Read ( note ) ;
End; { fin du while }
Moy := somme / I ;      Writeln( moy ) ;
End; { moyenne }

```

مثال 4.: إنهاء القراءة من الدخول إلا إذا ضرب المستعمل نهاية الجذاذة fin de fichier
input (ctrl +Z).

```

writeln(' entrer les données et finir par Ctrl+Z ');
read ( data );
I := 0 ; somme := 0 ;
While Not eof do
  Begin
    ..... read ( data );
  end;

```

مثال 5. قراءة بيانات إلى نهاية السطر: EOLN: الممثلة بالـ: ←

```

While not eoln do
  Begin
    Read ( data ); .....
  end;

```

3.5: تعليمة أعد ... إلى أن يصبح الشرط صحيح .

5.3 Repeatuntil (condition: true)

هذه التعليمة تكرارية شرطية كذلك و لكنها تختبر الشرط في نهاية القالب أي تبدأ بتنفيذ ما بداخلها ثم تختبر الشرط لتعيد:

فإذا كانت نتيجة تقييمه صحيحة true لا تعيد و إذا كانت النتيجة خطأ false تكرر . فهي تكرر على الأقل مرة واحدة .

القاعدة syntax : Repeat

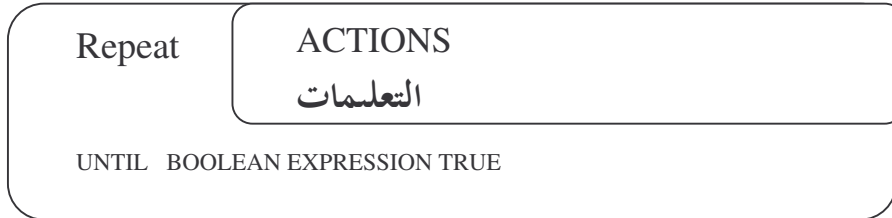
```

action(s); .....
until boolean expression ;

```

المعنى: semantics: نفذ التعليمات actions بين repeat و until : إذا كانت العبارة

المنطقية False عندما تصل إلى until كرر التنفيذ و هكذا حتى تصبح العبارة المنطقية true صحيح . و برسوم G.N.S كالتالي:



مثال 1: أكتب برنامجا يحول لوحة المفاتيح الرقمية إلى حاسب جيبي calculatrice de poche تقوم بالعمليات الحسابية الأربعة : الجمع , الطرح , الضرب , القسمة. ثم يبقى تحت التصرف المستعمل إن أراد تكرار التنفيذ يجب ب(ن)عم و إلا ينتهي التنفيذ .
الخوارزمية :

R E P E A T	Write(entrer 2 Nb:); readln(x, y); write(entrer l'operateur '); readln(op);			
	case op of			
	+	-	*	/
	z:=x+y	z := x-y	z:= x*y	if y = 0 then writeln('div / zero') else z := x/y
	writeln(z) ; write (' v-v une autre execution '); readln(op);			
Until rep <> 'o'				

و يترجم إلى برنامج أو إجراء كما يلي :

```

Program calculate ;
Uses WinCrt;
Var x, y , z : real ;
    Op , rep : char ;
Begin
    Writeln(' prog. Calculatrice : + ; - ; * ; / ' ) ;
    REPEAT
        Write ( ' donner 2 Nb. : ' ) ; Readln ( x , y ) ;
        Write ( ' entrer l'operateur ' ) ; Readln ( op ) ;
        Case op of
            '+' : z := x + y ;
            '-' : z := x - y ;
            '*' : z := x * y ;
            '/' : if y = 0 then
                        writeln ( ' divion par Zero ' )
                    else
                        z:= x /y ;
                    else ( ' operateur inconnu ' ) ;
            end ; { case }
        writeln ( ' le resultat est ' , z ) ;
        write ( ' هل تريد التنفيذ مرة أخرى '); Readln ( rep ) ;
    UNTIL rep <> ' o' ;
    Writeln ( ' مع السلامة ' );
End . { calculette }

```

*ملاحظة : بالرغم من أن القاعدة قلمي علينا كلما وجدت مجموعة تعليمات يجب كتابتها داخل قالب begin و end إلا في تعليمة repeat ... until التي تلعب هذا الدور كذلك في إحتواء التعليمات بداخلها دون اللجوء إلى وضعها بين begin..... end .

6.تمارين حول الباب :

6.Exercices

1.أكتب دالة تقرأ سطرا حركة بحركة و ترجع عدد الحركات فيه.

2. أكتب إجراء يقرأ 20 نقطة و يحسب عدد النقاط فوق المعدل و التي أقل منه . مع شرح لإدخال البيانات كالتالي : أدخل النقطة رقم : ...:
3. ما هي نتيجة تنفيذ هذا البرنامج :

```

Program loop ;
  Uses wincrt;
  Var halaka : integer ;
  Begin
    Halaka := 1 ; write ('follow up to :');
    While halaka < 20 do
      Begin
        Write ( halaka ); halaka := 3*halaka;
      End;
    Writeln;
    While halaka >= 1 do
      Begin
        Write (halaka);
        halaka := halaka -10;
      End;
    Writeln( halaka - 1 , ' this is the end ' );
  End.

```

4. ما هي قيمة المتغير x بعد تنفيذ كل من الأجزاء التالية :

```

X :=1;
While x < 1 do x := x +1 ;
X := 1 ,
Repeat
  X := x + 1 ;
Until not ( x < 10 );
x := 10 ;
while x < 10 do x := x + 1;
x := 10 ;
repeat
  x := x +1 ;

until not ( x < 10 );

```

5. أكتب برنامجا يجد جميع الحلول للمعادلة التالية :

ترجع التابع لـ x le successeur $\text{succ}(x)$

ترجع السابق لـ x $\text{pred}(x)$

ترجع الحركة التي رتبها x $\text{chr}(x)$

2. النوع الحقيقي .

2.Type réel / real type

وهي مجموعة الأعداد الحقيقة المصرح بها في مكتبة باسكال :

$$\text{Real} = 1.9e^{-39} .. 1.7e^{38}$$

و يضيف BPW7 ما يلي :

$$\text{Single} = 1.5e^{-45} .. 3.4e^{38}$$

$$\text{Double} = 5.0e^{-324} .. 1.7e^{308}$$

$$\text{Extended} = 3.4e^{-4932} .. 1.1e^{4932}$$

$$\text{Comp} = -9.2e^{18} .. +9.2e^{18}$$

الدوال succ و pred ليست لها معنى في النوع الحقيقي . كما لا يجوز استعمال المتغير

الحقيقي كعداد في تعليمة تكرارية أو غيرها .

الدوال المستعملة بعامل حقيقي :

تقريب قيمة x : $\text{round}(x)$: arrondir x

مثال : $\text{round}(1.6) \rightarrow 2$

$\text{round}(-1.6) \rightarrow -2$

دالة بتر x إلى الجزء الصحيح منه : $\text{trunc}(x:\text{real}):\text{integer}$

وتسمى في الرياضيات بالدالة ذات القسمة الصحيحة : la fonction a partie entière .

$$\text{If } x > 0 \quad \text{round}(x) \iff \text{trunc}(x + 0.5)$$

$$\text{If } x < 0 \quad \text{round}(x) \iff \text{trunc}(x - 0.5)$$

3. نوع الحركات.

3.Char and String type

و هو مجموعة الحركات الموفرة في باسكال القياسي . و غالبا ما تكون حركات ASCII و جميع حركات UNICODE الموفرة في نظام Windows .

يضيف BP7 النوع String سلسلة الحركات من طول 1 إلى 255 حركة.

A string type variable is a sequence of characters with a dynamic length, and a constant maximum size between 1 and 255 .

Syntax : string [constant] OR string

مثال :

```
const    LineLen = 79 ;
type
Name = string [25];
Line = string [LineLen] ;
```

الثابت يكتب بين هلالين : const a = ' H' ;

Bonjour = 'السلام عليكم' ;

توجد علاقة ترتيبية في المجموعة : الدوال ord و chr تسمح لنا بإيجاد تقابل بين مجموعة الحركات و النوع الصحيح .

عاملوا العلاقات يقبلون على نوع الحركات : إن كان c1 و c2 معاملان من نوع

الحركات و R عامل من عاملوا العلاقات فإن العلاقة :

$C1 \quad R \quad c2 \iff ord(c1) \leq ord(c2)$

الدوال : pred , succ :

$pred(c) = chr(ord(c) - 1)$

ترجع الحركة السابقة لـ c .

$Succ(c) = chr(ord(c) + 1)$

ترجع الحركة التي تأتي بعد c .

4. النوع المنطقي .

4.Boolean type

النوع المنطقي نوع قياسي يحتوي على ثابتين : `false` , `True` .

في BP7 نجد 4 أنواع منطقية :

There are four predefined boolean types: `Boolean`, `WordBool`, `LongBool`, and `ByteBool` .

تعرف جميعا كما يلي:

type

`Boolean = (False, True);`

`WordBool = (False, True);`

`LongBool = (False, True);`

`ByteBool = (False, True);`

فهي الأربع متكونة من ثابتين فقط و لكن تختلف بطول الكلمة الثنائية `Mot binaire`

التي تمثل كل ثابت في الذاكرة المركزية و هي 8 و 16 و 32 bit

. `ByteBool`, `WordBool`, and `LongBool` exist primarily to provide

compatibility with Windows .

- `Boolean` is Byte-sized (8 bits)
- `WordBool` is Word-sized (16 bits)
- `LongBool` is Longint-sized (32 bits)
- `ByteBool` is Byte-sized (8 bits)

بما أن النوع المنطقي ترتيبى نجد العلاقات التالية :

`False < True`

`Ord (False) = 0`

`Ord (True) = 1`

`Succ (False) = True`

`Pred (True) = False`

For Windows compatibility, booleans can assume ordinal values other than

0 and 1 .

يمكن للمتغير المنطقي أخذ قيم ترتيبية غير 0 و 1 من أجل التطابق مع نظام

. Windows

While true do writeln(' pascal is good '); أمثلة :

التعيين إلى متغير منطقي والعبارات المنطقية :

Var a , b ,c : boolean ;

```
.....
a := 5 > 3 ; b := a or ( 6 > 8 ) ;.....
c:= a AND b;.....
while not ( a > b ) do .....
If ( a < b ) and ( a > c ) then .....
if ( a = b ) OR ( c <> b ) Then .....
```

. les fonctions booléennes : الدوال المنطقية

الدالة EOLN: ترجع TRUE عندما تصل إلى إشارة نهاية السطر

chr (13):

مثال:

while Not Eoln do read (I) ;

الدالة EOF: تكون TRUE إذا وصلت إلى نهاية الجذاذة : fin de fichier وهي (:

.Ctrl + Z)

مثال :

while Not Eof do read(nom);

الدالة ODD(x) ترجع true إذا x وترية impaire .

قراءة و كتابة المتغير المنطقي : لا يمكن قراءة المتغير المنطقي.

أما ; write (x) تكتب القيمة x : false أو true .

5.النوع المصرح به إختياريا أو التعدادي .

5.Type scalaire déclaré ou énuméré

هذا النوع قياسي ترتيبي يصرح به في قسم التصريح بالنوع بإسم له ثم بين قوسين تسمى العناصر الواحد تلو الآخر بترتيب يفصل بينها بفاصلة و تعد ثوابت constantes

عند تعيينها للمتغير , الأول له الرتبة صفر (0) ثم 1 إلى آخره.

Enumerated types define ordered sets of values by enumerating the identifiers that denote these values. Their ordering follows the sequence in which the identifiers are enumerated .

Type name = (identifier ,identifier ,... ,identifier);

The first constant has an ordinality of 0, the second has an ordinality of 1, the third an ordinality of 2, and so on .

```

type taille = ( grand , moyen , petit );
mois = ( janvier , fevrier , mars , avril );
Suit = (Club, Diamond, Heart, Spade) ;
var mesure : taille ;
    travail : mois ;
    z : integer ;

```

الدوال المستعملة مع النوع المصريح به : succ ; pred ; ord :

```

      measure := grand ;
      z := ord ( measure ) ;

```

ترجع ord رتبة grand و هي (صفر) : 0 .

$$\text{Ord}(\text{Club}) = 0$$
$$\text{Ord}(\text{Diamond}) = 1$$

The `Ord` standard function returns the ordinality of an enumerated constant.

Travail := pred (mars) ; { avril وهو }

يمكن إستعمال عامل أو المقارنة : opérateur logique

(< ; > ; = ; <= ; >= ; <>) للمقارنة بين ثابتين أو متغيرين من نوع مصرح به .

* كتابة الثابت أو المتغير من هذا النوع غير ممكنة مباشرة و لكن مروراً بطريقة الاختيار

المتعدد case نتمكن من كتابتها كسلسلة حركات chaîne de caractères كما يمكن قراءة

المتغير ولكن بطريقة غير مباشرة على شكل عدد صحيح يمثل رتبة الثابت في النوع .

مثال :

```

Type jours = ( samedi, dimanche , lundi , mardi
               ,mercredi, jeudi , vendredi ) ;
Var j : jours ;
Begin
  Writeln(' entrer un nombre de 0 a 7 ');
  Readln ( j ) ;
  For j := samedi to vendredi do
    Case j of
      Samedi : writeln(' samedi ');
      Dimanche : writeln(' dimanche ');
      .....;
      vendredi : writeln ( ' vendredi ');
    end;{ case}
end.

```

6.نوع المجال .

6.Type intervalle /subrange type

هذا النوع يصرح به من طرف المبرمج و يأخذ مجاله من أي نوع بسيط كان إلا النوع الحقيقي real و يسمى النوع المضيف .

فالمجال زمرة من مجموعة , يعرف بتعيين قيمة كبرى و أخرى صغرى .

الأولى يجب أن تكون الأصغر متبوعة بنقطتين ثم القيمة الأكبر .

A sub range type is a range of values from an ordinal type called the host type .

Syntax

Constant1 .. Constant2

The \$R compiler directive controls range checking of sub range types .

مثلا :

```

Type index = 0..100;
Lettre = 'a'..'z';

```

كل العمليات الممكنة على النوع المضيف ممكنة في المجال التابع له .

الباب السابع : بنية القائمة أو المصفوفة من بعد واحد.

Chapitre 7: Structure de liste ou tableau a une dimension

المصطلحات : vocabulary:

القائمة أو المصفوفة من بعد واحد : liste ou matrice a une dimension / array

الفهرس أو الدليل : indice / index

المتغير المذيل : variable indicée

1. الهدف :

1.But

في بنية البيانات البسيطة structure de donnée simple عندما نصرح بمتغير تخضر له ذاكرة واحدة و لكن كثيرا من الأحيان نحتاج إلى أكثر من ذاكرة لشيء واحد .مثلا نريد تسجيل مقاييس الحرارة و سقوط الأمطار على مستوى الوطن من جميع مراكز الرصد الجوي . فهذه المسألة تطلب منا تحضير مجموعة ذاكرات لمتغير واحد : température تخضر لها مثلا 100 و pluies تخضر لها 100 كذلك.

من أجل ذلك نحتاج إلى بنية أخرى غير البسيطة و هي بنية القائمة .

2.التعريف :

2.définition

القائمة بنية للبيانات تكوّن من مجموعة من العناصر éléments من نفس النوع . أهم خصائص القائمة قابلية النفاذ المباشر لأي عنصر . كما تعالج تتابعيا .

3.التصريح بالقائمة :

3.Déclaration de liste / array

يصرح بالقائمة في قسم التصريح بالنوع أو مباشرة كمتغير :

Array [indexType] of TypeListe ;

نوع الفهرس index type / type d'index هو تحديد طريقة النفاذ إلى العناصر , يكون مجال interval / subrange أو نوع مصرح به.

نوع البيان TypeListe يكون من أي نوع قياسي أو مصرح به .
لا يجوز أن يكون نوع الفهرس كل المجموعة الصحيحة أو الحقيقية .
أمثلة :

Type boolean = Array [boolean] of boolean ;

Caractere = Array [Char] of Char ;

Liste = [1..100] of 1..100;

Age = 1..70 ;

Personne = Array [age] of integer ;

Homme = Array [age] of age ;

Semaine = (sam,dim, lun, mar,mer ,jeu, ven) ;

Mois = Array [semaine] of integer ;

Logique = Array [false..True] of Boolean ;

Annee = Array [semaine] of semaine ;

كما يجوز التصريح بالقائمة في قسم التصريح بالمتغير :

Var jours : Array [semaine] of integer ;

4.العمليات الشاملة على القائمة.

Opérations globales sur les listes .

1.4 التعيين .

4.1 L'affectation

يمكن تعيين قائمة إلى أخرى إذا كانا من نفس النوع .

مثال : var L,M : Array [1..100] of real ;

begin L := M ;

2.4 المقارنة .

4.2 L'évaluation logique

لا يمكن مقارنة المتغيرات من نوع قائمة إلا بالتساوي أو بالاختلاف

if L = M then مثال .

If L <> M then

أما المقارنة : أصغر : < , أكبر : > , <= , >= فهي غير ممكنة .

5. المتغير أو العنصر المذيل .

5.Variable indicé

كل عنصر من القائمة إذا أُشير إليه بعنوانه في المجموعة نسميه عنصر مذيل لأننا نشير إليه

برقمه في القائمة مثل : L[10] ; M[i] أو H[i+5] .

القاعدة إذاً هي : تعريف القائمة يتبع بعنوان العنصر بين عارضتين entre crochet

للإشارة إلى أي موضع في القائمة و يسمى المتغير المذيل variable indice .

6. التصريح بالوسيط الشكلي من نوع قائمة في إجراء أو

دالة

6.Parametre de type Array déclaré formel dans une procédure ou fonction.

التصريح التالي غير مسموح به في باسكال القياسي standard :

Procedure lit (var L : array [1..10] of integer);

لأنه يشير إلى نوع جديد في الإجراء وذلك يقتضي حسب قواعد باسكال التصريح

بالنوع في قسم التصريحات type مثل :

Type liste = Array [1..10] of integer ;

Var L,M : liste ;

Procedure lit (var N : liste) ;

7.قراءة القائمة.

7.Lécture d'une table/array

لقراءة القائمة يجب إدخال القيم عنصراً بعنصر .

مثال :


```

Program lecture ;
  Uses winCRT;
  Const Max = 100 ;
  Type vecteur = Array [ 1..Max] of integer ;
  Var w : vecteur ;
  Procedure lit ( var V : vecteur ) ;
    Var i : integer ;
    Begin
      Writeln(' entrer les éléments de vecteur :');
      For i := 1 to Max do
        Read ( v [i] ) ;
      End ; {lit}
  End ;
Begin
  Lit ( w ) ;....

```

لا يمكن قراءة القائمة دفعة واحدة كـ: Read (w) ;
8. كتابة القائمة.

8.Ecriture d'une liste

مثلما فعلنا في القراءة تكون الكتابة عنصرا بعنصر .

```

Procedure ecrit ( var H : vecteur ) ;
  Var j : 1..max ;
  Begin
    For j := 1 to Max do
      Write ( H [j] ) ;
    End ;

```

9. التعيين بين و إلى عناصر القائمة .

9.Afféctation entre élément d'une liste .

التعيين بين عناصر القائمة الواحدة مثل : $w[i] := w[i + 1]$;

من متغير إلى عنصر : $v[j] := K$;

من عنصر إلى آخر من قائمة ثانية: $V[i] := W[j]$;

القاعدة : يجب احترام ملائمة النوع بين قائمتين أو بين نوع المتغير و نوع القائمة.

مثال : إجراء تبديل عنصرين متتاليين في قائمة.

```

Procedure swap ( var x : liste; i : integer ) ;
  Var temp : integer ;
  Begin
    Temp := x [i] ; X[i] := x[i+1] ;
    X[i+1] := temp ;
  End;

```

لتبادل عنصرين ما ننادي بالإجراء بالوسيط الفعلي القائمة و عنوان العنصر الأول مثلا :

swap (w , j) ;

10.النفاز إلى عنصر من القائمة.

10 Accès a un élément de la liste .

النفاز إلى عنصر ما يعني البحث عن العنصر بمعرفة عنوانه مباشرة أو بمعرفة القيمة

المخزنة فيه .

بمعرفة رقمه , مثلا :

```

procedure get ( var v : vecteur );
  Var i : integer ;
  Begin
    Write(' donner le N° de l'element : ');Readln ( i ) ;
    Writeln(' l'element ',i,' est :', v[i] ) ;
  End ;

```

و هو نفاز مباشر لأنه في الذاكرة المركزية و بمعرفة عنوان العنصر .

أما البحث عن العنصر بمعرفة قيمته فهي عملية بحث عنه في كل القائمة مثل:

```

Procedure cherche ( var g : vecteur ) ;
  Var i , S : integer ;
  Begin
    Write (' donner la valeur de l'element a chercher:');
    Readln ( S ) ;
    For i := 1 to Max do
      If g [i] = S then
        Writeln('Trouvé',S,'est d'indice : ', i );
  End ;

```

11.تمارين :

11.Exercices

1. أكتب إجراء قراءة قائمة المبيعات من وسائل النقل مثل : camion , bicyclette ,

Bus , velomoteur , voiture .

ثم إجراء يكتب محتوى القائمة .

* الحل :

```
Type Moyen_transp = ( bicyclette ,
moto,camion,voiture,bus);
    dataMT : Array [ moyen_transp] of integer ;
    Var MT : Moyen_transp ;
    Data : dataMT ;
    Procedure      lit ( var data : dataMT ) ;
        Var i :      Moyen_transp ;
        Begin
            Writeln(' ceci est un exemple de liste avec indice   de type
de donnée déclare (énumère ) ');
            Writeln( ' entrer le nombre vendu dans l'ordre  suivant :
bicyclette ,moto,camion,voiture,bus ');
            For i := bicyclette To Bus Do
                Readln( data[i] );

        End;
    Procedure      ecrit ( var data : DataMT );
        Var i : Moyen_transp ;
        Begin
            Writeln(' les ventes sont comme suit :');
            For i := bicyclette to bus do
                Writeln( data [i] );

        End;
```

2.أكتب إجراء تقسيم قائمة أعداد إلى قائمتين : الأولى للأعداد الموجبة و الثانية للأعداد

السالبة .

* الحل :

```
Type vecteur = Array[ 1..10] of real ;
    Var w , pos, neg : vecteur ;
```

```

Procedure divide ( var V ,pos,neg : vecteur );
  Var i , j , k : integer ;
  Begin
    J := 1 ; k := 1 ;
    For i := 1 to 10 do
      If v[i] > 0 then
        Begin  Pos [j]:= v[i];  j:=j+1; End
      Else  Begin  Neg[k]:=v[i];K:=k+1;  End;
    End;
  End;

```

3. أكتب إجراء البحث عن أصغر عنصر في قائمة ما .

**الحل :

```

Type liste = Array [ 1..10] of integer ;
  Var LP : liste ;
  PPT : integer ;
  Procedure Ppliste ( var L : liste; var pp : integer );
    Var i: integer ;
    Begin
      PP := L[1];
      For i := 2 to 10 do
        If L[i] < pp then
          PP := L[i] ;
      End ;
    End ;

```

عند نداء الإجراء نقدم له عنوان القائمة أي الوسيط الفعلي LP الذي تكون قرأت و ترجع

قيمة الأصغر في ppt Ppliste (LP , PPT);:

4. أكتب إجراء البحث عن أكبر عنصر في قائمة من حركات. هنا نستعمل الدالة (c) ord

لمعرفة أكبر حركة في جدول ASCII .

**الحل :

```

Type chaine = Array [ 1..256] of char ;
  Var ch : chaine ;
  C : char ;
  Procedure Pgrand ( var ligne : chaine ; var carPg : char );
    Var i : integer ;
    Begin

```

```

CarPg := ligne[1];
For i := 2 to 256 do
    If ord ( ligne[i] ) > ord ( carPg ) then
        CarPg := Ligne[i] ;
End;

```

نقرأ القائمة ثم ننادي بالإجراء: $Pgrand(ch, c)$ و c ترجع الحركة الأكبر .

الباب الثامن : خوارزميات الترتيب .

Chapitre 8: Algorithmes de tri

Sort algorithms

1. الترتيب بطريقة التبادل.

1.Sorting by straight exchange or bubble sort

tri a bulle.

الهدف من هذا الترتيب الوصول إلى ترتيب عناصر القائمة تصاعديا بالشكل

التالي :

نقوم بمقارنة العناصر مثنى مثنى بهدف وضع الأصغر في الأول أي أننا سنقوم بتبادل العناصر لكي يصعد أصغر عنصر إلى الأعلى .

وهكذا نمر على القائمة $(n-1)$ مرة .

مثال : نأخذ قائمة مرتبة تنازليا لترتيبها تصاعديا.

الخطوة الأولى :

نقارن مثنى مثنى الأعداد بداية من آخر عنصر:

إلى آخر عنصر $n-1$ و نبدل العنصرين المتتاليين .

الخطوة الثانية :

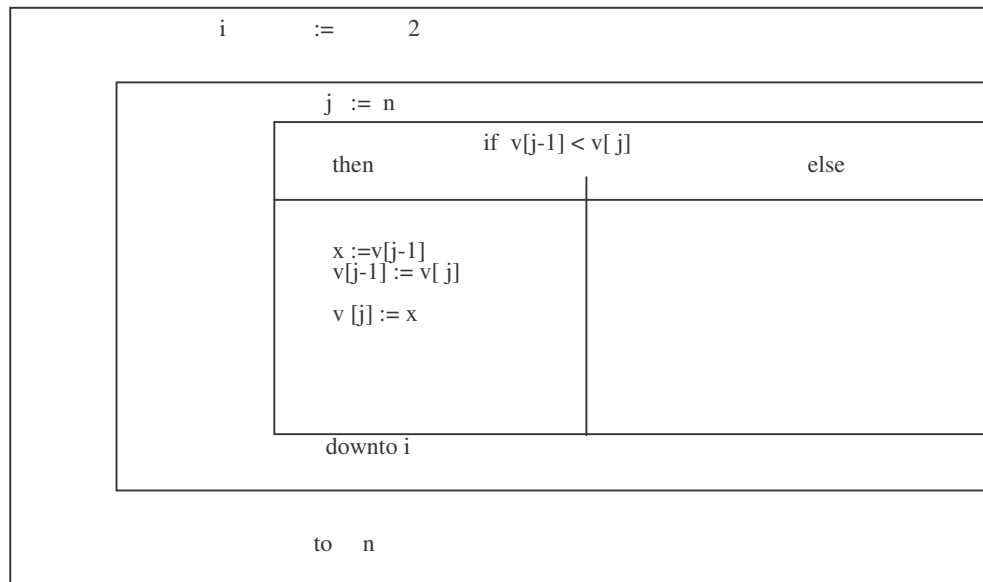
نكرر الخطوة الأولى $n-1$ مرة أي 9 مرات .

5	55	55	55	55	55	55	55	55	55	1
55	5	44	44	44	44	44	44	44	44	2
44	44	5	33	33	33	33	33	33	33	3
33	33	33	5	22	22	22	22	22	22	4
22	22	22	22	5	10	10	10	10	10	5
10	10	10	10	10	5	9	9	9	9	6
9	9	9	9	9	9	5	8	8	8	7
8	8	8	8	8	8	8	5	7	7	8
7	7	7	7	7	7	7	7	5	6	9
6	6	6	6	6	6	6	6	6	5	10

الخطوة الثانية : نكرر الخطوة الأولى n-1 مرة أي 9 مرات .

6	7	8	9	10	22	33	44	55	5	1
7	8	9	10	22	33	44	55	6	5	2
8	9	10	22	33	44	55	7	6	5	3
9	10	22	33	44	55	8	7	6	5	4
10	22	33	44	55	9	8	7	6	5	5
22	33	44	55	10	9	8	7	6	5	6
33	44	55	22	10	9	8	7	6	5	7
44	55	33	22	10	9	8	7	6	5	8
55	44	33	22	10	9	8	7	6	5	9

و هكذا تصبح القائمة مرتبة بعد هذه التكرارات .
و تكون الخوارزمية كالتالي :



في هذا المثال يتبين أن عدد المرات التي نمر بها على القائمة هي

$n-1$. و أن أصغر عنصر يصعد إلى الأعلى كما تصعد كويرات الصابون في الماء bulle de savon

لذلك سميت هذه الطريقة بـ bubble sort / tri a bulle و يكون الإجراء كما يلي :

```

const N = 10 ;
Type vecteur = Array [ 1..N ] of integer ;
var v : vecteur ;
procedure bubble_sort ( var v : vecteur ) ;
  var i , j , x : integer ;
  begin
    for i := 2 to N do
      for j := N downto i do
        if v [ j-1 ] > v [ j ] then
          begin
            x := v[j-1] ; v[j-1] := v[j] ; v[j] := x ;
          end;
        end;
      end;
    end;
  end;

```

مناقشة الترتيب :

أول دورة أخرجت أصغر عنصر فوضعت في بداية القائمة بعمليات التبادل , ثاني دورة : صعد الأصغر الثاني الموالي للأول . و هكذا نلاحظ أن عدد الدورات $n-1$. أما عدد المقارنات مثنى مثنى فيساوي : $n(n-1)/2$. و هو وقت طويل خاصة إذا كانت القائمة من 1000 أو أكثر . لذلك يجب إدخال تحسينات على خوارزمية هذا الترتيب ليكون أسرع .

التحسين الأول : ندخل متغير منطقي نضع فيه true إذا وقع تبادل و إلا فهي false حيث تكون القائمة رتب . و يصبح الإجراء كالتالي :

```

procedure Bubble2 ( var V : vecteur ) ;
  var echange : boolean ;
      I , M , X : integer ;
  begin
    echange := True ;      M := N ;
    While ( M >= 2 ) and ( Echange = True ) do
      begin
        Echange := False ;
        For I := 1 To ( M - 1 ) do
          If v [I] > v[ I+1] Then
            begin
              X := V[i] ;
              V[i] := V[i+1];
              V[i+1] := V[i];
              Echange := True ;
            end;
          M := M - 1 ;
        end;{ while }
      end;{ bubble2}

```

يتوقف تكرار الـ while إذا لم يتم أي تبادل خلال آخر دورة و هي علامة أن القائمة مرتبة . كما نستطيع إدخال تحسين آخر في التكرار For بتحديد عدد التكرارات إلى ما هو مفيد و لا داعية للمرور على جميع العناصر و ذلك بتذكر دليل indice العنصر الذي تم تبديله حتى لا نتجاوزه . و هو التحسين الثاني :


```

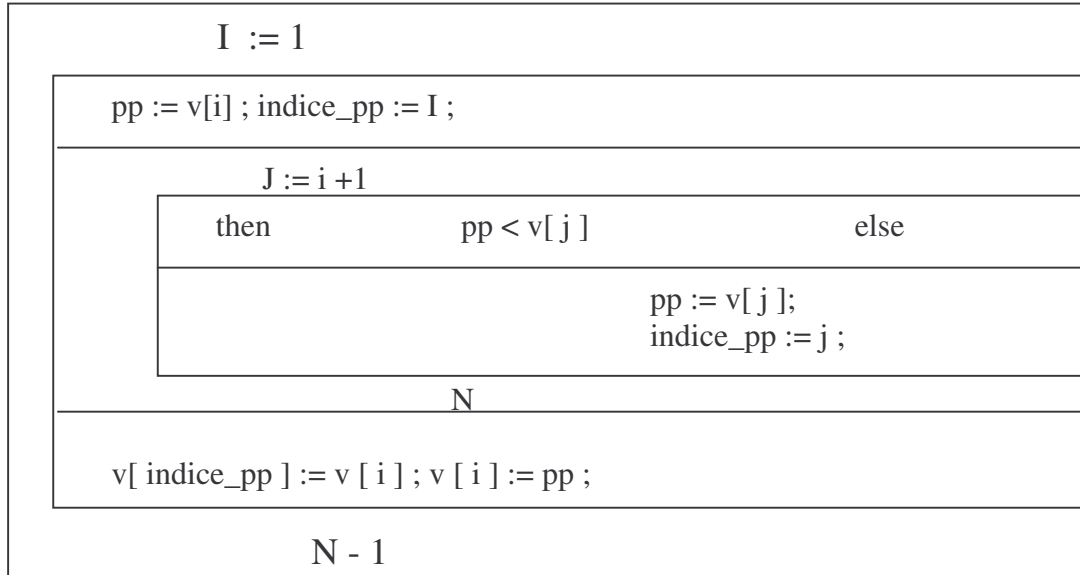
procedure Bubble3 ( var V : vecteur );
  Var Echange : Boolean ;
      I , M , X , W : Integer ;
begin
  Echange := True ; M := N ;
  While ( M >= 2 ) and ( echange = True ) do
    begin
      Echange := False ;
      For I := 1 To ( M - 1 ) Do
        If V(I) > V ( I + 1 ) Then
          Begin x := V [i]; V[i] :=V[i+1]; V[i+1] := X ;Echange := True ;
            {indice de l'element echange }W := I ;{ دليل العنصر } end;
        M := W ;{ نحدد الدورة إلى آخر عنصر مبدل } end ; { while}
    end;

```

2.الترتيب بالإنّخاب .

2.Tri par sélection

هذا النوع من الترتيب يعتمد على طريقة مباشرة للبحث عن أصغر عنصر ليبدل مع أول عنصر في القائمة ثم تتبع نفس الطريقة بداية من العنصر الثاني و هكذا تكرر العملية بداية من العنصر الثالث ثم الرابع إلى أن يبقى عنصرين فقط . و تكون الخوارزمية كما يلي :



```

Procedure Tri_selection ( var V : vecteur ) ;
  var i , j , indice_pp , pp : integer ;
  begin
    For i := 1 To ( N - 1 ) Do
      begin
        PP := V [ i ] ;
        indice_PP := i ;
        for j := i + 1 To N do
          If PP < V [ j ] Then
            (* البحث عن أصغر و الاحتفاظ بدليله . *)
            begin
              PP := V[ j ] ; indice_PP := j ;
            end ; { Then }
          V[indice_PP ] := V [ i ] ;
          V [ i ] := PP ;
        end;{ 1° For }
      end; { Tri_selection}

```

3. الترتيب بالدمج .

3.Tri par insertion

مبدأ هذا الترتيب نوضحه بالمثال التالي :

إذا كان لدينا قائمة $V[1..5]$ مرتبة كيف ندمج عنصر سادس $V[6]$ في مكانه بالقائمة :
 بالمقارنة و الإزاحة المتكررة . بعدما يدمج $V[6]$ نتابع نفس الأسلوب مع $V[7]$ إلى آخر
 عنصر في القائمة .

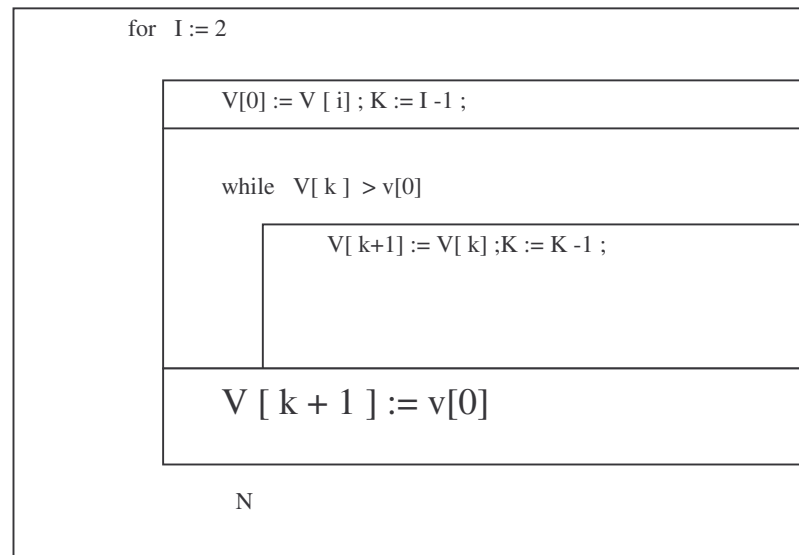
aux	9					
7	6	5	4	3	2	1
...	9	30	20	10	7	5
...	30	20	10	9	7	5

قمنا بوضع العنصر السادس في ذاكرة إضافية aux لنفرغها و نزيح العناصر 5 في 6 و 4 في 5 و 3 في 4 و ندمج القيمة من aux في المكان 3 . في البداية نقتصر على قائمة من عنصر واحد $v[1]$ ثم نقارن و نزيح العنصر الثاني حتى يدمج في مكانه فتصبح القائمة

التحتية sous - vecteur من عنصرين مرتبين نكرر العملية مرات حتى تصبح القائمة مرتبة . و تكون الخوارزمية و الإجراء كما يلي :

for i := 2 to N do

{ أدمج $v[i]$ في مكانه بالقائمة $v[1..I-1]$ }



procedure Tri_insertion (var V : vecteur) ;

var i , k , aux : integer ;

begin

for i := 2 to N do

begin

إفراغ الذاكرة $v[i]$ للإزاحة ; $V[0] := V [i]$;

$k := I - 1$;

While $V [k] > aux$ do

begin عملية الإزاحة

$V [k + 1] := V [k] ; K := K - 1$;

end; { while }

عملية الدمج ; $V [k + 1] := v[0]$;

end ; { for }

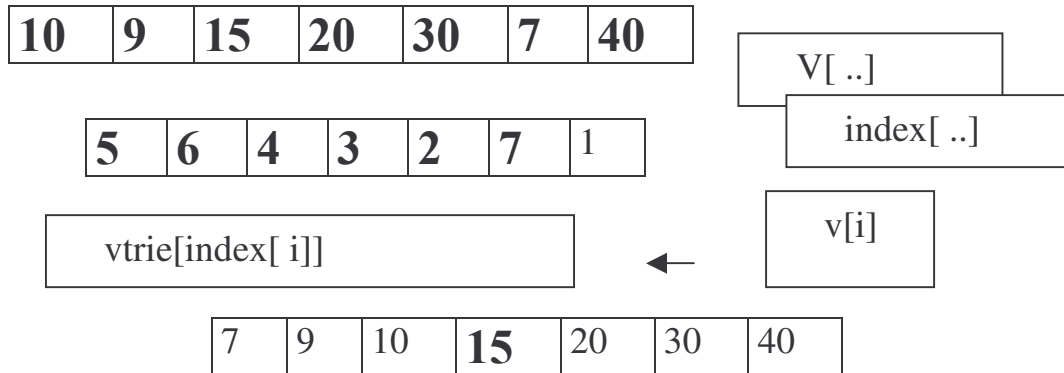
end ; { tri_insertion }

4.الترتيب بالعد .

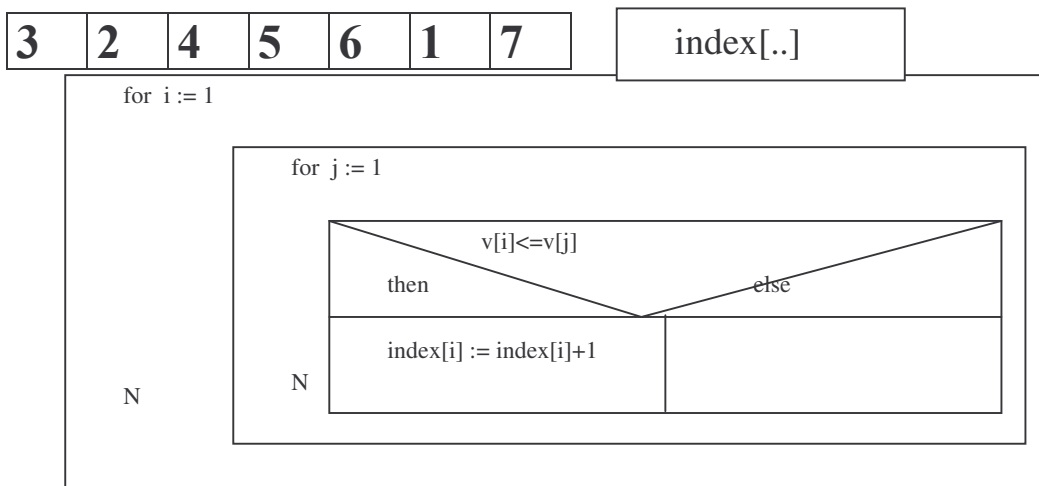
4.Tri par énumération

مبدأ الترتيب : علما بأن كل قائمة مرتبة من n عنصر إذا أخذنا منها عنصرا في المرتبة k , يعني أن فيها $(k-1)$ قيمة أصغر منه . وهذا هو المبدأ الذي نستخدمه عليه : نعد لكل قيمة v عدد القيم التي هي أصغر منها و نخزن هذا العدد في قائمة index فهرس ثم ترتب القائمة v في قائمة أخرى vTrie حسب الفهرس index .

مثال : قائمة من 7 عناصر نقوم بعدد المرات التي $v[i]$ أصغر أو يساوي $v[j]$ بدأً من i = 1 : عدد الحلقة الأولى 1° boucle for و $j = 1$ عدد الحلقة الثانية 2° boucle for .



و هكذا تصبح القائمة مرتبة تنازليا . إذا أردنا ترتيبا تصاعديا ما علينا إلا قلب العد من أصغر أو يساوي إلى أكبر أو يساوي ليكون الفهرس :



```
procedure tri( var v : vecteur );
```

```
  var vtrie,index:vecteur;
```

```
  i,j : integer ;
```

```
begin
```

```
  for i := 1 to n do
```

```
    for j := 1 to n do
```

```
      if v[i]<=v[j] then      index[i]:=index[i]+1;
```

ترتيب القائمة حسب الفهرس .

```
  for i:= 1 to n do
```

```
    vtrie[index[i]]:= v [i] ;
```

```
end;
```

5.الترتيب بالتبادل أو الإختزال .

5.Tri par transposition

مبدأ هذا الترتيب : مقارنة عنصرين متتاليين و التبادل بينهم إذا كان الترتيب

معكوس ثم العودة إلى الورااء للتأكد من ان الترتيب لم يتغير من فعل التبادل .

5	7	1	2	4	3	تبادلtransposition
5	1	7	2	4	3	الرجوع للوراءretour en arriere
1	5	7	2	4	3	تبادل
1	5	2	7	4	3	رجوع للوراء
1	2	5	7	4	3	تبادل
1	2	5	4	7	3	رجوع للخلف
1	2	4	5	7	3	تبادل
1	2	4	5	3	7	رجوع
1	2	4	3	5	7	رجوع
1	2	3	4	5	7	رتبت

```

PROCEDURE TRI(var v:vecteur );
var i ,ech : integer ;
    echage : boolean ;
begin
    i := 1; echage := false;
while (i<=(n-1))and(echage=true) do
    if v[i] >v[i+1] then
        begin
            ech := v[i]; v[i] := v[i+1]; v[i+1] := ech ;
            echage := true; i := i + 1;
        end
    else begin echage:=false; i := i + 1; end;
end;

```

6.الترتيب السريع.

6.Tri Rapide / quick sort

مبدأ هذا الترتيب يعتمد على الطريقة التراجعية récurisivité حيث تقسم القائمة إلى زميرين حسب قيمة مختارة توضع كعلامة : زمرة العناصر الصغرى أي الأصغر من العنصر العلامة و زمرة العناصر الكبرى أي الأكبر من العنصر العلامة . أو كما يقال زمرة يمينى و أخرى يسرى ثم كل زمرة بدورها تقسم إلى جزأين حسب نفس الأسلوب ... و نتابع التقسيم إلى أن تصبح الزمرة الأصغر مكونة من عنصرين يتم ترتيبهم بسهولة التبادل

مثال : 94 41 55 12 22 42 66 18 67 33

لنأخذ مثلاً 42 كقيمة وسطية لنفرق بها القائمة إلى زميرين : زمرة الأكبر من 42 و أخرى الأصغر من 42 . وذلك بتبادل أماكن العناصر الأكبر يحول على يمين 42 و الأصغر على اليسار .

94 41 55 67 42 66 18 12 33 :: هنا بدلنا بين 67 و 12

94 66 55 67 42 41 18 12 33 :: هنا بدلنا بين 66 و 41

ثم نتابع على يمين 42 و على يساره بنفس الطريقة .

33 12 18 41 42 67 55 66 94

نختار على اليمين مثلاً 55 لتقسيم الزمرة اليمنى فلا نجد على يمين 55 ما يبذل ولكن نجد 67 أكبر من 55 لذلك نبدلهما ثم على اليسار اخترنا 18 لتقسيم الزمرة اليسرى و كذلك لا نجد على يمين 18 ما يبذل فكل القيم أكبر من 18 و لكن على يسارها نجد 33 أكبر و توجد على اليسار نحوها بتبادلها مع 18 و تصبح القائمة كالتالي:

18 12 33 41 42 55 67 66 94

نتابع التقسيم فنجد على اليمين عنصرين قابلين للتبديل و هم 66 و 67 بقي على اليسار ما يحتاج إلى تبادل و هم 12 و 18 فيتم تبادلهم و تصبح القائمة مرتبة: 67 94

12 18 33 41 42 55 66

و هكذا تلاحظ أن القائمة رتبت بعدد قليل من الدورات و عمليات التبادل لذلك سمي بالترتيب السريع .

القيمة المختارة لفرق القائمة تنتخب عشوائياً : و يستحسن أن تختار كقيمة

وسيطية valeur médiane كالتالي:

B: limite basse d'une partition

H: limite haute d'une partition

ValeurMediane = partie entière (B+H) / 2

و تكون الخوارزمية هي نداء إلى إجراء الفرق بطريقة تراجعية .

و الإجراء كالتالي :

```
const N = 10 ;
Type Vecteur = Array [ 1..N ] of integer ;
Index = 1 ..N ;
Var V : Vecteur ;
Procedure Quick_sort( var V : vecteur ) ;
  procedure sort ( L , M : index ) ;
    var i , j : index ;
        x , w : integer ;
  begin{ sort }
    i := L ; j := M ;
    x := V[ ( L + M ) div 2]; { element median }
```

```

Repeat
  While V[ i ] < X Do i := i + 1 ;
  While X < V[ j ] Do j := j + 1 ;
  If i <= j Then { التبادل }
    begin{ then }
      W := V[ i ] ;
      V [ i ] := V [ j ] ;
      V [ j ] := W ;
      i := i + 1 ;
      j := j - 1 ;
    end;{ fin du then }
until i > j ;
if L < j Then Sort ( L , j ) ;{ appel recursif}
  { نداء الإجراء بالأسلوب التراجعي }
  if i < M Then Sort ( i , M );
end;{ fin de sort }
Begin { quick_sort }
  sort ( 1 , N ) ;{ corp de la procédure quick_sort}
end ; { quick_sort }

```

الباب التاسع : المجموعات .

Chapitre 9:Les ensembles / sets .

1. الهدف

1.But

لغة باسكال وفرت للمبرمج نوع المجموعات كوسيلة أخرى لإحداث بنية البيانات التي يحتاج إليها.

بما أن هذا النوع متداول في عالم الرياضيات و الهندسة و تصميم التـرجمـانات compilers و الدراسات النحوية اللغوية.

لا بد إذا من توفير هذا النوع من البيانات لكل من يريد التعمق في هندسة البرامج و كتابة تطبيقات متنوعة .

2. تعريف المجموعات.

2.Définition des ensembles.

المجموعة كِتْل أو تراكم للأشياء collection من غير تكرار أو ترتيب تأخذ من مجموعة معلومة .

مثال : المجموعة الأساسية : $U = \{ 1,2,3,4,5 \}$

الزمر $R=\{ 1,2 \}; S=\{ 1,2,3 \}; T=\{ 3,4,5 \}$: sous ensembles

الإتحاد : $R \cup T = \{ 1,2,3,4,5 \}$ union

التقاطع : $S \cap T = \{ 3 \}$ intersection

الفرق : $S - R = \{ 3 \}$ différence

التكملة : $\overline{S} = \{ 4,5 \}$ complément

المجموعة الفارغة : $T \cap R = \{ \}$ ensemble vide

$x \in S$: x ينتمي أو عنصر من S

المجموعة S تسمح بتكوين مجموعات منها و تسمى Power set of S تكتب $P(S) = 2^S$ و

هي مجموعة من الزمر من S : $P(s) = \{ T \mid T \subseteq S \}$

مثال :

. $P(s) = \{ \{ \} \}$; فإن $S = \{ \}$ إذا كانت $S = \{ 1,2 \}$; $P(s) = \{ \{ \}, \{ 1 \}, \{ 2 \}, \{ 1,2 \} \}$

أو ; $P(s) = \{ \{ \}, \{ a \}, \{ b \}, \{ a,b \} \}$ فإن $S = \{ a,b \}$

$P(s)$ تسمى مجموعة القوى Power set. وهي مجموعة الزمر S وهي 2^2

3. تعريف باسكال للمجموعات.

3.Les ensembles en Pascal

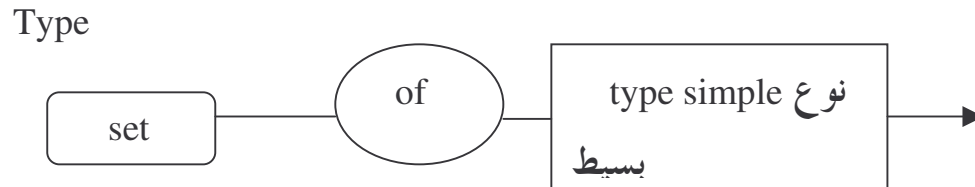
set هو نوع جديد يوفره باسكال حسب قواعد :

Type set_type = Set of T;

نوع المجموعة الأساسية التي منها تكون الزمر و يجب أن يكون من النوع التعدادي type

. intervalle énuméré أو نوع مجال

المتغير S زمرة من مجموعة الزمر أو من مجموعة القوى power set .
و القاعدية النحوية :



و يعرفها BPW 7 كما يلي :

The base type of a set must be an ordinal type with no more than 256 possible values .

The ordinal values of the upper and lower bounds of the base type must be between 0 and 255 .

A set constructor, which denotes a set-type value, is formed by writing expressions within brackets. Each expression denotes a value of the set .

The notation [] denotes the empty set, which is compatible with all set types .

النوع البسيط الذي يمكن إستعماله هو :

```

Type voiture = ( R4 , R5, R18 ) ;
ensemble_voiture = set of Voiture ;
Entier = 0..100 ;
ensemble_entier = set of entier ;
Caracteres = 'A'...'Z';
ensemble_car = set of caracteres ;
Integer_set = set of 0..30 ;
Car_set = Set of 'أ'..'ي' ;
ensemble_logque = Set of Boolean ;
  
```

لا يمكن إستعمال integer, Char, Real كمجموعة أساسية .

```

Var small_set : set of 0..2;
EV : ensemble_voiture ;
  
```

power small_set زمرة من $8 = 2^3$ زمرة مختلفة وهي جميع عناصر مجموعة القوى
 . {2,1,0} set

الزمرة هي قيمة من نوع مجموعة تعرّف بعدّ عناصرها بين حاضنتين [..]
 أمثلة :

```
Var A,B,C : Set of 'A'..'Z';
....
A := [ ] ; ensemble vide
B := [ 'A','B','C' ] ;
C := [ 'X'..'Z' ] ;
A := B + C ;
```

4. العمليات الممكنة على المجموعات.

4. Operations globales sur les ensembles

تقريباً كل العمليات ممكنة على المجموعات : التعيين affectation و المقارنة بالتساوي
 أو الاختلاف :- = و <> و الإتحاد :- + و الفرق :- - و اختبار هل عنصر من مجموعة
 :- in و اختبار هل يوجد زمرة داخل مجموعة أو زمرة أخرى :- = و <= .
 ولكن لا يمكن قراءة مجموعة مباشرة أو كتابتها مباشرة .

1.4 الإتحاد .

4.1 L'union

يرمز إليه في باسكال :- + أي مجموعة العناصر التي نجدها في الزمرة الأولى و الثانية و

... : { 'A'...'Z' } ; ['M'..'Z'] + ['A'..'M'] Z :=

Y := A + B + C ;

أو الثالثة مع عدم تكرار العناصر .

2.4 التقاطع .

4.2 L'intersection

هو مجموعة العناصر التي نجد مثلها في الزمرة الأولى والثانية .

Var x , y , z : Set of 1..10 ;

begin

x := [1..5] ; y := [2..7] ;

Z := x * y ; { 2,3,4,5} : نتيجة التقاطع هي

3.4 الفرق.

4.3 La différence

Type Days = (monday, tuesday ,wensday ,thursday
 ,friday,saturday,sunday) ;

weekset = Set of days ;

Var D : Days ;

Wd, wend : Weekset ;

D := Monday ;

Wd := [monday,tuesday,wensday,thursday);

Wend := [friday,saturday,sunday];

Wd := Wd - [D] ; { tuesday,wensday,thursday};

Wend := Wend - [friday] ; { Saturday , Sunday } ;

4.4 التساوي و الاختلاف.

4.4 Égalité et inégalité

If a = [1,2,3] then

if a < > [1,2] Then.....

نتيجة المقارنة منطقية : True أو false .

5.4 العنصر عضو من مجموعة.

4.5 L'élément appartient ou In

العامل In يستعمل لإختبار هل عنصر ما يوجد داخل زمرة أو مجموعة

Repeat

.....

writeln('repeter o(ui) ou n(on);readln(rep);

until rep In ['n','N']; { النتيجة منطقية }

if 3 In a Then

6.4 الإحتواء أو المجموعة العليا و المجموعة التحتية أو السفلى.

4.6 Super set and Subset

if [1,2] <= a Then

If [1,2,5] >= a Then

النتيجة تكون منطقية : true أو false .

الباب العاشر: المصفوفة من نون بعد.

Chapitre 10:Tableaux a n Dimensions.

1.تعريف المصفوفة.

1.Définition de la Matrice ou Tableau

عندما تكون لدينا مجموعة من البيانات لشيء واحد و لكن متنوعة و من شتى المصادر
وجب علينا إيجاد طريقة لعرضها و إستغلالها و ذلك بوضعها في جدول متعدد المداخل

. tableau a plusieurs entrée

مثلا: نتائج الطلبة في عدة مواد يمكن تمثيلها في جدول من عدة أبعاد dimensions ليكون
لدينا 4 طلبة و لهم نتائج: 4 نقاط في 5 مواد .

كيف يمكن التعبير عن هذه البنية للبيانات ؟

Type Resultat = Array[1..4] of { لتمثيل 4 طلبة }

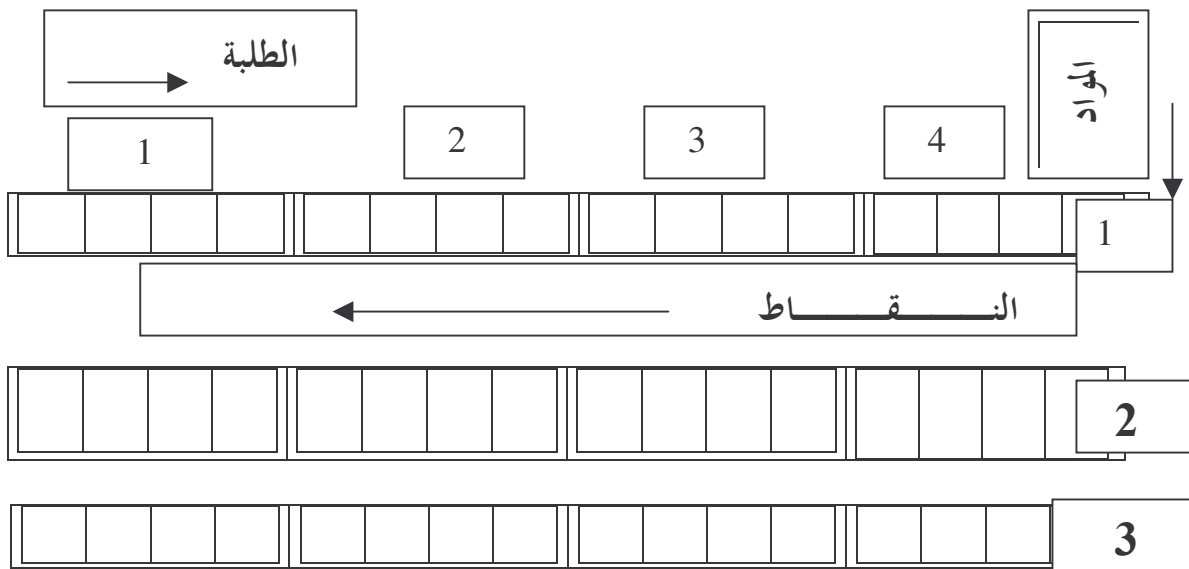
Array[1..5] of { لتمثيل 5 مواد }

Array [1..4] of real ; { لتمثيل 4 نقاط }

فهي تشبه مكعب :

1	2	3	4	5
2				
3				
4				

وهي قائمة من 3 أبعاد : تحتوي على 80 ذاكرة .



مثال آخر : استعمال الزمن: emploi du temps

الأيام	السبت	الأحد	الاثنين	الثلاثاء	الأربعاء	الخميس	الجمعة
الوقت							

							10-8
							16-14

يصرح به كالتالي :

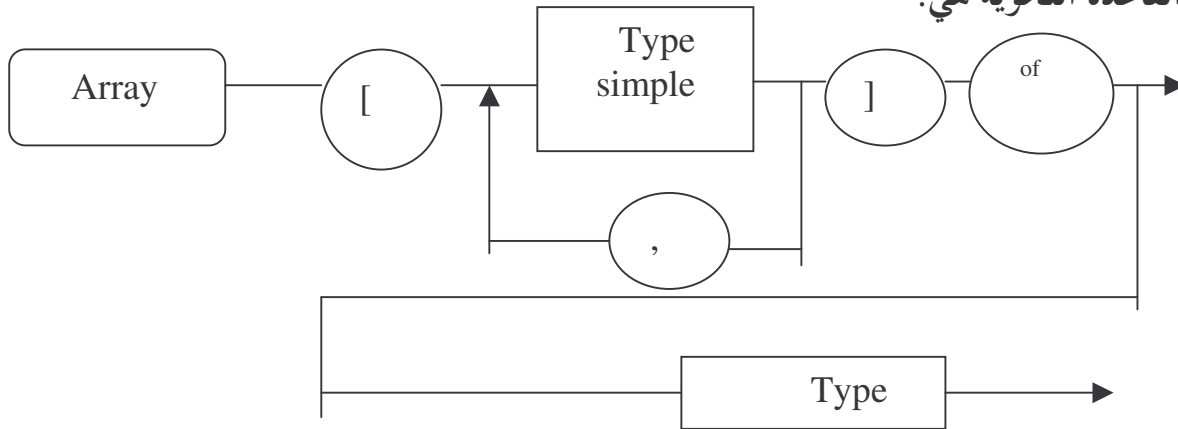
Type Emploi_temps = Array[1..8] of
Array[1..5] of string[20];

الخلاصة: المصفوفة متكونة من قائمتين أو أكثر كل واحدة تحمل بيانات.

2. التصريح بالمصفوفة.

2.Déclaration de Matrice.

القاعدة النحوية هي:



type simple النوع البسيط هو كل من المجال أو نوع مصرح به أو boolean أو char ولا يكون integer و لا real : و هو نوع الدليل أو الفهرس index / indice.

type : يمكن أن يكون من الأنواع البسيط : القياسية الأربع أو الأنواع البنيوية, مثل :
array أو set أو record و لا يكون من نوع file أي جذاذة fichier فهو غير مسموح به .
يفصل بين الأدلة indices بفاصلة لتحديد نوع المصفوفة فهي مصفوفة ثنائية أو ثلاثية أو رباعية أو ... حسب عدد الأدلة أو الفهارس .

3.المصفوفة الثنائية أو الجدول.

3.Matrice a 2 dimensions ou tableau

مثال :أكتب برنامجا يوفر الخدمات التالية :

1.قراءة نفقات الأسرة خلال الأسبوع من مواد غذائية..كتب و مجالات ملابس .. أمور

شتى .

2.عرض النفقات الخاصة بيوم ما .

3. عرض محتوى كل الجدول .

**الحل : نتصور الجدول حتى نتمكن من التصريح به .

فهو جدول من 8 أعمدة : 7 للأيام و عمود لتعريف النفقات .

ومن 5 أسطر : 4 للنفقات و سطر لتعريف الأيام .

الأيام ←	السبت	الأحد	الاثنين	الثلاثاء	الأربعاء	الخميس	الجمعة
↓ المواد	1	2	3	4	5	6	7
مواد غذائية 1:	150	165
التوثيق:2	0	12	15
ملابس:3	56
أمور شتى:4	58	47	69

فهو عند التصريح به جدول من 4 أسطر lignes و 7 أعمدة colonnes

و يكون النفاذ إلى السطر بدليله المشار إليه هنا بالمواد و النفاذ إلى الأعمدة بدليل الأيام .و

قد نختصر النفاذ إلى الأسطر و الأعمدة بالأرقام فقط أي الأسطر من 1..4 و الأعمدة من 1..7.

7 .

و يكون التصريح كما يلي :

```
Program Matrice ;
Uses wincrt;
Type Tableau = Array [ 1..4 , 1..7 ] of Real ;
Var Tab : Tableau ;
    Choix : Char ;
```

1. إجراء القراءة :

```
procedure lit_matrice ( var T : Tableau ) ;
    var ligne , colonne : 1..7 ;
begin
    writeln('أدخل نفقات الأسبوع لكل من الأشياء التالية: ');
    writeln('1: مواد غذائية. 2: وثائق. 3: ملابس. 4: أمور شتى');
    For ligne := 1 To 4 do
        For colonne := 1 to 7 Do
            Readln ( T [ ligne , colonne ] ) ;
        end;
    end;
```

يجب تحسين القراءة بإضافة تعاليق للمستعمل ليعرف ماذا عليه أن يفعل في كل خطوة و يدخل البيان المناسب في مكانه .

2. عرض النفقات الخاصة بيوم ما .

```
procedure depense_Jour ( var T : tableau ) ;
    var produit , jour : 1..7 ;
begin
    writeln('choisissez le jour par un nombre 1: Samedi
;2:dimanche;3:lundi;4:mardi;5:mercredi;6:jeudi;7:vendredi');
    readln( jour ) ;
    For produit := 1 to 4 do
        Write ( T [ produit , jour ] ) ;
    end;
```

هنا عرضنا طريقة النفاذ إلى عنصر ما في الجدول و ذلك بمعرفة دليل السطر و العمود.

3. عرض محتوى الجدول .

```
procedure listing ( var T : tableau ) ;
    var i , j : 1..7 ;
```

```

begin
  writeln(' tableau des depenses hebdomadaire ');
  For i := 1 to 4 do
    begin
      For j := 1 To 7 do      write ( T [ i, j ] ) ;
      writeln ;
    end;
  end;

```

4. البرنامج الرئيسي :

```

begin      write (' Menu : 1 . Lecture de la matrice :dépenses de toute la
semaine ; 2.listing des dépenses d'un jour x ; 3. listing de toute les dépenses
hebdomadaire ');
  readln( choix ) ;
  case choix of
    '1' : lit_matrice ( tab );
    '2' : depense_jour ( tab ) ;
    '3' : listing ( tab ) ;
  end; { case }
end.{ fin du prog. principal }

```

1.3 المصفوفة الوحدة.

3.1 Matrice unité

المصفوفة الوحدة تتمثل في أن جميع عناصرها صفر إلا العناصر الموجودة في القطر :
diagonal .

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{مثال :} \quad \forall i; a_{ii}=1 \text{ و } a_{ij}=0$$

و يكون الإجراء كالتالي :

```

Const Nbligne = 3 ;
  Nbcolonne = 3;
Type indligne = 1..Nbligne ;
  Indcolonne = 1..Nbcolonne;

```

```

    Ligne = Array [ indcolonne ] of real ;
    Matrice = array [ indligne ] of ligne ;
Var Mat1,Mat2,Mat3 : Matrice ;
Procedure unit ( var M : matrice );
    I : indligne ; J : indcolonne;
Begin{unit}
    For i := 1 to Nbligne do
        Begin{1}
            For j := 1 to Nbcolonne do
                Begin{2}
                    If I = J Then M [ i , j ] := 1
                        Else M [ i , j ] := 0 ;
                    Write ( M [ i , j ] ) ;
                End;{2}
            Writeln ;
        End ;{1}
    End; { unit }

```

2.3 جمع مصفوفتين.

3.2 Addition de 2 matrices

```

Procedure add_matrices ( var M1,M2,M3 : Matrice );
    Var i : indcolonne ;
        J : indligne ;
Begin
    For i := 1 to indcolonne do
        For j := 1 indligne do
            M3[i,j] := M1[i,j] + M[i,j] ;
        End;
    End;

```

عند النداء نقدم الوسطاء الفعليين :

```
add_matrices(mat1,mat2,mat3);
```

3.3 قلب السطر إلى عمود.

3.3 Transposée d'une matrice

$$TM = \begin{pmatrix} 147 \\ 258 \\ 369 \end{pmatrix} \quad M = \begin{pmatrix} 123 \\ 456 \\ 789 \end{pmatrix} \quad \text{مثلا : نتيجة قلب السطر إلى عمود :}$$

و يكون الإجراء :

```

Procedure transpose_mat ( var m,mt :matrice);
Var i : indligne ;
    J : indlcolonne ;
Begin
    For i := 1 to Nbligne do
        For 1 to Nbcolonne do
            Mt [ i , j ] := M[ j , I ] ;
        End;
    End;

```

4.3 تمارين .

3.4 Exercises

1. أكتب إجراء ضرب مصفوفة مربعة بشعاع .
produit d'une matrice carrée par un vecteur .
2. أكتب إجراء ضرب مصفوفتين .
Produit de 2 matrices
3. أكتب إجراء يجد محدد مصفوفة مربعة .
déterminant d'une matrice carrée.
4. أكتب في الختام البرنامج الرئيسي الذي يقدم جميع الخدمات السابقة الخاصة بالمصفوفة على شكل اختياري sous forme de menu .
- 4.المصفوفة من 3 أبعاد .

4.Matrice tridimensionnelle

- مثال :نود كتابة برنامج لمعالجة نتائج التلاميذ في الامتحانات و على سبيل المثال نأخذ
- 15 تلميذ و 3 مواد : اللغة , الرياضيات , العلوم و لكل مادة 3 نقاط : فرض , اختبار و

معدل .

**الحل :

1. تمثيل بنية البيانات : structure de donnée

الرياضيات			العلوم			اللغة			التلميذ: 1
08.3	10	05	11.6	10	15	10.6	10	12	
...	
...	

2. التصريحات ببنية البيانات:

Type Nb_eleve = 1..15 ;

Nb_mat = 1..3 ;

Nb_Note = 1..3;

Table_3D = Array[Nb_eleve, Nb_mat , Nb_Note] of Real;

1.4 قراءة مصفوفة من ثلاث أبعاد .

4.1 Lecture d'une matrice à 3 dimensions

Procedure lecture_matrice (var T3d : Table_3D);

Var I :Nb_eleve; J : Nb_mat; K :Nb_note;

Begin

Writeln(' Lecture d'une matrice a 3 dimensions');

For i := 1 To 15 do

For j := 1 To 3 do

For k := 1 To 3 Do

Read(T3d [i , j , k]);

End;

2.4 البحث عن عنصر من مصفوفة.

4.2 Recherche d'un élément d'une matrice

البحث عن نقاط تلميذ ما :

```

Procedure cherche ( var T3d : Table_3d ) ;
  Var i , j ,k : integer ;
  Begin{ cherche}
    Write(' donnez le numero de l'eleve : ');    Readln( I ) ;
    Writeln( ' voici ces notes par matières ');
    For J := 1 To 3 Do
      Begin{ 1}
        Write ( ' Note N°: ', J );
        For K := 1 To 3 Do
          Begin{2}
            Write ('matiere N°:,k ');
            Write ( T3d [i , j ,k ] );
          End;{2}    Writeln;
        End;{1}
      End;{ cherche}

```

5. المصفوفة من 4 أبعاد .

5. Matrice a 4 dimensions

نريد تمثيل نتائج العمل المدرسي السنوي :

1. السداسي الأول و الثاني .
2. النقاط : الفرض الإختبار و المعدل .
3. المواد : رياضيات - علوم - فيزياء - لغة.
4. التلاميذ: 5

السداسي 1			
4	3	2	1
			2
			3
			4

السداسي 2			
2	3	4	5

و يكون النفاذ إلى أي عنصر عبر 4 فهارس index / indices مثل : $A[i,j,k,L];$

6.تمارين .

6.Exercices

1.ما الفرق بين التصريحين التاليين :

Type L = array[1..10] of array[1..10] of Char ;

M = array [1..10,1..10] of char ;

2. سألتك شركة إدارة عمارات أن تكتب لها برنامجا حول كراء المنازل .

عدد العمارات 200 من نفس النوع : عدد الطبقات و عدد المساكن .

كل عمارة من 5 طوابق و كل طابق من 10 مساكن : A,B,C,D,E,F,G,H,I,J . برنامجك

يتابع عملية قبض الكراء .

تدخل البيانات سطرا بسطر , لكل منزل : رقم الطابق بحرف و رقم المنزل و قيمة الكراء

. مثلا : 4000Da : 5B .

الدخل غير مرتب حسب الطابق و المنزل و لكن عشوائي aléatoire .

أكتب إجراء يضع الكراء في جدول خاص للعرض و الفرز و إخراج الفواتير لكل من

يطلبها و للعمليات الإحصائية .

أكتب إجراء عرض كراء كل منزل حسب معرفة رقم الطابق و المنزل.

أكتب إجراء مجموع الكراء لكل طابق .

أكتب إجراء المجموع الكلي بالعمارة و إجراء المجموع الكلي بجميع العمارات.

كل هذه الخدمات تقدم على شكل خيارات : Menu .

الباب الحادي عشر : بنية بيانات معالجة النصوص .

Chapitre 11: Traitement de Texte /Data

structure : Text Processing.

1. الهدف .

1. But

كما تعلم فإن البرمجة تشمل معالجة الأعداد و العمليات الحسابية المعقدة و تشمل جميع أشكال النصوص و الرسوم و الصور المتحركة و الأصوات, لكل من هذه الخدمات كتبت برامج تطبيقية و التي تعرف بـ: software بالإنجليزية و بالفرنسية: logiciel. أهم و أكثر تداولاً محرر النصوص و الصفحات éditeur de texte ou de page مثل Word لـ Microsoft أو الناشر المكتبي للـ: Macintosh و غيرها .

لكتابة هذا النوع من البرامج التطبيقية الخاصة بمعالجة سلاسل الحركات chaîne de string / caractères يوفر باسكال على المستوى الإضافي نوع String. أما على المستوى القياسي فلا نجد إلا النوع Char.

لذلك سنحاول إضافة هذا النوع : سلسلة الحركات String كتطبيق و تمارين . أما إذا أردت إستعمالها فهي موفرة في BP 7 و كذلك في Ms-Pascal و FreePascal. و جميع الترجمات ...

2. إحداث النوع : سلسلة الحركات في باسكال القياسي.

2. Implementing the string package in standard

Pascal

سنضرب أمثلة عن كيفية إحداث هذا النوع و لك أن تتمرن على كتابة كل الدوال و الإجراءات التي تجدها موفرة في الترجمات compilers.

المثال الأول :

1. التصريح بالنوع :


```

Const Max = 128;
Type stsize = 0..Max ;
  String = Record
    Long : stsize ;
    Text : Array [1..max] of char ;
  End; { string record}

```

2. قراءة سلسلة :

```

Procedure Readst ( var S : string );
Begin
  S.long := 0 ;
  While ( not eoln) and ( S.long < Max ) Do
    Begin
      S.long := S.long + 1 ;
      Read ( S.text [ s.long ] ) ;
    End;{ while }
  Readln;
End;{ readst}

```

لإستعمالها نناديها لقراءة أي سلسلة من 0 إلى 128 حركة :

3. كتابة سلسلة حركات :

```

Procedure Writest( var S : string );
  Var i : stsize;
Begin
  For I := 1 To S.long Do      Write ( S.Text[ I ] );
End;

```

لإستعمالها نناديها : Writest(v); بالوسيط الفعلي V .

المثال الثاني :

التصريح بهذا النوع بأسلوب الـ: Packed array

```

Const stlong = 80 ;
Type string = Packed Array [ 1..stlong ] of Char ;
Var Name : string ;
Procedure Readst ( var S : String ) ;
  Var ctr , c : integer ;
Begin
  Ctr := 1 ;

```

```

While ( not eoln ) and ( ctr < stlong ) do
  Begin
    Read( S[ ctr ] ) ;
    Ctr : ctr + 1 ;
  End;{ while }
  For c := ctr to stlong do S[c] := ' ';
{   ملاً ما بقي من السلسلة بالفراغات   }
End;

```

في هذا المثال صرحنا بالنوع string على شكل جدول متراص packed array .

الجدول من نوع packed تمتاز عن غيرها بإمكانية التعيين إلى بعضها و مقارنتها : = و < و > و <= و >= .

3.مسألة .

3. Projet

بما أن باسكال القياسي لا يوفر سلسلة الحركات string طُلب منك إحداث هذا النوع و أدوات إستعماله من إجراءات و دوال .

النوع يسمى Lstring أي large string من 0 إلى 32767 حركة .

نقترح عليك الإجراءات و الدوال التالية :

1.إجراء ربط سلسلتين الواحدة في الأخرى:

```
concat ( var A,B :string);
```

2.إجراء نقل سلسلة copy إلى أخرى : copyStr(var A,B :string);

3.إجراء حذف n حركة delete من سلسلة S بداية من S[i] :

```
Delete ( var S : string ;i,n :integer );
```

4.إجراء دمج insert سلسلة s مباشرة بعد D[i] :

```
Insert( S :string;var D:string; i : integer );
```

5.إجراء البحث عن سلسلة:أي كلمة ما و إرجاع النتيجة بعرض كل الكلمة الواحدة تلو

الأخرى بلون مقلوب. Scanne (var mot,S :string);

و قد تختار كتابة دوال محل إجراءات لتقديم هذه الخدمات .
 أكتب البرنامج الرئيسي بعرض هذه الخدمات على شكل menu .
 هذه المسألة تصلح كمشروع نهاية دراسة إذا أنجزت باللغة العربية أي إذا تم كتابة
 البرنامج بترجمان باسكال تحت : windowsMe أو windows2000 و باستعمال الخطوط
 العربية التي يوفرها نظام windowsMe ليصبح برنامجك محرر نصوص صغير بالعربية
 تستطيع تطويره خطوة خطوة بعدما تتمكن من البرمجة بلغة الأشياء و الصور و النوافذ و هو
 ممكن بترجمان BorlandPascal for windows و FreePascal .

الباب الثاني عشر : التسجيل .

Chapitre 12 : L'enregistrement / Record

1.الهدف .

1.But

من المعلوم أن المعلومات التي نبحثها متوفرة في الحيات اليومية ليست بالضرورة من نوع
 واحد : بيانات من نوع صحيح أو حقيقي أو حركات فقط أو منطقي إنما هي مزيج من
 الأنواع و أبسط مثال على ذلك دفتر العائلة carnet de famille نجد فيه المعلومات من نوع
 الحركات و الأعداد :الأسماء و التواريخ . لغة باسكال توفر بنية التسجيل Record لتمكّنك
 تسجيل هذه المعلومات كوحدة entité و بجميع الأشكال .

2.تعريف التسجيل .

2.Définition de l'enregistrement.

إذا أخذنا بطاقة التعريف لشخص ما فهي مجموعة من البيانات من أنواع مختلفة :
 الإسم و اللقب - تاريخ الازدياد - العلامات الخاصة - طوله - التاريخ- الصورة - الخاتم
إلى آخره .مجموعة هذه البيانات تمثل وحدة entite و كل عنصر منها élément يسمى
 حقل Champs و البنية ككل تسمى تسجيل record/enregistrement و هي بنية مركبة من

عناصر بسيطة أو/و مركبة كذلك أي يمكن أن تحتوي بدورها على تسجيل.
و هكذا يمكن تمثيل كل بنية للبيانات تعرض علينا . ما عدا بنية الصورة أو الأشياء
الطبيعية الأخرى و التي سنتطرق إليها لاحقا إن شاء الله .
لو صرحنا ببنية بطاقة التعريف بالطريقة البسيطة تكون كما يلي :

Var

```
Nom_prenom : string ( 30 );
Date_naissance : Integer ;
Adresse : string (45);
Taille: real;
....
```

هذا بالنسبة لشخص واحد. و لو أردنا معالجة مجموعة من الأشخاص لا بد من إستعمال
بنية القائمة :

```
var      Liste_nom_prenom : Array [ 1..N] of string(30) ;
      liste_date_naissance : Array [ 1..N] of integer ;
      .....
```

فهي مجموعة قوائم بحسب عدد أنواع البيانات و لا تُظهر أي ترابط للبيانات فيما
بينها أو تُشكل معلومة موحدة, لذلك فإن بنية التسجيل تمثل الحل الأنسب لتمثيل
الوحدة المعلوماتية كالتالي :

تعريف Identite			
Taille	Adresse	Date_naissance	Nom_prenom
1.75	حي العقلاء	1955 05 12	عبد الله الكريم

هذا إذا تسجيل لتعريف شخص ما : Identité: مكون من 4 حقول champs كل حقل
معرف بإسم و نوع بيان .

3. التصريح بالتسجيل .

3. Déclaration de l'enregistrement.

كيف نعلن للترجمان بأننا نريد إستعمال بنية التسجيل ؟

مثل تعريف شخص كما تقدم :

Type Identite = **Record**

```
Nom_prenom : String ( 30);
Date_naissance : string (10);
Adresse : String( 50);
Taille : Real ;
End;
```



syntaxe: القاعدة النحوية

Type «Identificateur du Record» = **Record**

« تعريف الحقل Ident . du Champs » : « Type de donnee»;
.....;

End ; { نهاية التسجيل }

Var « تعريف المتغير التسجيل » : « Identificteur du type
Record»;

الكلمة الخاصة record تعلن عن نوع التسجيل و تنتهي دائما بالكلمة الخاصة end و كل تعريف لنوع التسجيل و الحقول يتبع قواعد الإعلان عن التعاريف و مجال هذا التعريف هو التسجيل فقط أي أن كل تعريف لحقل ما بتسجيل لا مجال له إلا التسجيل نفسه و هذا يعني أن حقلًا آخر بنفس الاسم ممكن في تسجيل آخر . نوع الحقل كذلك يمكن أن يكون بسيطًا أو مركبًا أي integer,real, char, boolean,array,set , string ,Record.:

و هي إذا كل الأنواع ما عدا الجذاذة file .

4.العمليات الممكنة الشاملة على التسجيل.

4.Opérations globales sur les enregistrements

العمليات التالية ممكنة بشرط أن تكون التسجيلات من نفس النوع :

التعيين: L'affectation:

المقارنة بالتساوي و الاختلاف: (= و < >) .

Type Personne = Record

مثال:

```
Numero_social : integer ;
Nom_prenom : String( 30);
```

```

        Adresse      :String( 40);
    End;
    Var P1,P2 : Personne ;
    .....

```

```

    P1  :=    P2 ; { l'affectation التعيين }

```

```

    If    P1 = P2 Then .....

```

```

    If    P2 <> P1 Then .....

```

أما العمليات مثل : If P1 < P2 Then فهي غير مسموح بها .

5.النفاز إلى الحقول,

5.Accès au champs

من المثال السابق نمثل التسجيل بذاكرة كبيرة مركبة من 4 حقول تمثل ذاكرات لكل منها إسم أو تعريف .

للنفاز إلى أي حقل نمر بإسم التسجيل ثم إسم الحقل

التسجيل P1			
Adresse		Nom_prenom	Numero_social
.....	

النفاز يعني قراءة محتوى حقل أو كتابة بيان فيه أو التعيين إليه .

القاعدة المتبعة هي : أولا كتابة إسم التسجيل ثم كتابة إسم الحقل بعد النقطة مثال : readln(

```

        P1.Numero_social );
    Readln( P1.Nom_prenom);
    Readln(P1.Adresse);
    Writeln( P2.numero_social );
    Writeln(P2.nom_prenom);
    Writeln( P2.adresse);
    P1.numero_social := 1001;
    P1.Nom_prenom := 'Ali el Baraka';
    P1.adresse := 'cite des normaux Algiers';

```

القراءة

الكتابة

التعيين

لا يجوز إستعمال : read(P1) أو write(P2) .

*تمرين : أكتب إجراء قراءة معلومة خاصة بتعريف كتاب و كتابة أي بيان من المعلومة حسب إختيار المستعمل . يعرف الكتاب بـ:

رقم تسجيله في المكتبة: N° inventaire

إسم الكاتب: Nom de l'auteur

العنوان : Titre

إسم الناشر: éditeur

الملخص: Résumé

*الحل : أولا نقوم برسم صورة التسجيل كذاكرة كبيرة فيها حقول:

Livre				
résumé	Editeur	Titre	Nom_auteur	N°inventaire
.....

```

Type Livre      =      Record
  N_inventaire   :      integer;
  Nom_auteur:    String(30);
  Titre         :      string(25);
  Editeur       :      String(20);
  Resume        :      String(200);
End;
Var      LV      :      Livre ;
  
```

إجراء قراءة التسجيل : حقل بحقل .

```

Procedure      lit_enregistrement( var  L      :      livre );
Begin
  Write('donner le N° d'inventaire :');
  Readln( L.N_inventaire );
  Writ('Titre :');Readln(L.titre);
  Write ('Auteur :');Readln( L.auteur);
  Write ('Editeur:');Readln(L.editeur);
  Write ('resume:');while not eof Read( L.resume);
End;
  
```

إجراء عرض أي بيان أو حقل من التسجيل :

```

Procedure Ecris_champ( var L : livre );
  Var c : char ;
  Begin
  Writeln ( ' contenu de l'enregistrement Livre :');
  Write ( ' 1:N°inventaire ;2:Nom auteur;3:Titre
;4:Éditeur;5:résumé;Choisissez un Nombre / champs ');
  Redaln( c );
  Case c      of
  '1'       :   writeln(L.N_inventaire);
  '2'       :   writeln(L.Nom_auteur);
  '3'       :   writeln(L.Titre);
  '4'       :   writeln(L.resume);
  end;{ case }
end;{ecris_champ}

```

6.التعليمة With .

6.L'instruction With

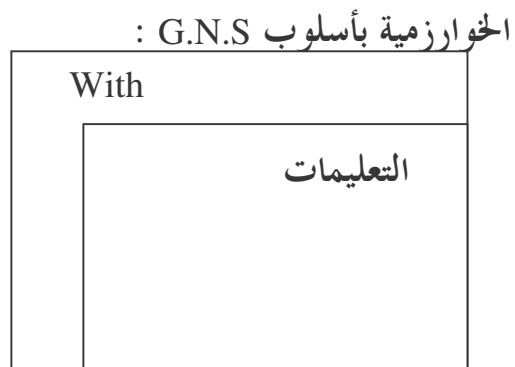
كما تلاحظ فإن النفاذ إلى أي حقل يكون باسم التسجيل ثم إسم الحقل و هو عمل تكراري، لتفادي هذا الحرج يوفر باسكال طريقة لتسهيل التعامل بالحقول و ذلك بإستعمال تعليمة with كالتالي :

With L Do

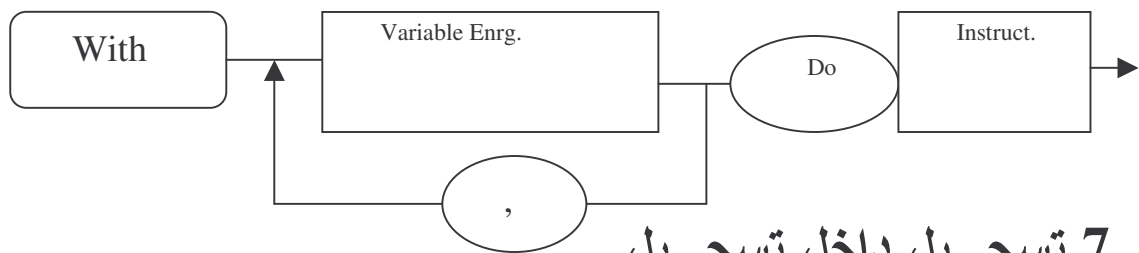
```

Begin
  Readln( n_inventaire );
  Readln( Titre );
  Readln(Nom_ auteur );
  Readln(resume);
End;{ fin du with }

```



و المعنى أن With تحمل إسم التسجيل و تضيفه إلى كل حقل داخل قالبها .
و القاعدة syntaxe .



7.تسجيل داخل تسجيل .

7.Enregistrements imbriqués

مثال : تعريف شخص :الإسم و اللقب و العنوان و تاريخ و مكان الإزدياد كتسجيل

داخل التسجيل .

بنية هذا التسجيل تكون كما يلي :

personne					
adresse	Date_lieu_naissance				Nom_prenom
	lieu	a	mois	Jour

و التصريح به :

```

Type  Personne  =  Record
Nom_prenom      :String( 30);
Date_lieu_naissance  :Record
    Jour  :    1..31;
    Mois  :    1..12;
    Annee  :1900..2020;
    Lieu   :    string(20);
    End;{date_lieu_naissance}
Adresse       :    string(45);
End;{personne}
Var id : personne ;
  
```

إجراء القراءة :

```

Procedure lit ( var p : personne );
Begin
  With P do
    Begin
      Readln( nom_prenom);
    
```

```

        With date_lieu_naissance ) do
            Begin
                Readln(jour);readln(mois);
                Readln( annee);readln( lieu);
            End{with date_...}
            Readln( adresse );
        End;{with P}
    End;{lit}

```

يمكن إستعمال with واحد نعين فيه إسم التسجيل الأول و الثاني كالتالي :
إجراء الكتابة:

```

Procedure ecrit ( var T : personne );
Begin
    With T , date_lieu_naissance Do
        Begin
            Writeln( Nom_prenom);
            Write(jour);write(mois);write(annee);
            Writeln( lieu);
            Writeln( adresse );
        End;
    End;

```

طريقة أخرى للتصريح بالتسجيل الداخلي هي إخراجها من التسجيل الشامل له و التصريح
به كنوع آخر ثم إستعماله كنوع جديد : مثل :

```

Type Date_lieu_naissance = record
    Jour : 1..31;
    Mois:1..12;
    Annnee:1900..2200;
    Lieu      :      string(45);
End;
Personne = record
    Nom_prenom :string ( 30);
    Dt_L_Nss   :      date_lieu_naissance;
    Adresse    :      string( 50);
End;
Var  p       :personne ;

```

8.التسجيل المتغير.

8.Enrégistrement variant / variant record

في المعلوماتية data processing / informatique لا تكون المعلومات دائما ثابتة بل تكون في كثير من الأحيان متغيرة حسب شروط معينة.

فهي بنية للبيانات حركية dynamique غير ساكنة statique فمثلا:

في تعريف الشخص نجد المتزوج و الأعزب و المطلق و المتوفى عنه وفي كل حالة نضيف شيء أو آخر فالمتزوج نضيف له إسم الزوجة و الأطفال و المطلق أسماء الأطفال و الأعزب لا نضيف شيء و كلها حالات يجب التكفل بها في التسجيل .

نوضحها في بنية التسجيل المتغير كالتالي :

Situation_familiale	adresse	Dat_nais	Nom_prenom															
<table><tr><td colspan="3">C</td></tr><tr><td>Nb_ef</td><td>N_ep</td><td>M</td></tr><tr><td></td><td></td><td></td></tr><tr><td>Nb_ef</td><td colspan="2">D أو V</td></tr><tr><td></td><td colspan="2"></td></tr></table>	C			Nb_ef	N_ep	M				Nb_ef	D أو V							
C																		
Nb_ef	N_ep	M																
Nb_ef	D أو V																	

الحقل : الحالة المدنية situation familiale ينقسم إلى ثلاث حالات :

إذا كانت قيمته c أي أعزب célibataire فلا يضاف شيء

و إذا كانت قيمته M يعني متزوج marié يضاف الحقول إسم الزوجة و عدد الأطفال .

و إذا كانت قيمته v زوجته متوفاة أو D مطلقة : يضاف حقل واحد عدد الأطفال nb.

. enfant

و يصرح به كما يلي :

Type Personne = record

Nom_prenom : string(30);

Date_naissance : integer ;

Adresse : string(50);

Case situation_familiale : Char of

```

'C' : ( );
'M' : ( nom_epouse :string(20);
       Nombre_enfant : integer ; );
'V','D':( Nb_enfant : integer ; );
End;{ fin du case et du record }

```

مثال آخر :

الجدول السنوي للشهور الشمسية calendrier يتبع قواعد لحساب عدد الأيام للشهور حيث نجد شهوراً بـ: 30 يوم طولا و أخرى بـ: 31 يوم و فبراير بـ: 28 و 29 . كيف يمكن وضع بنية بيانات تتكفل بهذه الحالات .

mois_long(1..31)			annee	Mois
Mois_court				
Jac	True	An_bis		
<input type="checkbox"/>				
Jtc	False	An_bis		
<input type="checkbox"/>				

هنا عندنا تسجيل متغير حسب الشهور و هي ثلاث حالات الأشهر الطويلة والتي تصل إلى 31 يوم و الشهور القصيرة و التي تصل إلى 30 يوم و شهر فبراير الذي يتغير بدوره حسب قاعدة الأربع سنوات année bissextile إن كانت true يكون الشهر من 29 يوم وإن كانت false يكون من 28 يوم و يصبح التصريح كما يلي :

```

Type mois = ( jv,fr,ms,av,ma,jn,jl,ao,st,oc,no,de,);
Date = record
  Mm : mois;
  Aa : 1990..2200;
  Case mois of
    Jv,ms,ma,jl,ao,oc,de : ( mois_long : 1..31);
    Av,jn,st,no : ( mois_court : 1..30);
    Fv : ( case an_bis : boolean of
      True : ( jac : 1..29);
      False : ( jtc : 1..28); );

```

End;{ fin des case..of et du Record }

والقاعدة هي :

Record

....

case type:T ou def.variable V : T **of**

val1 : (.....);

val2 : (.....);

.....;

val_n : (.....);

end;

القيم val_n تأخذ من النوع T لتحديد الحالات .

الحقل المتغير يجب أن يكون في آخر التصريح و بالتالي لا نحتاج إلا إلى end واحد ينهي

الـ: case و الـ: record .

كل أسماء الحقول يجب أن تكون مختلفة .

و كل تسجيل لا يحمل إلا حقل واحد متغير و لكن يسمح بإدراج حقل آخر متغير ضمن

الحقل المتغير لتصبح الحقول المتغيرة متداخلة فيما بينها .

يمكن إستعمال الحالة الفارغة : () ; valj: .

يمكن إستعمال حقل ثابت كموجه sélecteur للـ: case و قيمه تحدد الحالات التي

نبينها مثل : case c:char of أين c حقل موجه للحالات . و لكن يمكن إستعمال نوع

معين كموجه مثل : case boolean of أين boolean نوع فقط و ليس بحقل ثابت للتوجيه

و إنما قيم النوع هي التي نستعملها لتحديد الحالات و المثال السابق يبين كلتا

الطريقتين .

كلمة تحذير : التسجيلات المتغيرة لا تحترم قاعدة تلاؤم الأنواع compatibilité de type

عند تعيين المتغيرات إذ يمكن نقل تسجيل لآخر بقيم مختلفة نوعا ما.

مثال :

Type Not_good = record

```

Case boolean of
  True : ( car : char );
  False : ( x : real );
End;
Var R1, R2 : Not_good ;
Begin
  R1.Car := 'Z';
  R2.X := 1.2345;
  R1 := R2 ;

```

و هذا يبين أن R2 تنقل إلى R1 مع أن القيم مختلفة: تعيين عدد حقيقي إلى حقل يبدو أنه من نوع حركات .

9. قائمة التسجيلات .

9. Tableau d'enregistrements / Array of record

مثال: في الحالة المدنية تعرض قائمة أطفال العائلة . كيف يمكن لنا تسجيل هذه البيانات

: كل طفل يعرف بإسمه و تاريخ و مكان الإزدياد ؟

تكون بنية قائمة الأطفال كما يلي : مثلاً قائمة من 3 تسجيلات :

Prenom	Date_naissnce			Lieu_naissance
محمد	jour	Mois	An	العاصمة
	14	12	1999	
Prenom	Date_naissnce			Lieu_naissance
عائشة	jour	Mois	An	جيجل
	25	05	2000	
Prenom	Date_naissnce			Lieu_naissance
الطاهر	jour	Mois	An	وهران
	19	06	2001	

و التصريح بالقائمة يكون كما يلي :

```

Type Etat_civil = record
  Prenom : string ( 30);
  Date_naissance : record

```

```

    Jour : 1..31;
    Mois : 1..12 ;
    An    : 1990..2200;
    End;
    Lieu_naissance : string ( 25 );
    End;
    Liste_enfants = Array [ 1..10] of Etat_civil ;
Var Lenf    : Liste_enfants ;

```

قراءة البيانات في قائمة التسجيلات.

```

Procedure lit_liste_enfants ( var Lef : liste_enfant );
    Var indice : 1..10 ;
    Begin
        For indice := 1 to 10 do
            With Lef [ indice ] , Date_naissance Do
                Begin
                    Readln( prenom ) ;
                    Readln( jour , mois , an ) ;
                    Readln( lieu_naissance ) ;
                End ; { with }
            End ; { lit_liste_enfant }
    End ;

```

النفاذ إلى أي عنصر يكون بنفس الطريقة و القواعد المتبعة في معالجة القائمة .

مثال عرض محتوى التسجيل رقم 3 :

```

With Lef [ 3 ] , date_naissance do
    Begin
        Write( prenom:8);
        Write ( jour:5,mois:5,an:5 );
        Writeln( lieu_naissance :10);
    End;

```

10. قائمة كحقل بتسجيل .

10. Tableau champs dans un enregistrement

من أهم ميزات بنية لغة باسكال structure du langage Pascal توفير إمكانية إدخال

قائمة array كحقل بالتسجيل و هو ما لا توفره لغة مهمة كلغة معالجة قواعد البيانات مثل

. Dbase

مثال : الحالة المدنية للعائلة :

إسم الأب و تاريخ و مكان ازدياده , إسم الزوجة و تاريخ و مكان الأزدیاد ثم قائمة الأطفال , كل طفل بلقبه و تاريخ و مكان إزدیاده .

Nom_pr	Dat_l_ns	Nom_pr_Ep	Dat_l_Nss	Liste_enf	
.....	Pren	Dt_L_n
			

Type Date_lieu_naissance = record

Jour : 1..31;

Mois : 1..12;

An : 1900..2200;

Lieu : string (25);

End;

Etat_civil = **Record**

Nom_prenom_pere : string(30);

Date_L_naiss_pere :Date_lieu_naissance;

Nom_prenom_mere : string(30);

Date_L_naiss_mere : Date_lieu_naissance;

Liste_enfant: **array[1..10]of record**

Prenom : string (30);

Date_L_Naiss_enf: date_lieu_naissance;

End;{ fin array of record }

End;{ fin record etat_civil }

Var fiche_familiale : Etat_civil ;

إجراء القراءة :

Procedure lit (var ff : Etat_civil);

Var indice : 1..10;

Begin

With ff , date_L_naiss_pere,date_L_naiss_mere

Do

Begin

Readln(nom_prenom_pere);

Readln(jour,mois,an); readln(Lieu);

Readln(nom_prenom_mere);


```

      Readln( jour,mois,an ); readln( Lieu );
      For indice := 1 To 10 do
        With liste_enfant[indice] , date_L_naiss_enf           Do
          Begin
            Readln( Prenom );
            Readln( jour, mois ,an ); readln ( lieu ) ;
            End;{ 2 ° with }
          End;{ 1 ° with }
        End;{ lit}

```

11.المجموعات داخل التسجيل.

11.Ensembles dans un enregistrement / Sets in Record

مثال :

```

Type couleur = ( rouge , bleu , vert );
Jeu_couleur = Record
  dessin: couleur ;
  Ens   : Set of couleur ;
End;
Table_couleur = Array [1..100] of Jeu_couleur ;
Var      JC : Jeu_couleur ;
      TC   : Table_couleur;  i : integer ;

```

تعيين القيم :

```

JC.dessin := Rouge ;
JC.Ens := [ rouge, bleu];
For  i := 1 To 100 do
  Begin
    TC[i].dessin := vert ;TC[i].Ens := [ rouge,bleu];
  End;

```

12.الوسيط من نوع التسجيل في الإجراء و الدالة.

12.L'enregistrement comme paramètre dans une fonction et procédure .

التصريح التالي في التسجيل غير مسموح به :

```

Procedure Lit ( var E : record

```

```
Nom : string( 20);
Adresse : string( 50);
End; );
```

هنا صرحت بوسيط شكلي من نوع جديد و هو يتناقض مع قواعد باسكال أكان في إجراء أو دالة .

13.تمارين .

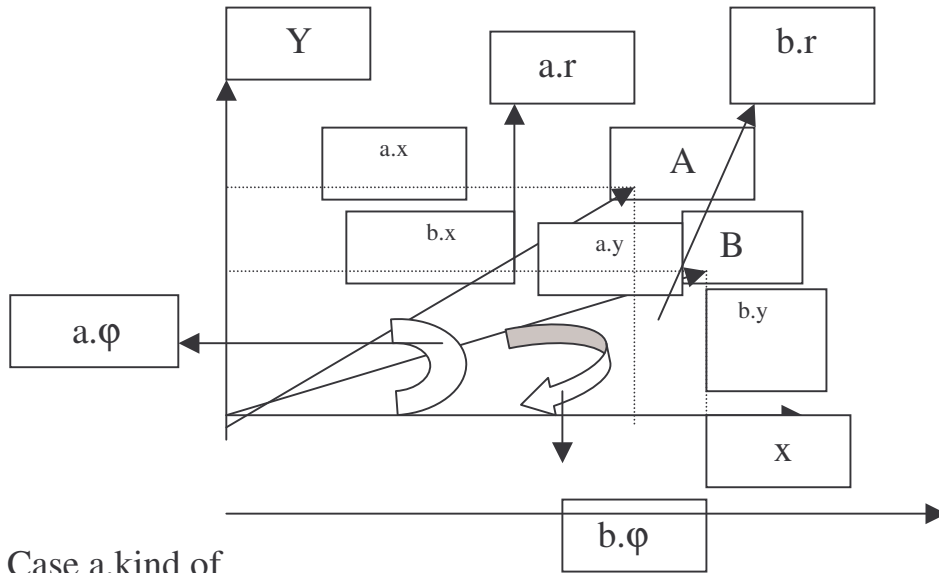
13.Exercices

1. أنظر إلى التصريح التالي و اذكر هل هو صحيح :

```
Var a : integer;
    B : record
        A : integer ;
        B : char ;
    End;
```

2. أكتب التصريحات الكاملة و أتمم الإجراء الذي يحسب المسافة بين A و B بناء على

معرفة قيمة المتغيرات a و b :



Case a.kind of

Cartesien : case b.kind of

Cartesien : $d := \text{sqrt} (\text{sqr} (a.x - b.x) + \text{sqr} (a.y - b.y));$

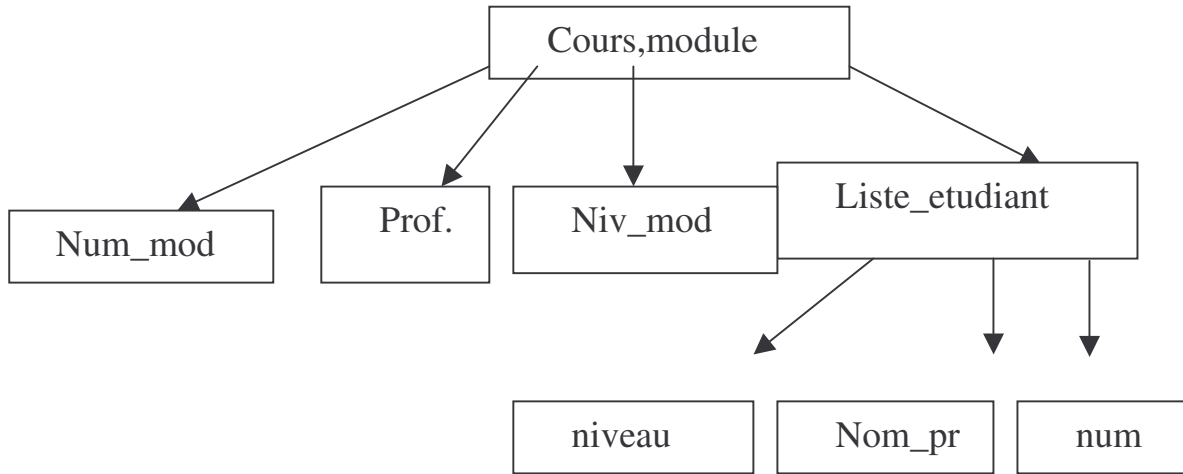
Polar : $d := \text{sqrt} (\text{sqr} (a.x - b.r * \cos(\theta)) + \text{sqr} (a.y - b.r * \sin(\theta)));$

```

cos (b.φ)+sqr(a.y-      b.r * sin(b.φ));
End;
Polar : case b.kind of
  Cartesien : d := sqrt(sqr(a.r * cos (a.φ)-b.x
+ sqr(a.r * sin(a.φ) - b.y ));
  Polar : d := sqrt ( sqr(a.r)+sqr(b.r)-
2*a.r*b.r* cos (a.φ - b.φ));
End;

```

3. طلبت منك جامعة أن تكتب لها برنامجا يدير تسجيل الطلبة . علما أن الطلبة يسجلون أنفسهم في المقاييس modules حسب مستوياهم . و كل مقياس يعرف حسب المعطيات التالية :



حقل niv_mod يعبر عن السنة وهي من 1..5 و num_mod يعبر عن رقم المقياس من 1 ..100 . عدد الطلبة لا يتعدى 50 في المقياس .

1. أكتب إجراء قراءة المقاييس إذا كان عددها لا يتعدى 200 .

2. أكتب إجراء عرض مقياس معين .

3. أكتب إجراء عرض قائمة الطلبة المسجلين في كل المقاييس حسب معرفة المستوى : 1 .. 5.

4. أكتب إجراء عرض أرقام المقاييس من مستوى 4 و قد سجل فيها الطلبة من مستوى 1 و 2 .

5. أكتب الإجراء الذي يقدم مجموع الخدمات السابقة حسب إختيار المستعمل.
الباب الثالث عشر : الجذاذات .

Chapitre 13:Les fichiers / Files .

1. الهدف.

1.But.

إذا كنا نكتب برامج لحل مسائل و إشكالات عدة فإننا نهدف إلى إستعمال هذه البرامج ببياناتها و نتائجها عدة مرات بدون اللجوء إلى إدخال المعطيات في كل مرة نريد معرفة النتائج كما نود إضافة بيانات جديدة و حذف أخرى و الاحتفاظ بالنتائج للاستعمالات المقبلة و هذا الذي لحد الآن لم نلتطرق إليه , و هو الهدف من إستعمال الحاسب ككل : الاحتفاظ بالبرامج و البيانات و النفاذ إليها و توفيرها بسرعة عند الحاجة . كل هذه الخصائص و المهام يجب أن نتمكن منها بلغة البرمجة حيث تمكنا من تخزين بيانات البرامج و نتائج التنفيذ بدلا من إدخالها دائما باستعمال لوحة المفاتيح و قراءة النتائج على الشاشة .

فالهدف من كتابة البرامج هو توفير حلول و خدمات للمستعمل و هذا بتخزين البرامج و بياناتها و نتائجها و هو ما نتعلمه باستعمال بنية الجذاذة fichier / file و كما تعلم وسائل التخزين الحالية : الشريط المغناطيسي bande magnétique أو القرص المغناطيسي disque magnétique و القرص CD-rom و الأشرطة الخاصة . و أما الطباعة فنعتبرها وسيلة خرج ضرورية للاحتفاظ بالنتائج و تخزينها في الأرشيف و تقديم التقارير و غير ذلك..مكتوبة على الورق.

عندما تكتب برنامجك في جذاذة نصية fichier texte و المسماة بالجذاذة المنبع fichier source و عندما تترجمها بالترجمان compilateur يحدث لك جذاذة ثانية تسمى الجذاذة المنفذة EXE. fichier exécutable و هو ما يبين لك أن الجذاذات نوعين :

النصية Fichier texte و الثنائية Fichier binaire .

*ملاحظة: لماذا نستعمل المصطلح الجذاذة لترجمة الكلمة الإنجليزية File و الفرنسية

Fichier لسبب بسيط أولا تعريف و بنية الـ: File في علوم الحاسبات Computer science أو Informatique هي : مجموعة من التسجيلات . و هو ما يتناسب مع معنى الجذاذة أو الجزاة و هي كلمة عربية أصيلة وردت في القرآن الكريم . ثانيا لو فرضنا أن ترجمة الكلمة File / Fichier هي الملف , فكيف نسمي : Folder / document و التي حقيقة هي الملف حيث تجمع فيه الأوراق والمعلومات . و هكذا نسهل على أنفسنا ترجمة المصطلحات الجديدة في عالم الحاسبات . و عدم اللجوء إلى كلمات بعيدة عن المعنى الإنجليزي و التي تدخل إلتباس على المتعلم و المستعمل العربي . مثل كلمة مستندات و التي يقصد بها File مرة و أخرى folder و غيرها من المصطلحات المترجلة.

2.تعريف الجذاذة .

2.Définition du Fichier / Files

تعرف الجذاذة بأنها مجموعة من التسجيلات تخزن على وسيلة تخزين مثل القرص و الأشرطة المغناطيسية .

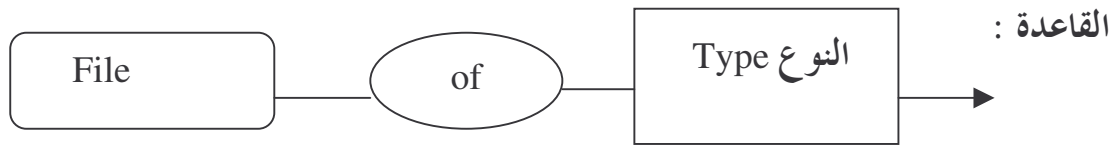
3.التصريح بالجذاذة .

3.Déclaration de fichier .

التسجيل في مفهوم علوم الحاسبات هو كل عنصر معلوماتي متكون من صنف أو أكثر من البيانات . فالتسجيل قد يكون بسيطا مثل :

integer,real,boolean,char,declaré. أو بنيوي : set,array,record .

و الجذاذة مجموعة من هذه التسجيلات يتم تخزينها كوحدة مستقلة و تتم عليها جميع العمليات كالكتابة و القراءة و الحذف و الإضافة .



النوع هنا هو نوع التسجيلات و يكون بسيط أو بنوي .
 مبدئيا لا يوجد أي اعتراض على نوع file نفسه مثل : file of file و لكن الترجمات
 الحالية لا تقبلها و قد تجد ترجمان يسمح بذلك .
 *أمثلة :

```
Type Fichier_entier = File of integer ;
Fichier_reel = file of Real ;
Fichier_car = File of Char ;
Fichier_logique = file of boolean ;
Liste_entier = Array[1..10] of Integer ;
File_tab_entier = File of Liste_entier;
Matrice = Array[1..10,1..10] of Char ;
Fichier_matrice = File of Matrice ;
Personne = Record
  Nom_prenom : String( 25);
  Date_naiss : integer;
End;
Fichier_personne = File of Personne ;
```

*ملاحظة :

1. من تعريف الجذاذة يتبين لنا أن عدد التسجيلات غير محدد إلا ما يفرضه عليك
 نظام التشغيل / système d'exploitation / operating system من حد أقصى للتخزين على
 القرص أو الشريط .

2. تعريف BPW7 للجذاذات يتبع نفس القواعد السابقة مع إضافة بسيطة :

A file type consists of a linear sequence of components of the component
 type , which can be any type except a file type .

Syntax : file of type
 OR
 file

If the word **of** and the component type are omitted, the type denotes an untyped file . الجذاذة بدون نوع .

The predefined file type **Text** signifies a file containing characters organized into lines .

type swapFile = file ;

هذه هي الإضافة التي يسمح بها 7 BPW الجذاذة بدون نوع .

4.أنواع الجذاذات .

4.Type de fichier.

أنواع الجذاذات إثنين : النصية texte و الثنائية binaire .

الجذاذة النصية أي أنها من تسجيلات من نوع الحركات مثل التي نحررها بمحرر نصوص éditeur de texte أو محرر صفحات مثل word و هي جذاذات يمكن مشاهدة محتواها و قراءتها و طباعتها مباشرة لأنها من حركات Ascii و كل الحركات المكتوبة مثل العربية . يوفر باسكال هذا النوع و نستطيع إستعماله بالتصريح بالجذاذة كما يلي :

Type livre = Text ;

لأن النوع Text مصرح به ;: text = file of char

أما النوع الثاني : الجذاذة الثنائية fichier binaire فهي التي تكون من الأنواع الأخرى غير char مثل التي عرفنا في الفقرة السابقة : لا يمكن قراءة محتواها بمحرر نصوص أو محرر صفحات فهي مخزنة على شكل كلمات ثنائية mot binaire / binary words .

5.الجذاذة المنطقية و الفيزيائية .

5.Fichier logique et physique.

عندما نُحدث جذاذة في قرص أو شريط فإنها بالفعل جذاذة فيزيائية fichier

physique أي حقيقية , قد أنشأت و للتأكد من وجودها نطلب فهرس القرص في نظام

MsDos بتعليمة Dir أو في نظام Windows نبحث عنها بـ: Search for Files and

folders في الباب : Start تجدها و يعرض عليك محتواها بالحركات : Octets / Bytes .

أما عندما نصرح بجذاذة في البرنامج فإن المترجمان compilateur يحدث لنا في الذاكرة المركزية جذاذة منطقية fichier logique.

و ما هي إلا عنصر واحد مكون للجذاذة أي تسجيل واحد .
 إذا سعة الجذاذة المنطقية في الذاكرة المركزية تساوي سعة التسجيل
 الذي منه تكون الجذاذة و يسمى buffer أو حازم مؤقت يخزن فيه تسجيل واحد الذي
 تم إنشائه أو جلبه إلى الذاكرة من القرص.

مثال: عندما نصرح بجذاذة يحضر لنا متغير حازم variable buffer مثل F و FP سعتها

تسجيل واحد	
F	105
FP	ali Kadour 12 05 1999

Type Fichier = File of Integer ;
 Date = record
 Jour : 1..31;
 Mois : 1..12;
 An : 1909..2200;
 End;
 Personne = Record
 Nom,prenom : string(25);
 Dat_Nais: date;
 End;
 Fichier_Per = File of Personne ;
 Var F : Fichier;
 FP : Fichier_Per ;

*ملاحظة : الجذاذة المنطقية أو الحازم أو الـ Buffer كلها كلمات تعني نفس الشيء .

6.تنظيم الجذاذات .

6.Organisation des fichiers.

عندما نحدث جذاذة ما على وسط مثل القرص أو الشريط نعين طريقة النفاذ إلى
 عناصرها فإما نستعمل الأسلوب التتابعي Séquentiel أو المباشر directe أو التتابعي
 المفهرس séquentiel indexe .

يوفر BPW7 ثلاثة طرق للنفاذ : النفاذ بالقراءة فقط , النفاذ بالكتابة فقط , النفاذ بالقراءة

والكتابة في نفس الوقت بإستعمال المتغير filemode كالتالي :

FileMode (variable) (System unit)

The FileMode variable determines the access code to pass to DOS when typed and untyped files (not text files) are opened using the Reset procedure .

The default FileMode is 2

أي أن التنظيم التلقائي هو المباشر : القراءة و الكتابة .

all these modes are defined :

- 0 للقراءة فقط filemode := 0
- 1 للكتابة فقط filemode := 1
- 2 للقراءة و الكتابة معا filemode := 2

1.6 التنظيم التتابعي .

6.1 L'organisation Séquentiel .

عندما نصرح بجذاذة في لغة باسكال فإننا نعلن عن تنظيمها تلقائيا par défaut بالتتابعي لأن باسكال يوفر التنظيم التتابعي على المستوى القياسي تلقائيا .

و لكن BPascal 7 يوفر التنظيم المباشر تلقائيا.

لإختيار التنظيم التتابعي في BPw7 نعين filemode := 0 أي للقراءة فقط أو filemode 1 := للكتابة فقط .

و كما يدل اسمه عليه فإن التنظيم التتابعي أو التسلسلي لا يسمح بالقفز إلى أي تسجيل إلا بعد المرور على التسجيلات السابقة له من بداية الجذاذة . و الشرط المغناطيسي أحسن مثال على هذا التنظيم حيث أن الشرط لا يسمح إلا بالتنظيم التتابعي . في هذا التنظيم لا يمكن فتح الجذاذة إلا للقراءة أو الكتابة في نفس الوقت .

2.6 التنظيم المباشر .

6.2 L'organisation Directe.

لتنفيذ النفاذ المباشر يجب توفير وسيلة تخزين تسمح بذلك و هي القرص لما له من خصائص حيث أنه مشكّل formaté من مسارات pistes و قطع secteurs وله رؤوس قراءة و كتابة têtes de lecture écriture متحركة , و هو الذي يسمح بالنفاذ المباشر إلى أي كتلة bloc في أي مسار .

3.6 التنظيم التتابعي المفهرس .

6.3 Le séquentiel indexé.

هذا النوع من النفاذ يستعمل جذادة فهرس fichier indexé أو فهرس على شكل قائمة خطية حركية في الذاكرة المركزية (pointeur) liste linéaire dynamique للنفاذ المباشر إلى التسجيلات المخزنة بالضرورة على قرص .

7. الإجراءات و الدوال الموفرة لمعالجة الجذادات.

7. Procédures et fonctions disponibles pour les fichiers.

سنعرض الدوال و الإجراءات القياسية Standard و بعض الدوال و الإجراءات الإضافية في ترجمان BPW7 . و في الأخير تجد القائمة الكاملة يمكن أن ترجع إلى الشروح الموفرة في المساعدة الآنية help عند إستعمالك للترجمان تحت نظام Windows .

1.7 الربط بين الجذادة المنطقية و الفيزيائية.

7.1 Liaison fichier logique - physique :

Assign(..);

لربط بين الجذادة المنطقية و الفيزيائية يوفر باسكال إجراء Assign

و الذي يعني إربط :

Assign (var F ; consts N : string);

مثلا : assign(fichier, 'letter.doc'); ربطنا الجذاذة المنطقية fichier بإسم الجذاذة الفيزيائية letter.doc الموجودة على وسيلة تخزين أو التي ستحدث عليها و غالبا ما تكون القرص بجميع أشكالها .

يعرف BPw7 هذا الإجراء كما يلي :

Assign :Assigns the name of an external file to a file variable .

Declaration: procedure Assign(var f; String);

A file name consists of a path of zero or more directory names separated by backslashes, followed by the actual file name :

Drive:\DirName\...\DirName\FileName

2.7 إجراء فتح الجذاذة للكتابة فيها .

7.2 Procédure d'ouverture pour écrire : Rewrite

Procedure Rewrite (var F);

هذا الإجراء يغلق الجذاذة f ثم يفتحها إن كانت موجودة و إذا كانت غير موجودة

يحدثها . مع الملاحظة إن كانت موجودة كل محتواها يهمل و يمحي. الدالة eof(f)

الخاصة بمراقبة نهاية الجذاذة تصبح true و الذي هو ضروري و واجب قبل الكتابة فيها .

في BPW7 نجد نفس الإجراء مع إضافة وسيط recsize طول التسجيل في حالة

الجذاذة بدون نوع untyped file .

procedure Rewrite(var F: File [; Recsize: Word];

Remarks

F is a variable of any file type associated with an external file using Assign. RecSize is an optional expression of type Word, which can be specified only if F is an untyped file. If F is an untyped file, RecSize specifies the record size to be used in data transfers. If RecSize is omitted, a default record size of 128 bytes is assumed .

If F was assigned an empty name, such as Assign(F,"), then after the call to Rewrite, F refers to the standard output file (standard handle number 1);

يستعمل BPW المصطلح : للكتابة فقط Write-only للتعبير عن النظام التتابعي . فيصرح بـ: إذا كانت F من نوع نص text تصبح تلقائياً بنفاذ تتابعي.

If F is a text file, F becomes write-only. After a call to Rewrite, Eof(F) is always True .

3.7 إجراء فتح الجذاذة للقراءة.

7.3 Procédure Reset(var f);

procedure reset (var f);

تفتح الجذاذة f ثم تقوم بوضع أول تسجيل أي الجذاذة المنطقية في الذاكرة و تصبح تشير إلى أول تسجيل في الجذاذة الفيزيائية هذا إذا كانت الجذاذة غير فارغة و الدالة eof(f) تصبح false .

أما إذا كانت f فارغة : f تصبح غير معلومة القيمة و eof (f) = true

Reset (f) تفتح f للقراءة فقط في النظام التتابعي . و لكن في النظام المباشر

تسمح بالكتابة كذلك .

و في BPW7 نجد تقريبا نفس الإجراء مع إضافة reresize طول التسجيل في حالة الجذاذة بدون نوع وهي إضافة اختيارية .

procedure Reset(var F [: File; Reresize: Word]);

Reset opens the existing external file with the name assigned to F . An error results if no existing external file of the given name exists. If F is already open, it is first closed and then reopened. The current file position is set to the beginning of the file .

If F is assigned an empty name, such as Assign(F, ""), then after the call to Reset, F refers to the standard input file (standard handle number 0)

إذا كانت الجذاذة من نوع نص Text تصبح للقراءة فقط و هو النظام التتابعي في

.BPW 7

If F is a text file, F becomes read-only.

4.7 الدالة : لمراقبة نهاية الجذاذة.

7.4 Function Eof (var f) : boolean;

تراقب نهاية الجذاذة و ترجع true إذا كانت f^ في نهاية الجذاذة في النظام التتابعي:
فإذا أن الجذاذة في حالة كتابة أو أنها وصلت إلى قراءة آخر تسجيل . في حالة النفاذ المباشر
eof (f) = true إذا كانت آخر عملية: write أو آخر عملية كانت قراءة .

5.7 الدالة : نهاية السطر في الجذاذة.

7.5 Function eoln(var f):boolean ;

ترجع true إذا كانت f في نهاية السطر في الجذاذة f من نوع text أي من حركات
ascii . إذا كانت eoln (f) = true قيمة f = ' ' فراغ و لكن الجذاذة متوقفة في علامة نهاية
السطر line marker .

6.7 الإجراء : get .

7.6 Procédure get (var F);

إجراء قراءة الجذاذة عنصرا بعنصر .
إجراء قياسي standard خاص بالجذاذات a primitive file system intrinsic
procedure نجدها في Ms-Pascal و للأسف لا يوجد بـ: **Borland- Pascal** .
ولكن يمكن إحداثها فيه . تعمل كالتالي :
1. تتقدم بخطوة في الجذاذة الفيزيائية للعنصر الموالي .
2. تعين قيمته إلى المتغير الحاجر f^ buffer variable .
3. تصبح eof (f) = false .
وهي إذا عمليتان تقدم ثم عين advance and assign .

إذا لم تجد أي عنصر eof(f) تصبح true و قيمة f^ تكون مهمة غير معروفة undefined لذلك يجب أن تكون eof(f) = false قبل كل get(f) . أما إذا كانت f منظمة بالنفاذ المباشر mode direct فإن eof(f) لا تهم قيمتها أكانت false أو true بما أن هذا النفاذ يسمح بتكرار get في نهاية الجذاذة .

إذا كانت f^ تسجيل بـ: case أي تسجيل متغير variant record فإن الترجمان يقرأ التسجيل بأكبر حالة له reads the variant with the maximun size .

7.7 إجراء الكتابة في الجذاذة : put .

7.7 Procedure Put (var F);

إجراء الكتابة في الجذاذة : تكتب قيمة المتغير الحاجر f^ في الجذاذة في المكان الحالي the current position الذي تشير إليه ثم تتقدم بخطوة أي إلى العنصر الموالي : هذا الإجراء قياسي , كذلك لا يوفره Turbo/BPascal ولكن يمكن إحداثها كإجراء جديد . في التنظيم التتابعي put ممكنة بعد rewrite أو put آخر أو write في f . و لكنها غير ممكنة بعد reset و get أو read . و في النظام المباشر put ممكنة بعد reset أو get . يجب أن تكون eof(f) = true قبل كل put إلا في النفاذ المباشر . و قيمتها بعد put تكون دائما true . إذا كانت F^ تسجيل متغير فإن put تكتب أكبر التسجيل بأطول حالة .

8.7 الإجراءات الخاصة بالقراءة : read , readln .

7.8 Procedures Read et readln .

read إجراء قراءة من جذاذة من نوع نص text وثنائي binary . أما readln فلا تستعمل إلا في النوع نص text . إذا كانت f من نوع text و p من نوع char فإن read(f, p) ; تشبه التعليمات التالية

و تعني أنقل قيمة المتغير المنطقي Fichier logique إلى المتغير p ثم تتقدم إلى التسجيل الموالى في الجذاذة الفيزيائية .

في Turbo/BPascal نجد نفس الإجراء مع بعض الإضافات :

Read (procedure)

- For typed files, reads a file component into a variable .
- For text files, reads one or more values into one or more variables

Declaration

Typed files: procedure Read(F,V);

Text files: procedure Read([var F: Text;] V1 [, V2,...,Vn]);

9.7 الإجراءات الخاصة بالكتابة . Write , writeln

7.9 Les procedures Write , writeln .

write إجراء يكتب في الجذاذة النصية و الثنائية . أما writeln فلا تكتب إلا في الجذاذة من نوع النص .

كما يمكن إستعمال أكثر من وسيط : write (F,P1,P2,...P_n);

أما writeln فإنها تكتب علامة نهاية السطر في الجذاذة .

writeln(f,P1,..P_n) تشبه التعليمات التالية :

begin

write (F, P1,P2,P_n);

writeln(F);

end;

في BPW7 نجد نفس الإجراء مع بعض الإضافات :

Write (procedure)

For typed files, writes a variable into a file component. For text files, writes one or more values to the file

Declaration:

Typed files: procedure Write(F, V1 [, V2,...,Vn])

Text files: procedure Write([var F: Text;] P1

[,P2,...,Pn])

10.7 إجراء غلق الجذاذة.

7.10 Procedure Close (var F);

تقوم بغلق الجذاذة منطقيا و فيزيائيا. و إذا كانت الجذاذة نص و كان آخر سطر لا يحتوي على علامة eofn تقوم close بوضع هذه العلامة في آخر السطر. و إذا كانت الجذاذة بنفذ تتابعي يوضع بآخرها علامة نهاية الجذاذة eof كما أن close على جذاذة مغلقة لم تفتح لا بـ: reset أو rewrite ممكن .

11.7 إجراء النفاذ المباشر إلى أي تسجيل .

7.11 Procedure Seek (var F ; N : integer);

هذا الإجراء ينفذ إلى التسجيل رقم n في الجذاذة f ليسمح لك بقراءة أو الكتابة عليه . n يجب أن يكون عددا صحيحا و f جذاذة بتنظيم مباشر. بعد كل reset و rewrite تشير seek إلى التسجيل الأول .
يوفر BPW7 نفس الإجراء مع بعض التغييرات فيه :

Seek (procedure)

Moves the current position of a file to a specified component .

Declaration : procedure Seek(var F; N: Longint);

F is any file variable type except text, and N is an expression of type Longint. The current file position of F is moved to component number N . The number of the first component of a file is 0. To expand a file, you can seek one component beyond the last component; that is, the statement Seek(F, FileSize(F)) moves the current file position to the end of the file .

Restrictions : حدود الإستعمال Cannot be used on text files. File must be open

لا يستعمل على جذاذة نصية و يجب فتح الجذاذة مسبقا.

12.7 إجراء حذف الجذاذة فيزيائياً.

7.12 Procedure de suppression de fichier:Erase .

يوفر BPW7 إجراء لحذف أي جذاذة كيف ما كان نوعها من القرص .

Erase (procedure);

Erases an external file .

Declaration procedure Erase(var F);

Target :Windows, Real, Protected

Remarks: F is a file variable of any file type. The external file associated with F is erased .

Restrictions: حدود الإستعمال

. لا تستعمل أبدا هذا الإجراء على جذاذة مفتوحة

مفتوحة

13.7 إجراء الإضافة في الجذاذة النصية.

7.13 Procédure d'ajout dans un fichier text :Append .

يوفر BPW v.7 هذا الإجراء للإضافة

Append (procedure) لإضافة تسجيل في الجذاذة من نوع نص Text

Opens an existing file for appending .

Declaration :procedure Append(var f: Text) ;

where : f is a text-file variable .

Target: Windows, Real, Protected

Remarks : F is a text file variable that must have been associated with an external file using Assign .

تفتح الجذاذة المعينة تحت الاسم المقدم في Assign وإلا تعلن عن خطأ إذا لم تجدها . إذا

كانت الجذاذة مفتوحة تغلق ثم تفتح من جديد , و تشير إلى آخر الجذاذة .
بعد النداء لـ: Append تصبح F للكتابة فقط .

After a call to Append, F becomes write-only, and the file pointer is at the end of the file .

مثال لإستعمال Append :

```
uses WinCrt ;
var F: Text ;
begin
  Assign(F, 'TEST.TXT');
  Rewrite(F);
  Writeln(F, 'original text ');
  Close(F);           { Close file, save changes }
  Append(F);          { Add more text onto end }
  Writeln(F, 'appended text ');
  Close(F);           { Close file, save changes }
End
```

14.7 دالة ترجع لك أين هي نافذة الجذاذة .

7.14 Fonction :Position dans le fichier:filePos

هذه الدالة ترجع لك أين توجد نافذة الجذاذة: التي تشير إلى المكان في الجذاذة الفيزيائية لقراءة التسجيل أو الكتابة. ترجع صفر إن كانت في البداية أو ترجع القيمة FileSize(f) إن كانت في نهاية الجذاذة .

Returns the current file position of a file FilePos (function):

Declaration : function FilePos(var F): Longint;

Target : Windows, Real, Protected

Remarks : F is a file variable. If the current file position is at the beginning of the file, FilePos(F) returns 0. If the current file position is at the end of the file--that is, if Eof(F) is True--FilePos(F) is equal to FileSize(F)

Restrictions : حدود الإستعمال

Cannot be used on a text file. File must be open .

لا يمكن إستعمالها على جذاذة نص . و يجب فتح الجذاذة من قبل .

15.7 دالة ترجع عدد التسجيلات في الجذاذة .

7.15 Fonction : Taille du fichier:fileSize.

ترجع عدد عناصر الجذاذة . إن كانت فارغة ترجع صفر .

FileSize (function) : Returns the current size of a file .

Declaration : function FileSize(var F): Longint ;

Target : Windows, Real, Protected

Remarks : F is a file variable. FileSize(F) returns the number of components in F . If the file is empty, FileSize(F) returns 0 .

Restrictions : حدود الإستعمال

لا تعمل على جذاذة من نوع نص . و يجب فتح الجذاذة من قبل .

Cannot be used on a text file. File must be open .

16.7 دالة ترجع ما قيمة نهاية الجذاذة .

7.16 Fonction : donne la valeur de fin de fichier:seekEof.

SeekEof (function) : Returns the end-of-file status of a file .

Declaration :

function SeekEof [(var F: Text)]: Boolean ;

Target : Windows, Real, Protected

Remarks : لا تستعمل إلا على الجذاذات النصية. و يجب فتح الجذاذة :

Can only be used on text files. File must be open .

17.7 دالة ترجع قيمة نهاية السطر.

7.17 Fonction : donne la valeur de fin de ligne: SeekEoln.

SeekEoln (function);

Returns the end-of-line status of a file .

Declaration :

function SeekEoln [(var F: Text)]: Boolean ;

Remarks : لا تستعمل إلا على الجذاذة النص .

Can only be used on text files. File must be open .

** الخلاصة : تعرضنا للدوال و الإجراءات القياسية standard و إضافات Borland

Pascal for Windows V.7 الخاصة بالجذاذات إلا القليل منها لم نتعرض له تجد القائمة كاملة

في نهاية الكتاب و هي موجودة في المساعدة HELP التي يوفرها المترجم BPW v.7 .

8. الجذاذات النص.

8.Les fichiers Textes./ Text files.

1.8 التعريف .

8.1 Definition.

الجذاذة النص مصرح بها في باسكال كالتالي :

Type Text = file of Char ;

و هي بنية حركات ascii و التسجيل هو السطر في الأنظمة التي تقسم الجذاذات إلى أسطر

, كل سطر ينتهي بعلامة نهاية السطر line marker .

كما يمكن إضافة طول السطر في التصريح بالنوع text في ترجمان MsPascal و لكن

BPW لا يسمح بذلك و لا يسمح باستعمال الإجراء seek في الجذاذات النصية Text .

مثلا:

```
Type fichier_text = Text ;
Page = text ;
```

2.8 إحداث الجذادة .

8.2 Procédure de Création de fichier .

الخوارزمية :

1. فتح الجذادة للكتابة بـ: rewrite .
2. قراءة تسجيل من الدخل : هنا حركة واحدة أو سلسلة حركات string.
3. كتابة التسجيل في الجذادة.
4. إعادة الخطوات 2 و 3 إلى أن ينتهي الدخل.

3.8 نقل جذادة إلى أخرى.

8.3 Copie de Fichiers.

الخطوات:

1. فتح الجذادة المنبع source للقراءة و الجذادة الهدف destination للكتابة.
2. قراءة تسجيل من الجذادة المنبع .
3. كتابة التسجيل في الجذادة الهدف.
4. تكرار الخطوات 2 و 3 إلى نهاية الجذادة المنبع .

4.8 طبع الجذادة .

8.4 Imprimer un Fichier .

! الخطوات :

1. فتح الجذادة للقراءة.
2. فتح الطابعة للكتابة .

3. قراءة تسجيل :سطر من 80 حركة .

4. كتابة التسجيل على الطابعة .

5. تكرار الخطوات 3 و 4 إلى نهاية الجذاذة المنبع .

6. غلق الجذاذات في النهاية .

بنفس الطريقة يمكن عرض الجذاذة على الشاشة و لكن بدون حاجة إلى التصريح بجذاذة الخرج output .

5.8 إضافة تسجيل في النظام التتابعي .

8.5 Ajout en mode séquentiel .

لإضافة تسجيل في جذاذة نصية text تحت النفاذ التتابعي أو كما يسميه BPW7 :
حالة القراءة فقط read only أو الكتابة فقط write only نضطر إلى إستعمال جذاذة ثانية للقيام بهذه العملية حيث أن فتح الجذاذة بـ: rewrite للكتابة فيها يسمح محتواها و بالتالي نحتاج إلى جذاذة ثانية تفتح للكتابة و هي إذا فارغة و فتح الجذاذة التي ننقل محتواها إلى الثانية للقراءة فقط .

*الخطوات :

1. فتح الجذاذة الأولى للقراءة و الثانية للكتابة .

2. نقل محتوى الأولى للثانية .

3. قراءة التسجيل من الدخل و كتابته في الثانية .

4. كرر الخطوة 3 إلى نهاية الإضافات .

5. غلق الجذاذات .

5.8 إستحداث الجذاذة في النظام التتابعي .

8.5 Mise a jour en mode séquentiel .

كما ذكرنا فإن النفاذ التتابعي يعيق العمل إذا كان البرنامج يستعمل الأسلوب التخاطبي *conversationnel* حيث تقوم بطلب خدمات من البرنامج و تنتظر تنفيذها فوراً و بما أن النفاذ التتابعي لا يسمح بفتح الجذاذة للقراءة و الكتابة في نفس الوقت فإننا نضطر لإحداث جذاذتين لكل عملية نطلبها وهو ما يأخذ وقتاً كبيراً مثل : الإضافة *ajout* و التغيير *modification* و الحذف *suppression* و التي تمثل عمليات إستحداث الجذاذات *mise a jour*. فالتغيير عملية تبديل سطر بآخر حسب إختيار المستعمل بمعرفة رقمه داخل النص أو بمحتواه و هي عملية بحث و تحتاج إلى قراءة و كتابة : إذا جذاذتين . أما الحذف فهي عملية للبحث على السطر برتبته أو بمحتواه كذلك ثم القيام بنقل كل الجذاذة إلى أخرى ما عدى السطر المختار للحذف . فجميع عمليات الإستحداث تتطلب جذاذتين لضرورة القراءة و الكتابة و عمليات طويلة و مكلفة في الوقه و مساحة التخزين أكان قرص أو شريط لذلك لا نجد هذا النوع من الإستحداث على الجذاذة النصية إلا في النفاذ المباشر .

6.8 الإستحداث في النظام المباشر.

8.6 Mise à jour en mode directe.

مثلاً: نريد كتابة محرر نصوص بسيط. لتدرب على هذا النوع من التطبيقات و التي أصبحت واسعة و تشكل أكبر خدمة يوفرها الحاسب اليوم لأغلب الناس ألا و هي تحري النصوص و الصفحات و الجذاذات و الملفات و إستحداثها.

يوفر هذا المحرر الخدمات التالية :

1. تحرير نص و تخزينه على القرص .
2. الإضافة إلى النص سطر أو أكثر .
3. تغيير سطر أو أكثر .

4. حذف سطر أو أكثر .

5. عرض محتوى الجذاذة .

6. طبع النص صفحة صفحة.

و نستطيع إضافة خدمات أخرى و تطويرها و تحسين جودتها .

ملاحظة : بما أن BPW7 لا يوفر الإجراء seek للنفذ المباشر إلى السطر الذي نريد و لا يسمح بتحديد طول السطر فإننا لا نستطيع الآن القيام بعمليات التغيير مباشرة على الجذاذة أو الحذف و لكن نستطيع الإضافة و العرض و الطبع . و من أجل التعلم نعرض هذه الإجراءات بترجمان MsPascal الذي يسمح بالنفذ المباشر بـ: seek و بتحديد طول السطر .

فكيف يمكن لك حل إشكال الإستحداث mise a jour للجذاذات النصية ؟

هنا نرى ضرورة وجود آلية تمكنا من شحن الجذاذة إلى الذاكرة و القيام بكل العمليات في الذاكرة و هي البنية الحركية structure dynamique بنية القائمة الخطية (liste linéaire pointeur)

Π الإجراءات في MsPascal:

```
Program Editeur_text ;
  Type Texte = Text ( 80 );
  Var Tex , imp : Texte;
  Choix : char ;
```

1. لإحداث الجذاذة : قراءة الدخل ثم تخزينه على الجذاذة الفيزيائية.

```
Procedure edit ( var tex : texte );{ creation du fichier }
  Var x : string ( 80);
  Begin
    Write(' file name : '); readfn( input, tex );
    Rewrite ( tex ) ;
    Writeln ('.....Debut.....');
    While not eof ( tex ) do
      Begin
        Readln ( x ) ; writln ( tex, x );
```



```

        End;
        Writeln ( '.....fin.....');
    End;{ fin creation }

```

2. لإضافة سطر إلى الجذاذة أو أكثر . ajout d'une ligne ou plus

```

Procedure ajout ( var tex : texte );
    Var x : string ( 80);
    Begin
        Tex.mode := direct ;
        Reset ( tex );
        While not eof ( tex ) do
            Readln ( tex , x );{ للذهاب إلى نهاية الجذاذة فقط }
        Writeln ( ' commencer l'ajout et Ctrl +Z pour
        Terminer l'entree des lignes ');
        While not eof do { لقراءة التسجيلات من الدخل }
            Begin
                Readln ( x ); writeln ( tex , x );
            End;{ fin }
    End;

```

3. لتغيير سطر أو أكثر.

```

Procedure modify ( var tex : texte );
    Var x : string ( 80);
        I : integer ; rep : char ;
    Begin
        Tex.mode := direct ;
        Reset ( tex );
        Repeat
            Write(' donner le N° de ligne a modifier '); Readln( i );
            Writeln ( ' entrer la nouvelle ligne '); Readln ( x );
            Seek ( tex , i ); { acce direct a la ligne N° i }
            Writeln ( tex , x );
            Writeln ( ' تريد تغيير سطر آخر نعم أم لا '); Readln ( rep );
        Until rep In [ 'n', 'N'];
    End; { fin Modify}

```

4. حذف سطر أو أكثر .

خوارزمية الحذف تعتمد على مبدئين الحذف المنطقي و الحذف الفيزيائي.

1. الحذف المنطقي يضع علامة على السطر و يتركه في الجذاذة لا يحى منها و لكن عند عرضها أو طبعها لا تظهر التسجيلات المحذوفة .

2. الحذف الفيزيائي يعتمد على مبدأ الحذف الحقيقي أي لا تبقى التسجيلات في الجذاذة و طريقة الحذف تنقل كل التسجيلات ماعدا التي علّمت marque خلال الحذف المنطقي إلى جذاذة و نسخ مؤقتة fichier brouillon ثم إعادة نقل الجذاذة المؤقتة إلى الجذاذة المنبع وقد فقدت جميع التسجيلات المحذوفة منطقيا .

```

Procedure delete ( var tex : texte );
  Var n, i, j : integer ;
  X : string ( 80 ); rep : char ;
  Begin
    Tex.mode := direct ; reset ( tex );
    Repeat
      Write ( ' donner le N° de la ligne a supprimer');Readln ( i );
      X := '*****';{ marquage }
      Seek ( tex, i ); writeln ( tex , x );
      Writeln( 'autre ligne a supprimer o(ui) ou n(on)'); Readln( rep );
      Until rep in [ 'n','N' ] ;
      Close ( tex );
    End;{ fin delete}
  
```

الحذف الفيزيائي :

```

Program deleteAll ( var tex : texte );
  Const Mark = '*****';
  Var x : string ( 80 );
  Temp : Text ;
  Begin
    Reset ( tex ); rewrite ( temp );
    While not eof ( tex ) do
      Begin
        If tex^ = Mark Then      Readln ( tex , x )
        Else
          Begin
            Writeln( temp , x );
          
```

```

                                Readln( tex , x );
                                End;
                                { إعادة نقل من الجذاذة المؤقتة إلى الأصلية }
Rewrite ( tex ); reset ( temp );
While not eof ( temp ) do
    Begin
        Readln ( temp , x ); writeln( tex , x );
    End;
Close ( temp ); close ( tex );
End; { fin deleteAll}

```

الإجراءات في BPW v7 : جمعها في وحدة unit و نناديها في البرنامج.

unit unittext;

```

interface
    type livre = text;
    var page : livre;
    lst : text;
    procedure liText;
    FUNCTION ARRET : boolean;
    procedure ajout;
    procedure edit;
implementation
    procedure liText;
    var x : string;
    begin
        rewrite ( page);
        writeln(' entrer ligne / ligne');
        repeat
            readln(x) ;writeln(page , x);
        until arret;
        close( page);
    end;
    procedure edit;
    var x : string;
    begin
        reset( page);
    end;

```

```

        while not eof( page ) do
            begin
readln(page , x);writeln( x);
            ends;
            close ( page);
        end;
    procedure ajout;
        var x : string;
        begin
            reset( page);
            writeln(' procédure d"ajout :ajouter ligne/ligne');
            repeat
                { to go to the end of the file}
                append(page);  readln( x);  writeln(page , x);
            until arret;
            close ( page);
        end;
    function arret;
        var c : char;
        begin
            writeln('autre fois o(ui) ou n(on)');  readln ( c );
            arret := c in ['n','N'];
        end;
    begin {initialization }
    end.
program fichText;

    uses wincrt, unittext;

    var c : char;

begin
    repeat
        writeln(' Mise a joue de fichier text')' ;
        assign( page , 'fText');  writeln(' c:creation');
        writeln(' a : ajout');      writeln(' e : edition');
        readln( c);
    
```

```

case c of
  'c' : litext;
  'a' : ajout;
  'e' : edit;
end;

until arret;

writeln('end of up dating of text file') ;

end.

```

9. الجذاذات الثنائية المتتالية.

9.Les fichiers Binaires séquentiels.

الجذاذة الثنائية هي كل أنواع الجذاذات ماعدا النوع file of char أو النوع text أو file of string , و التنظيم المتتالي كما ذكرنا تلقائي في باسكال القياسي و لكن كل ترجمان يحدد نوع التنظيم التلقائي l'organisation par défaut ففي BPW v7 التنفيذ المباشر أو ما يسميه القراءة و الكتابة في نفس الوقت Read/write هو التنظيم التلقائي . من خصائص المتتالي أو الكتابة فقط write only أو القراءة فقط read only كما يسميه BPW أنك تفتح الجذاذة للكتابة أو القراءة فقط لا يمكن فعل الإثنين معا و يتم الإعلان عن هذا التنفيذ بالتعيين التالي :

Filemode := 0 ; إذا أردنا القراءة فقط ;

FileMode := 1 ; إذا أردنا الكتابة فقط ;

*مثال : طلبت منك شركة تأمين أن تكتب لها برنامجا يقدم الخدمات التالية :

1. إحداث جذاذة الزبائن علما بأن كل مشترك يسجل حسب النموذج التالي:

* رقم التأمين * الإسم و اللقب * تاريخ الإزدياد * الوظيفة * الحالة المدنية: إن كان متزوجا

: إسم ولقب الزوجة و عدد الأطفال . و إن كان أعزب : لا شيء * العنوان * الجنسية .

2. إضافة مشترك أو أكثر .

3. تغيير معلومات مشترك أو أكثر .

4. حذف زبون أو أكثر .

γ الحل : في هذا المثال سنعرض طريقة إستحداث mise à jour الجذاذة المنظمة تتابعيا

: خطوة خطوة و نشرح الأسباب و الحلول للوصول إلى الخوارزمية الأمثل . بإستعمال

ترجمان BPW7 .

& التصريح بالبنية :

Program Assurance;

Uses wincrt ;

Type assure = Record

Número = integer;

Nom_Pr : string(. 30.);{ou string[30]}

Date_Naiss : string(.15.) ;

Profession : string (. 25.);

Adresse : string(. 30.);

Nationalite : string (. 25.);

Case sit_Fam : char OF

'm','M' : (nom_EP : string (. 30.);

Nb_Enf : integer);

'c','C' : ();

end ; { fin du case et du record }

Fichier_ass = File of assure ;

Var FasPer , FasNv : Fichier_ass ;

1.9 إحداث الجذاذة .

9.1 Création du fichier.

* الخطوات :

1. فتح الجذاذة للكتابة .

2. قراءة تسجيل واحد .

3. كتابة التسجيل في الجذاذة فيزيائيا .

4. أعد الخطوات 2 و 3 إلى أن ينتهي المستعمل من إدخال التسجيلات .

5. غلق الجذاذة .

* الإجراء : نبدأ بكتابة إجراء لقراءة تسجيل واحد : الخطوة 2 .

```

Procedure lit_enrg ( var E : assure );
Begin
  With E do
    Begin
      Write('Numero d"assuré :');readln(numero);
      Write('Nom prenom:');readln(nom_Pr);
      Write('date naissance:');readln(date_naiss);
      Write('profession:');readln(profession);
      Write('adresse:');readln(adresse);
      Write ('nationalité:');readln(nationalité);
      Write('situation familiale :m(arié),c(elib)');
      readln(sit_fam);
    case sit_fam of
      'm','M':begin
        write('nom epouse');readln( nom_ep);
        write('nombre d"enfant");readln(nb_enf);
        end;
      'c','C': ;
    end;{ fin du case}
  end;{ with}
end;{ lit_enrg}
procedure create ( var f : fichier_ass);
var personne : assuré ;
begin
  filemode := 1;{ write only file}
  assign( f ,'fichier_ass');
  rewrite ( f);
  repeat
    lit_enrg( personne);
    write( f , personne );
  until arret ;{ fonction arret externe dans une unit}

```

```
close ( f);
end;{ fin create }
```

2.9 الإضافة في النظام التتابعي.

9.2 L'ajout en mode séquentiel.

* نذكر بمبدأ الإضافة في النظام التتابعي وتعني كتابة تسجيل جديد في الجذاذة و تعني فتح الجذاذة للكتابة بـ: rewrite وهي عملية تمحي محتوى الجذاذة .. لذلك نضطر إلى فتح جذاذة ثانية للكتابة و نقل فيها محتوى الأولى ثم نضيف التسجيلات الجديدة . و تكون الخطوات كالتالي :

1. فتح الجذاذة المنبع للقراءة و الهدف للكتابة .
 2. نقل جميع التسجيلات من المنبع إلى الهدف .
 3. قراءة تسجيل من الدخل .
 4. كتابة التسجيل في الجذاذة الهدف .
 5. إعادة الخطوات 3 و 4 إلى نهاية الدخل .
- غلق الجذاذتين .
- *نبدأ بكتاب إجراء نقل جذاذة إلى أخرى .

```
Procedure copy ( var FA,FN: Fichier_ass);
  Var enrg : assure
  Begin
    While not eof ( fa ) do
      Begin
        Read ( FA,enrg);
        write(FN,enrg);
      end;
    end;{copy}
```

~ إجراء الإضافة :

```
Procedure ajout ( var FA,FN : Fichier_ass );
```



```

Var personne : assure;
Begin
  Reset ( Fa);Rewrite( FN);
  Copy ( fa ,fn );
  نقل الجذاذة المنبع إلى الهدف حتى نظيف في نهاية الجذاذة الهدف
Repeat
  Writeln('Lecture d'un assuré);
  Lit_enrg ( personne );
  write ( fn , personne ) ;
  until arret ;
  close ( fa ); close ( fn );
end; { fin ajout }

```

3.9 التغيير في النفاذ التتابعي .

9.3 Modification en mode séquentiel.

~ خطوات التغيير .

مثل الإضافة التغيير يحتاج إلى كتابة التسجيل المطلوب تغييره في الجذاذة و هذا يقتضي فتح الجذاذة للكتابة بـ: .rewrite

و هي عملية تمحي محتوى الجذاذة. فإننا بالضرورة نحتاج إلى جذاذة ثانية للقيام بالكتابة فيها .وهي كالتالي :

1. فتح الجذاذة المنبع للقراءة و الجذاذة الهدف للكتابة .

2. قراءة عنوان التسجيل المراد تغييره و هو رقم التأمين و الذي نسميه مفتاح البحث عن التسجيلات .

3. البحث عن التسجيل كالتالي:

مادام لم يعثر عليه ..ننقل التسجيلات إلى الجذاذة الهدف .

إن وجد إدخال التسجيل الجديد و كتابته في الجذاذة الهدف

..و إن لم يوجد يعلن عن خطأ

4. نقل ما تبقى من تسجيلات من الجذاذة المنبع إلى الهدف ثم غلقها.
5. نقل الجذاذة الهدف إلى المنبع لتصبح التغييرات في المنبع . و أما الهدف لا تستعمل إلا للقيام بعمليات التغييرات .
6. إذا أردنا التغيير أكثر من مرة ما علينا إلا نداء الإجراء مرات عديدة .

4.9 الحذف في النظام التتابعي .

9.4 Suppression en Mode Séquentiel.

مثل التغيير الحذف يعني نقل التسجيلات الدائمة و نسيان التسجيلات الغير مرغوب فيها التي يحددها المستعمل برقمها و التي نفترض أنها مسجلة تتابعيا أي أن خلال الأحداث تم إدخال التسجيلات مرتبة حسب الحقل رقم التسجيل و إلا علينا بالقيام بترتيبها بإجراء ترتيب الجذاذات tri de fichier قبل أن نقوم بأي تغيير أو حذف فيها.

~ الخطوات :

1. فتح الجذاذة المنبع للقراءة و الهدف للكتابة .
 2. قراءة مفتاح التسجيل : أي رقم التأمين . المراد حذفه.
 3. البحث عن التسجيل :
- ما دام التسجيل لم يعثر عليه و لم نصل إلى نهاية الجذاذة المنبع :
- أنقل FA إلى FN
4. إن وجد نقفز إلى التسجيل الموالي و نتابع النقل ثم نغلق الجذاذتين.
 5. ننقل الجذاذة الهدف إلى المنبع ليتم الحذف على المنبع .
 6. إذا أردنا حذف تسجيل آخر ما علينا إلا نداء الإجراء مرات عديدة .

5.9 عرض محتوى الجذاذة .

9.5 Consultation du fichier .

عملية العرض على الشاشة بسيطة:

1. فتح الجذاذة للقراءة .
2. قراءة التسجيل من الجذاذة في متغير .
3. كتابة المتغير على الشاشة أو الطابعة.
4. تكرار الخطوات 2 و 3 إلى نهاية الجذاذة .

6.9 الخلاصة :

9.6 Conclusion

إكتفينا بعرض خطوات كل إجراء و بعض الإجراءات بدون اللجوء إلى كتابة كل الإجراءات حيث أنها متشابهة .

تعرفنا على طريقة إستحداث الجذاذة التتابعية séquentiel و لحظنا كيف نظطر إلى القيام بكل عملية من إضافة أو تغيير أو حذف على جذاذة جديدة حيث يمكن الكتابة فيها و الرجوع للجذاذة المنبع لكي تصبح كل عملية كأنها تمت على الجذاذة الأولى , و هي طريقة تأخذ الوقت و غير بسيطة الإستعمال فلا بد من إقتراح خوارزمية إستحداث بسيطة و تمكننا من القيام بكل العمليات دفعة واحدة و بدون اللجوء إلى إدخال البيانات بطريقة التخاطب conversationnel و لكن دفعة واحدة batch processing/traitement par lot . لكي نقوم بهذا الإستحداث mise a jour بشكل جيد و دفعة واحدة نقترح الخوارزمية التالية:

1. إحداث جذاذة تُجمع فيها طلبات الإستحداث من تغيير , حذف أو إضافة و لنسميها الجذاذة المتحولة fichier mouvement.

2. كتابة إجراء يقرأ طلبات الإستحداث من الجذاذة المتحولة و يدخل على الجذاذة المنبع أو الدائمة fichier permanent الإستحداث المطلوبة بإحداث جذاذة هدف fichier

. destination

و هكذا سنعيد التفكير في بنية بيانات data structure لنتمكن من إحداث جذادة الطلبات ببياناتها المسماة الجذادة المتحولة fichier mouvement .

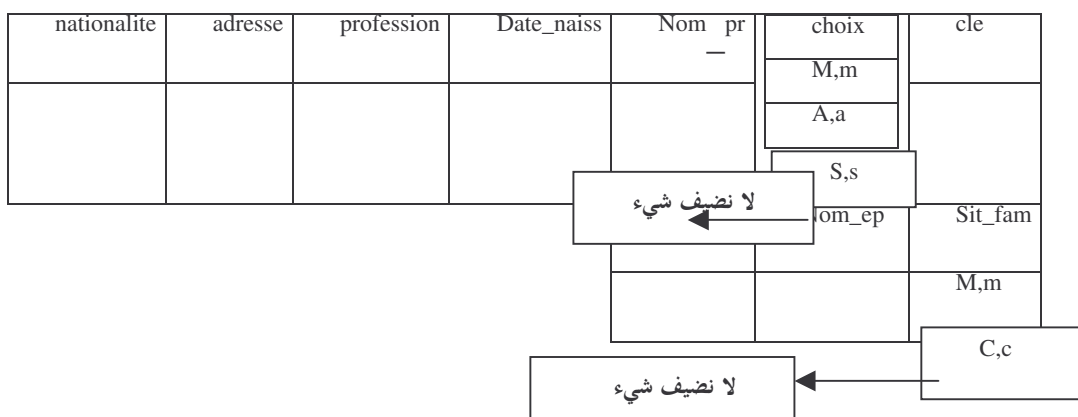
ملاحظة هامة : إستحداث الجذادات المتتابعة la mise à jour séquentiel كان متداولاً كثيراً لإستعمال الأشرطة أما اليوم فقد يكون غير متداول كثيراً و لكن دراسة طريقة الإستحداث توفر لك شيء من الوقت عند إطلاعك بمهام كتابة و تخطيط برامج تطبيقية كبيرة أو مراجعة البرامج القديمة على حاسبات ما زالت تستعمل الأشرطة أو برامج الصيانة maintenance إلى آخره ...

7.9 التصريح بالجذادة المتحولة.

9.7 Déclaration du fichier mouvement.

بنية الجذادة المتحولة :

1. مفتاح النفاذ إلى التسجيل الهدف المطلوب حذفه أو تغيير محتواه .
2. وضع حقل لإختيار كل من الإضافة أو التغيير أو الحذف: يتبع بالبيانات المطلوبة لكل من الإضافة أو التغيير و لا شيء في حالة الحذف .
3. و الحقول الأخرى الثابتة .



كما ترى لا تضيف شيء في حالة الحذف: s و في حالة الأعزب: c و هي بنية فيها متغيرين
2 case . نقسمها إلى جزئين كتالي:

```

Type information = record
    Nom_pr : string (. 30. );
    Date_naiss : integer ;
    Profession : string (. 30.);
    Adresse : string (. 30.);
    Nationalité : string (.30.);
    Case sit_fam : char of
        'm','M' : (nom_ep :string( 30);{marié}
                    nb_enf:integer);
        'c','C' : ( );{célibataire}
    end;
enrg_mouvt = record
    cle : integer ;{ est le Numero d"assurance }
    case choix :char of
        'm','M' : ( info_ass : information );{modify}
        'a','A' : (inf_assur : information );{add}
        's','S' : ( );{delete}
    end;
enrg_permt= record
    cle : integer ;
    data_ass : information ;
end;
fichier_mvt= file of enrg_mouvt ;
fichier_permt = file of enrg_permt ;
var FM : fichier_mvt ;
    FPAnc ,FPNouv : fichier_permt ;

```

8.9 إحدات الجذاذة المتحولة .

9.8 Création du fichier mouvement.

الجذاذة المتحولة أو جذاذة الإستحداث **fichier de mise a jour** هي التي ستُخزن
فيها طلبات الإستحداث :

* الخطوات :

1. فتح الجذاذة للكتابة .
2. قراءة الطلب الأول أي التسجيل الأول المتحول.
3. كتابته في الجذاذة .
4. إعادة الخطوات 2 و 3 إلى نهاية الطلبات .
5. غلق الجذاذة .

χ إجراء قراءة تسجيل واحد :

```

Procedure Lit_enrgMv (var Eg : enrg_mouvt );
Begin
    With eg do
        Begin
            Write(' Donner la clé :Numero assuré);
            Readln( cle );
            Write('Choix:m,M=marié;a,A=ajout;s,S=suppression);
            Readln( choix);
            Case choix of
                'a','A': with inf_assur do{ cas ajout }
                    begin
                        readln( nom_pr);readln(date_naiss);
                        readln(profesion);readln(adresse);
                        readln(nationalité);readln(sit_fam);
                        case sit_fam of
                            'm','M':begin
                                readln(nom_ep);
                                readln(nb_enf);
                                end;{ fin choix m,M }
                            'c','C' :    ;{ celibataire : vide }
                        end ;{ fin case sit_fam}
                    end; { fin with inf_assur }
                'm','M' : with info_ass do{ cas Modification}
                    begin
                        readln( nom_pr);readln(date_naiss);
                        readln(profesion);readln(adresse);

```

```

readln(nationalité);readln(sit_fam);
case sit_fam of
  'm','M':begin
    readln(nom_ep);
    readln(nb_enf);
    end;{ fin choix m,M }
  'c','C' :   ;{ celibataire : vide }
end ;{ fin case sit_fam }
end;{ fin with info_ass }
's','S' :   ;{ cas suppression vide la cle suffit }
end;{ fin case choix }
end;{ fin with eg }
end;{ fin lit_enrg }

```

χ إجراء الإحداث .

```

Procedure creat ( var FM : Fichier_mvt );
Var em : enrg_mouvt ;
Begin
  Rewrite ( FM );
  Repeat
    Lit_enrgMv( em );
    Write( FM,em);
  until arrêt ;
  close ( fm);
end;{ creat }

```

9.9 إحداث الجذاذة الدائمة.

9.9 Création du fichier Permanent .

* الخطوات :

1. فتح الجذاذة للكتابة .
2. قراءة تسجيل دائم .
3. كتابة التسجيل في الجذاذة الفيزيائية .
4. تكرار الخطوات 2 و 3 إلى إنهاء الدخل .

5. غلق الجذاذة .

X إجراء قراءة تسجيل دائم :

```

Procedure lit_ergPer( var EP : enrg_permt );
Begin
  With EP,inf_ass do
    Begin
      Readln(cle);readln(nom_pr);
      Readln(date_naiss);readln(profession);
      Readln(nationalité);readln(sit_fam);
      Case sit_fam of
        'm','M':begin
          readln(nom_ep);
          readln(nb_enf);
          end;
        'c','C': ;
      end;{ fin du case}
    end;{with }
  end;{lit_ergPer}

```

X إجراء الإحداث .

```

Procedure creat ( var FP : fichier_permt);
Var ep : erg_permt ;
Begin
  Rewrite ( fp);
  Repeat
    Lit_errgPr( ep );
    Write ( fp, ep );
  until arret ;
  close ( fp);
end;{creat}

```

10.9 خوارزمية الإستحداث التتابعي.

9.10 Algorithme de mise à jour.

الإستحداث الذي سنقوم به على الجذاذة الدائمة fichier permanent بإستعمال الجذاذة

المتحولة fichier mouvement يُحدث جذاذة جديدة new file تحتوي على كل التسجيلات التي لم تغير أو تحذف و التسجيلات التي غيرت و أضيفت . وتكون الخوارزمية كالتالي :

1. فتح الجذاذة الدائمة للقراءة و الجذاذة الجديدة للكتابة .

2. فتح الجذاذة المتحولة للقراءة .

3. قراءة أول طلب من الجذاذة المتحولة :

إذا كان طلبا للتغير نبحث عن التسجيل الذي رقمه يناسب المفتاح في الجذاذة الدائمة وهو ما يقتضي المرور على التسجيلات من البداية حتى نجده فنُدخل التغيرات في الجذاذة الجديدة .

4. قراءة الطلب الثاني من الجذاذة المتحولة FM هب أنه طلب حذف لتسجيل ما

ويقتضي الرجوع إلى بداية الجذاذة الدائمة FP للبحث عنه ثم نمر على جميع التسجيلات التي رقمها لا يناسب مفتاح البحث و نقلها حتى إذا وجدنا التسجيل المطلوب لا ننقله و نواصل نقل الباقي إلى نهاية الجذاذة الدائمة .

ففي كل مرة للبحث عن التسجيل المطلوب إستحدثه نضطر للرجوع إلى بداية الجذاذة الأم و لا ننسى أننا قد قمنا بفعل الإستحداث عليها و تم ذلك بإحداث جذاذة جديدة و التي يجب نقلها إلى الجذاذة الدائمة و هكذا نرى أن لكل عملية نقوم بها نمر على الجذاذة الدائمة كليا . ثم ننقل الجذاذة الجديدة إليها نفس عدد المرات , و هي طريقة كما تلاحظ لم تزدنا إلا تعباً و ضياعاً للوقت بالرغم أننا إختصرنا الطلبات في جذاذة واحدة . فما الحل إذا حتى نقوم بالإستحداث دفعة واحدة و مرة واحدة ؟

X الحل : هو أن نجد طريقة تسمح لنا بمعالجة الطلبات دفعة واحدة على الجذاذة FP

تسجيلا بتسجيل , وهذا ممكن إذا رتبنا trié /sorted الجذاذتين : المتحولة و الدائمة على نفس الحقل الذي هو مفتاح clé البحث , حيث عندما يكون طلب إستحداث التسجيل رقم 15 مثلا فإننا سنجده في الجذاذة الدائمة في المرتبة 15 و هكذا كلما تقدمنا في قراءة الجذاذة

المتحولة نجد كل طلب يناسبه تسجيل في نفس الرتبة في الجذاذة الدائمة , و لا حاجة للبحث عنه و الرجوع للوراء و بالتالي يتم إستحداث الجذاذة الدائمة دفعة واحدة , و تحدث جذاذة جديدة مباشرة تحتوي على التسجيلات القديمة و التي تم إستحداثها.
نلخص هذا الحل كما يلي :

مثلا : الجذاذة الدائمة : fichier permanent التالية :

5	خ	8	ق	9	ن	21	ض
---	---	---	---	---	---	----	---

على سبيل المثال مثلنا حقلين فقط : المفتاح و حقل واحد بحركة واحدة.

و لتكن الجذاذة المتحول fichier mouvement كما يلي :

8	M	س	9	s	66	A	ك
---	---	---	---	---	----	---	---

عبرنا عن المعلومة information بحرف واحد أما الحقل المفتاح فهو مرتب مثل حقل الجذاذة الدائمة بالإضافة إلى حقل الطلبات او الحقل الاختياري choix المتمثل في s و m و a : حذف أو تغيير أو إضافة .

و تكون النتيجة الجذاذة الهدف أو الجديدة fichier résultat ou nouveau

5	خ	8	س	21	ض	66	ك
---	---	---	---	----	---	----	---

و هي كما تلاحظ تحتوي على تسجيلات لم تُغير من الجذاذة الدائمة مثل رقم 5 و 21 و تسجيلات غُيرت من الجذاذة المتحولة مثل رقم 8 و تسجيلات أُحدثت أو أُضيفت: نقلت من الجذاذة المتحولة مثل رقم 66 و لكن لا نجد التسجيلات التي طلب حذفها مثل رقم 9 .
**مثال 2: من أجل تبسيط المسألة نقترح مثال آخر : إستحداث جذاذة معلومات زبائن

بنك: كل زبون معرف برقم و اسمه و نوع العملية (دفع أو إستلام نقود) و كم . cle
+intitule du compte(nom prenom)+operation(depot ou retrait)+montant.

بجذاذة متحركة جمعت فيها الخدمات fichier mouvement des services

*** و إليك الإجراءات كاملة للإستحداث في النظام التتبعي:

```

program bank;
    uses wincrt;
    const hv = maxInt;
    type information = record
        int : string ;
        op : char ;
        mt : real;
    end;
    comptPert = record
        cle : integer;
        inf : information;
    end;
    comptMvt = record
        cle : integer;
        case choix : char of
            'm','i' : (inf : information );
            's' : ( );
        end;
    FbPert = File of comptPert ;
    FbMvt = File of comptMvt;
    var fba , fbn : FbPert;
        Fbm : FbMvt ;
        choix,rep : char;
    function arret : boolean ;
        var rep :char ;
        begin
            write(' autre enrg. o/n');readln(rep);
            arret := rep in ['n','N'];
        end;
    procedure creatFb ( var fb : FbPert );
        var enrg : comptPert ;
        rep : char ;
        begin
            rewrite ( fb );
            writeln(' Création du Fichier Permanent des comptes:');

```

```

repeat
  with enrg , inf do
    begin
      write('Numero de compte :');readln( cle );
      write('Intitule du compte:');readln( int );
      write('Code d"operation");readln( op );
      write('Montant ?:');readln( mt );
    end;
    write(fb,enrg);
  until arret;
writeln('Création du fichier Permanent Termine');
close ( fb );
end;{creatFb}
procedure creatFm( var fm : FbMvt );
var cb : comptMvt ;
begin
  rewrite ( fm );
  writeln( 'Saisi du Fichier mouvement:');
  repeat
    with cb do
      Begin
        write('entrer le numéro de compte :'); readln( cle );
        write('Entrer m(odifier) i(nserer) s(uprimer)');
        readln( choix ) ;
        case choix of
          'm','i' :{modification ou insertion}
            with inf do
              begin
                write('entrer le Nom prenom :'); readln( int );
                write('Operation :');readln( op );
                write('Montant :');readln( mt );
                write(fm,cb);
              end;
          's':with inf do
            begin  int := ' '; op := ' '; mt := 0;
              write(fm,cb) ;
            end
        end;{case}
      end;
    end;
  until arret;
end;{creatFm}

```

```

        end;{with}
    until arret;
    close( fm ); writeln('Fin fichier Mouvement');
end;{creatFm}
procedure miseAJour( var Fa,Fn : Fbpert;Var Fm : FbMvt );
    var cp,cn : comptPert;
        cm : comptMvt;
        fin : boolean;
        inf : information;
begin
    reset( fa ); reset( fm ); rewrite ( fn );
    read(fa,cp);    read(fm,cm);
    repeat
        fin := eof(fa) and eof(fm) ;
        if cp.cle < cm.cle Then{ copy }
            begin
                write( fn , cp );
                if not eof(fa) then
                    read(fa , cp ) else cp.cle := hv ;
                end ;
                { cas modification ou suppression }
                if cp.cle = cm.cle Then
                    begin
                        case cm.choix of
                            'm':begin
                                cn.cle := cm.cle;
                                cn.inf := cm.inf ;
                                write(fn,cn);
                                end;
                            's': ;
                        end;{ fin case }
                    {j'avance dans le fichier mouvement}
                    if not eof(fm) then
                        read(fm,cm ) else cm.cle := hv ;
                    { j'avance dans le fichier permanent }
                    if not eof( fa ) then
                        read( fa , cp ) else cp.cle := hv ;
                    end ;{ fin then }
            end ;
    until fin;
end;

```

```

if cp.cle > cm.cle then
    {cas d'une insertion seulement}
    begin
        case cm.choix of
            'i': begin
                cn.cle := cm.cle ;
                cn.inf := cm.inf ;
                write(fn,cn);
            end;
        end;{fin case}
        {j'avance dans le fichier mouvement}
        if not eof( fm ) Then
            read(fm,cm) else
                cm.cle := hv ;
        end;
    until fin;
    close(fa);close(fn);close(fm);
end;{fin miseAJour}
procedure ecritFbn( var fbn : fbPert);
var c : comptPert;
begin
    writeln('Contenu du fichier: ');
    reset( fbn );
    while not eof ( fbn ) do
        begin
            read( fbn,c );
            with c , inf do
                begin
                    write(cle:15 );write(int:15); write(op:15);
                    writeln(mt:15);
                end;
            end;
        close( fbn);
        writeln('Fin de fichier');
    end;{fin ecritFbn}
procedure ecritFMouv( var fm : fbMvt);
var c : comptMvt ;
begin

```

```

    reset( fm );
    writeln('Contenu du Fichier Mouvement');
    while not eof( fm ) do
        begin
            read(fm,c);
            with c , inf Do
                begin
                    write(cle:15);write(choix:10);
                    write(int:25); write(op:5);
                    writeln(mt:15);
                end;
            end;
        close( fm );
    end;
begin{ prog. Principal}
    assign( fba ,'FichPer');
    assign( fbn ,'FichNv');
    assign( fbm ,'FichMvt');
    Repeat
write('Exemple de mise a jour de fichier permanent en mode séquentiel ')
write('on doit créer le fichier permanent et le fichier');
write(' mouvement et ensuite on appel la procédure de mise a jour');
writeln('et pour s"assurer on affiche le contenu des fichier');
        writeln('1:pour créer le fichier permanent');
        writeln('2:pour afficher le contenu du fichier permanent');
        writeln('3:pour creer le fichier mouvement');
        writeln('4:pour afficher le contenu du fich. Mouvement');
        writeln('5:pour démarrer la mise a jour');
        writeln('6:pour afficher le contenu du Nouveau fichier');
    readln(choix);
    case choix of
        '1': creatfb( fba );
        '2': ecritfbn( fba );
        '3': creatfm( fbm );
        '4': ecritFMouv( fbm );
        '5': miseAJour( fba,fbn,fbm );
        '6': ecritfbn( fbn );
    end;

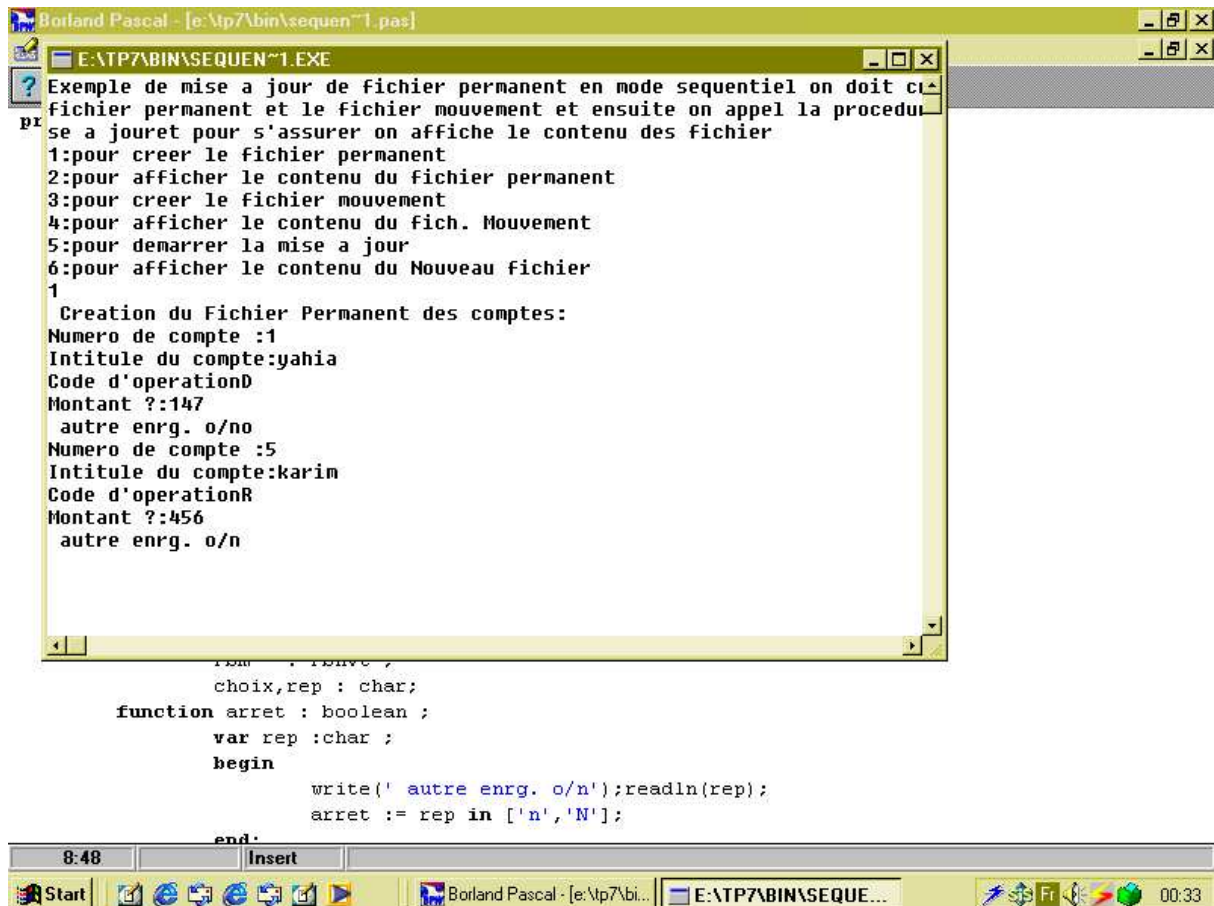
```

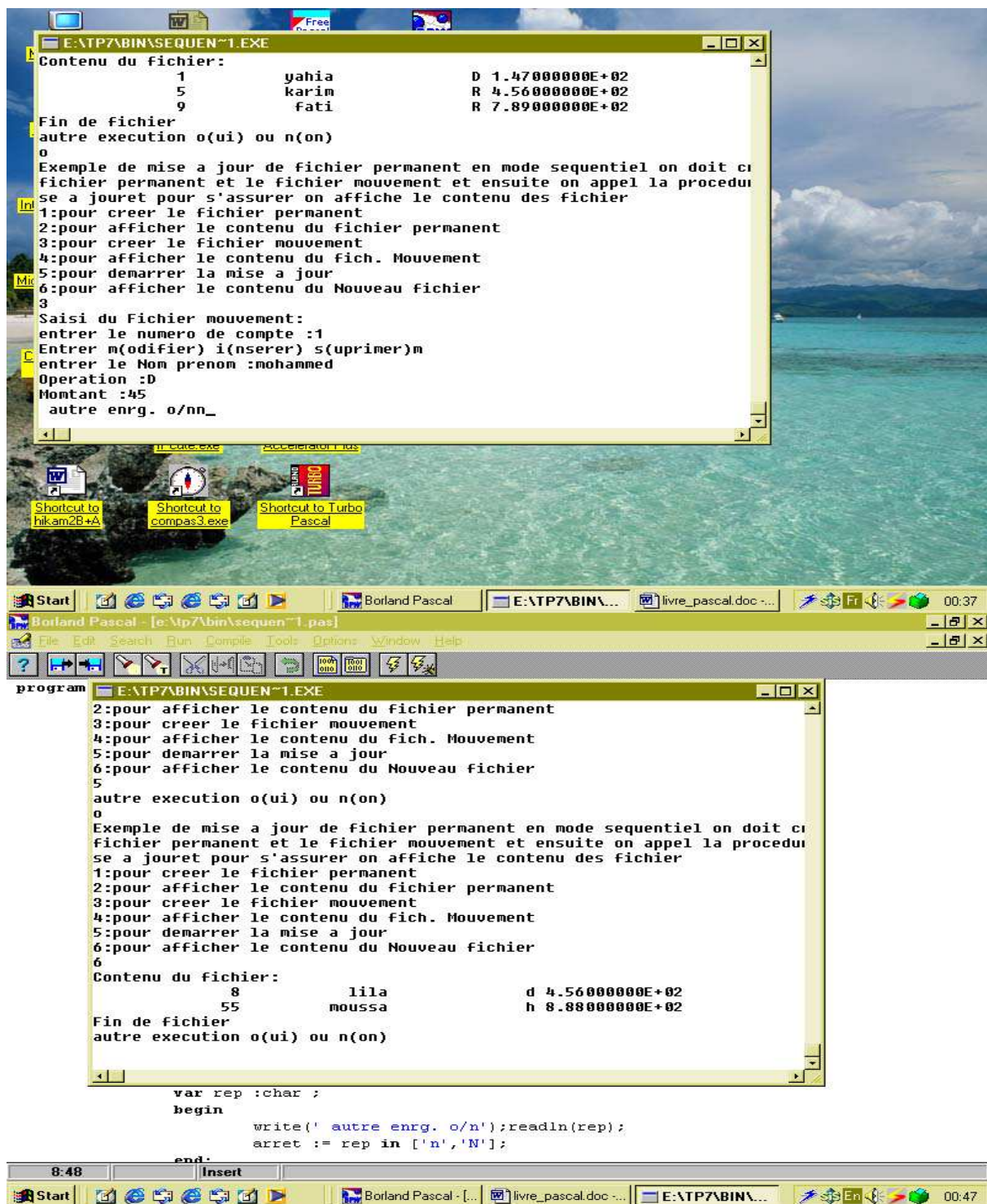
```

writeln('autre execution o(ui) ou n(on) ');
readln(rep);
    until rep in ['n','N'];
writeln('au revoir');
end. {fin programme}

```

و إليك التنفيذ تحت WindowsMe





11.9 خلاصة النفاذ التتابعي .

9.11 Résumé sur l'accès séquentielle.

عندما نريد معالجة الجذاذة بأكثر من 50% يستحسن إختيار النظام التتابعي . التتابعي يعني المرور على جميع التسجيلات من بداية الجذاذة إلى نهايتها . التتابعي لا يقبل القراءة و الكتابة على نفس الجذاذة في نفس الوقت .

ميزاته : avantage du séquentiel

المعالجة دفعة واحدة : traitement par lot .

توفير جذاذة قديمة و أخرى جديدة للإحتفاظ بها في الأرشفة .

عيوب التتابعي :

لا يسمح بالنفاذ الفردي لتسجيل واحد بغية إستحداثه , يجب إنتظار جمع الطلبات في جذاذة متحولة قبل البدء بالمعالجة و هي عملية قد تطول... وجود طلبات خاطئة قد يأخر نتيجة الإستحداث.

10 التنظيم المباشر للجذاذات.

10 Accès direct aux fichiers/ Direct file access .

1.10 الهدف .

10.1 But.

الهدف من النفاذ المباشر هو علاج المشاكل التي تطرح في النظام التتابعي : القراءة و الكتابة ممكنة على نفس الجذاذة في نفس الوقت بدون أن تحمل التسجيلات السابقة و هو ما يوفر لنا جذاذة كنا نستعملها للكتابة في النظام التتابعي . النفاذ المباشر يعني بداهة استعمال القرص كوسيلة تخزين إذ تنظيم القرص يسمح بالنفاذ المباشر حيث يمكن النفاذ إلى التسجيل عن طريق الكتلة bloc المخزن فيها في القطاع المعين secteur و المسار أو السكة

المعيّنة piste و الوجه المعين face مع وجود رؤوس عدّة للقراءة و الكتابة .
 نذكر بتنظيم القرص الفيزيائي : يغطي القرص من مادة قابلة للتمغنط ثم يشكل
 formater برنامج خاص يوفره نظام التشغيل وهي عملية تقوم برسم سكك
 pistes على كل الوجه ثم يقسم كل وجه إلى قطاعات secteurs و كل قطاع يقسم إلى كتل
 blocs , كل كتلة يوضع بها فهرس يسمح بالنفاد المباشر للتسجيلات المخزنة عليها .

2.10 التصريح بالنفاد المباشر .

10.2 Déclarer le mode direct

النفاد المباشر تلقائي في BPW 7 .

Le mode directe est declare par Défaut en BPW 7.

من خصائصه :

*الإجراء reset (f) يفتح الجذاذة للقراءة و يسمح بالكتابة .

* الإجراء rewrite(f) يفتح الجذاذة للكتابة و يسمح بالقراءة كذلك و لا يمحي محتواها

3.10 الإجراء الخاص بالنفاد المباشر .

10.3 Procedure Seek(var f;n:integer);

هذا الإجراء يشير إلى التسجيل رقم n في الجذاذة f و n من نوع longint في BPW7

علما بأن seek تبدأ العد من الصفر.

4.10 الإضافة في النظام المباشر.

10.4 L'ajout en mode direct.

**مثال : أكتب برنامج يقدم خدمات تسيير مكتبة: كل كتاب مسجل كما يلي:

رقم التسجيل * عنوان الكتاب * مؤلفه *

* إضافة كتاب أو أكثر إلى جذاذة المكتبة .

- * تغيير معلومات كتاب ما أو أكثر إلى الجذاذة .
- * حذف كتاب من الجذاذة أو أكثر .
- * عرض محتوى الجذاذة أي كل المكتبة.
- نعرض الخطوات فقط ثم في نهاية الفقرة تجد البرنامج كاملاً مع التنفيذ.
- هذا المثال يشبه المسألة رقم 5 , أنظر النص و الحل في آخر الكتاب .
- * خطوات الإضافة :

1. الإعلان عن النفاذ المباشر إن كان ضروريا .
2. فتح الجذاذة للقراءة أو الكتابة .
- 3 . الذهاب إلى نهاية الجذاذة باستعمال الإجراء : seek(f,filesize(f)).
4. قراءة التسجيل الجديد أي معلومات كتاب واحد .
5. كتابة التسجيل في الجذاذة أي معلومات الكتاب في المكتبة.
6. إعادة الخطوات 4 و 5 إلى نهاية الإضافات .
7. غلق الجذاذة .

10.5 التغيير في النفاذ المباشر.

10.5 Modification en mode directe.

χ خطوات التغيير :

1. فتح الجذاذة للقراءة أو الكتابة .
- قراءة مفتاح البحث و الذي يمكن أن يكون :رقم التسجيل أو العنوان أو المؤلف ...
- البحث عن التسجيل بمعرفة المفتاح : إذا وجد نقرأ التغيير أي الكتاب الجديد ثم
- كتابته في المكان المشار إليه .
- إذا لم يوجد نعلن عن الخطأ .

تكرار 2, 3, 4 إلى نهاية طلبات التغيير.

غلق الجذاذة .

* طريقة ثانية: تستعمل فيها دالة للبحث عن التسجيل بمعرفة المفتاح إن وجدته ترجع رتبته في n و هو الوسيط الذي ننفذ به مباشرة إلى التسجيل بالإجراء $seek(f,n)$.

6.10 الحذف في النفاذ المباشر.

10.6 La suppression en mode Directe.

الحذف عملية إلغاء التسجيل من الجذاذة على وجهين :

إما أن يكون منطقيا بمعنى أن يصبح التسجيل غير قابل للإستعمال و لكن يبقى موجودا في الجذاذة و هو الحذف المنطقي $suppression\ logique$ و إما أن يلغى تماما بمعنى أن لا يبقى موجودا في الجذاذة و تسمى الحذف الفيزيائي $suppression\ physique$.
& الحذف المنطقي :

هذا النوع من الحذف يقتضي وضع علامة خاصة على حقل أساسي في التسجيل مثل رقم التسجيل كي يصبح غير قابل للإستعمال .
*الخطوات :

1. تعيين النفاذ المباشر.

2. فتح الجذاذة للقراءة أو الكتابة .

3. قراءة مفتاح البحث عن التسجيل المراد حذفه.

4. البحث عن التسجيل بطريقة تتابعية أو إستعمال دالة البحث وإرجاع الرتبة .

5. وضع العلامة على الحقل المفتاح مثل : N_{inv} في التسجيل المشار إليه.

6. تكرار الخطوات 3 إلى 5 إذا أردنا حذف آخر.

7. غلق الجذاذة.

/ الحذف الفيزيائي: بما أن هذا النوع من الحذف يقتضي مسح التسجيل كليا فإن أحسن طريقة للوصول إلى هذا الهدف هي استعمال جذادة ثانية تنقل إليها جميع التسجيلات ما عدى التي يطلب حذفها ثم نعيد نقل الجذادة الجديدة إلى الأولى حيث تصبح خالية من التسجيلات التي تم حذفها , هذه الطريقة لا تتطلب نفاذا مباشرا إذ لا نحتاج إلى القراءة و الكتابة في نفس الجذادة و لا النفاذ المباشر للتسجيل المراد حذفه .

و قد نستعمل النظام المباشر في الحذف الفيزيائي إذا تتبعنا طريقة الحذف المنطقي أولا ثم عليه نبني الحذف الفيزيائي بالبحث عن التسجيلات المعلقة marqué أي التي حذفت منطقيا لنمسحها نهائيا.

** و إليك الإجراءات كاملة :

```

program maktaba ;
  uses winCRT;
type livre = record
  ninv : integer;
  auteur : string;
  titre : string ;
  editeur : string;
  nvol : integer;
  datEd : string;
  case sortie : char of
    'O' : ( date : string[10] );
    'N' : ( );
  end;
  fichLiv = file of livre ;
var F : FichLiv;
  c : char;
  function arret : boolean;
  begin
    write('entrer n ou N pour sortir ');
    arret := readkey in ['n','N'];
  end;

  procedure lit_enrg( var enr : livre );

```

```

begin
  with enr do
    begin
      write('Numero d'inventaire : ');readln( ninv );
      write('Titre :');readln( titre );
      write('Auteur :');readln( auteur );
      write('Editeur :');readln( editeur );
      write('Date d'edition');readln( datEd );
      write('Nombre de volume :');readln( nvol );
      write ('Livre sortie O(ui) ou N(on) :');readln(sortie);
      case sortie of
        'O','o' : begin
          write('entrer la date de sortie');
          readln( date );
        end;
        'N' : ;
      end;{ case }
    end;{ with }
  end;
procedure creation ( var f : fichLiv );
  var book : livre ;
  begin
    rewrite( f );
    repeat
      writeln('entrer un enregistrement');
      lit_enrg( book );
      write ( f , book );
    until arret ;
  end;
procedure ajout( var f : fichLiv );
  var e : livre ;
  begin
    reset( f );  seek ( f , filesize( f ));
    Repeat
      lit_enrg( e );
      write( f , e );
    until arret ;
    close( f );

```

end;

```

Procedure sup( var f : fichLiv );
{ suppression logique en ecrivant - numeroInventaire dans le champs num
inventaire }
var i : integer ; trouve : boolean ; L : livre;
begin
repeat
  reset( f ); trouve := false;
  writeln('entrer le numéro d'inventaire du livre a supprimer :');
  readln( i );
  while not eof( f ) and not trouve do
    begin
      read( f , L );
      if i = L.Ninv Then begin
        L.Ninv := - L.Ninv ;
        seek( f, filepos( f ) -1 );
        write ( f , L ); Trouve := true;
      end;
    end;
  if not trouve then writeln(' Enregistremnet non trouve');
until arret;
close( f );
end;
procedure ecris ( var f : fichLiv );
var e : livre ;
begin
  reset( f );
  writeln('N°inventaire Titre Auteur Editeur Date d"edi Nb Vol');
  while not eof ( f ) do
    begin
      read( f ,e );
      with e do
        begin
          writeln(Ninv:8,Titre:8,Auteur:8,Editeur:10,DatEd:15,Nvol:10);
          case sortie of
            'O' : writeln('livre sortie a la date :',date:8);
            'N' : writeln(' livre non sortie ');
          end;{case}
        end;
      end;
    end;
  end;

```



```

    end;{with}    end;{while}
        close( f );
    end;{ecris}
procedure ecris1livre ( var L : livre );
begin
    with L do
    begin
        writeln(Ninv:8,Titre:8,Auteur:8,Editeur:10,DatEd:15,Nvol:10);
        case sortie of
            'O' : writeln('livre sortie a la date :',date:8);
            'N' : writeln(' livre non sortie ');
        end;{case}
    end;
end;
{modifie un ou plusieurs enregistrements en mode direct }
procedure modif( var F : Fichliv );
    var L : livre ; trouve : boolean;  Nmax, i : integer ;
begin
    Repeat
        reset( f );    Nmax := filesize( F );
        writeln(' Il y "a ',Nmax :4 , 'Enregistrement');
        writeln('Entrer la num Inventaire du livre a modifier');
        readln( i ); trouve := false;
        while not eof ( f ) and not trouve do
            begin
                read( f , L );
                if i = L.Ninv then begin
                    Lit_Enrg( L );seek ( f , filepos( f )-1);
                    write ( f, L );trouve := true ;
                end;
            end;
        if not trouve then writeln(' Enregistrement absent' ) ;
    until arret;
    close ( f );
end;
procedure chercheTitre ( var f : fichliv );
    var L : livre ; T : string;trouve : boolean ;
begin

```

```

writeln('recherche d"un livre par Titre ');
repeat
    reset( f );trouve := false ;
    write('donner le titre a chercher :'); readln( T );
    while not eof( f ) and not trouve do
        begin
            read( f , L );
            if L.titre = T then
                begin
                    writeln('Livre trouve :'); ecris1livre( L );
                    trouve := true;
                end;
            end;
        if not trouve then writeln('livre absent ');
        writeln('another search y(es ) or N(o)');
    until readkey in ['N','n'];
    close ( f );
end;
Begin    { Programme principal }
Assign( f ,'maktaba');
Repeat
clrscr;
writeln(' Menu Principal :');
writeln('c:creation du fichier ');
writeln('e:édition du fichier');
writeln('a:ajout en fin de fichier');
writeln('S:suppression Logique ');
writeln('M:modification ');
writeln('T : cherche ivre par Titre ');
writeln('B:pour un Break: sortir ');
    Readln( c );
case c of
    'c','C': creation ( f );
    'a','A': ajout( f );
    'm','M': Modif( f );
    's','S': sup( f );
    't', 'T' : chercheTitre ( f );
    'E','e': ecris( f );

```

```

        'b','B': break;
    end;
until arret;
writeln(' Au revoir ');
end.

```

11. التنظيم التتابعي المفهرس .

11.L'organisation séquentiel indexée.

1.11 الهدف.

11.1.But.

عندما تكون عندنا قاعدة بيانات base de donnée من مجموعة كبيرة من الجذاذات بنيتها كبيرة و معقدة ... لتتبع إدارتها و الإستفادة من محتواها يجب أن نوفر للمستعمل خدمات البحث بصورة دقيقة و سريعة ... لتحقيق هذا الهدف نحدث فهارس indexes متنوعة على المواضيع المختلفة التي تحتويها الجذاذات و نقدم الخدمات بإستعمل هذه الفهارس .

2.11 طرق الفهرسة .

11.2 Méthodes d'indexages.

مثال: نريد إحداث قاعدة بيانات مختصة في جمع المعلومات الخاصة بالإستهلاك الفردي للمواد الغذائية : حليب - الخبز - البيض - الجبن-اللحم ... و ذلك حسب الولاية - المدينة - سن المستهلك - وفي أي شهر من السنة. عدد المستهلكين المستوجبين غير محدد مبدئيا , حيث نبدأ بعينة صغيرة ثم كلما حضرنا معلومات جديدة نضيفها للقاعدة . لإستغلال هذه المعلومات تحدث فهارس حسب نوع الخدمة : فهرس بنوع المادة -

فهرس بالولاية - فهرس بالشهر - فهرس بالسن

? بنية الجذاذة:

Matière	Wilaya	Ville	Age	Mois	Année
....

Type Statistics = Record

Matière : string(30);

Wilaya : string(20);

Ville :string(20);

Age : 1..110;

Mois :1..12;

Année :1990..2010;

End;

Base = File of Statistics;

Var Bs : Base ;

مثلا : عندنا الإحصائيات التالية :

Matiere	Wilaya	Ville	Age	Mois	Année
Lait	Oran	Oran	15	2	2000
Pain	Alger	Hyra	24	5	2001
Lait	Annaba	Annaba	19	2	2000
Viande	Laghouat	Lagouat	25	8	2000
lait	Tlemcen	Maghnia	45	6	1999
Sucre	Constantine	Constantine	55	7	2001
Fromage	Bejaia	Bejaia	12	6	2003
Lait	Souk-ahras	Souk-ahras	30	1	2003

1.2.11 الجذاذة الفهرس على شكل قائمة تسجيلات .

11.2.1 Table index dans un fichier index .

من المثال السابق نحدث فهرس على مادة الحليب : أي نحدث قائمة تشير إلى جميع

التسجيلات التي فيها المادة:حليب lait .

Type Table_index = Array [1..max] of integer ;

File_Index = File of Table_index;
Var inx : File_index ;

يعني أن التسجيلات التي تحتوي على معلومة استهلاك الحليب
توجد في الرتب : 1 و 3 و 105 و 205 ...

N°enrg	Posit/file
1	1
2	3
3	105
4	205

هذا الفهرس على شكل قائمة يخزن في جذاذة فهرس الحليب

متكونة من تسجيل واحد من نوع Table_index .

هكذا أحدثنا فهرس خاص بمادة الحليب وقد نعممه ليصبح قابلا للتنفيذ لإحداث

فهرس أي مادة حسب إختيار المستعمل.

أكتب الإجراء كتمرين.

2.2.11 الجذاذة الفهرس .

11.2.2.Fichier Index.

مثال : نريد إدارة مكتبة تحتوي على أكثر من 100000 كتاب بإستعمال النظام

التابعي الفهرس حيث نحتاج إلى فهارس للبحث الآلي و كذا لتقديم خدمات للقراء
كالبحث عن مواضيع معينة . كل كتاب مسجل بـ: رقم التسجيل - العنوان - المؤلف -
الموضوع - الناشر - التاريخ...

```
Const Nmax = 100000;
Type Livre = record
  Ninscp : real ;
  Titre : string( 50 );
  Auteur : string( 30 );
  Sujet : string( 500 );
  Editeur : string(30);
  Date : integer ;
End;
Biblio = File of livre ;
Index_titre = record
  NumOrdre : real ;
  Titre : string ( 30 );
```

```

End;
Index_auteur      = record
    NumOrdre      : real ;
    Auteur        : string ( 30 );
End;
Index_sujet = record
    Numordre      : real ;
    Sujet         : string( 30 );
End;
Fich_index_titre  = file of Index_titre;
Fich_index_auteur= file of index_auteur;
Fich_index_sujet  = file of index_sujet ;

```

بعدما نحدث الجذاذة الفهرس لننظر كيف يمكننا إستعمالها في البحث عن عنوان ما فتكون الخطوات كما يلي :

1. فتح الجذاذة القاعدة و الفهرس للقراءة.
2. قراءة مفتاح البحث : العنوان .
3. البحث في الجذاذة الفهرس عن رتبة التسجيل المذكور.
4. إن وجد ننقل الرتبة إلى الذاكرة .
5. الإشارة إليه بـ: seek .
6. نقل التسجيل إلى الذاكرة .
7. عرض محتوى التسجيل .
8. إذا أردنا الإعادة نكرر 2 إلى 7 .
9. غلق الجذاذتين .

3.11 إجراءات الإستحداث في النظام التتابعي المفهرس.

11.3 Mise a jour en séquentiel indexé.

إستحداث الجذاذة في النظام التتابعي المفهرس يشبه الإستحداث في النفاذ المباشر إلا أن

في التابعي المفهرس رتبة التسجيل توجد في المفهرس لا نبحث عنها في الجذاذة القاعدية
 . fichier de base

1.3.11 إجراء الإضافة .

11.3.1 L'ajout en séquentiel indexé.

* الخطوات :

1. فتح الجذاذة القاعدة و المفهرس للقراءة و إعلان النفاذ المباشر إن كان غير تلقائي .
2. قراءة المفتاح وهو عندنا عنوان الكتاب لإننا أحدثنا جذاذة فهرس على العنوان
3. البحث عنه في المفهرس : إن وجد ... نعلن أنه موجود .. لا داعية لإضافته..و إلا نطلب إدخال التسجيل الجديد .
4. وضع التسجيل في آخر الجذاذة القاعدة .
5. وضع رتبة التسجيل و العنوان في الجذاذة المفهرس.
6. إذا أردنا إدخال إضافات أخرى نكرر الخطوات 2 إلى 5 .
7. غلق الجذاذتين .

2.3.11 إجراء الحذف .

11.3.2 Procédure de suppression.

* خطوات الحذف المنطقي :

1. فتح الجذاذة المفهرس للكتابة و تعيين النفاذ المباشر .
2. قراءة مفتاح البحث .
3. البحث عن المفتاح في المفهرس : إن وجد نضع على الرتبة أو العنوان علامة الحذف
 marquer l'enregistrement. إن لم نجده نعلن عن الخطأ.
4. إذا أردنا الحذف من جديد نكرر الخطوات 2 و 3 .

5. غلق الجذاذة .

3.3.11 إجراء التغيير .

11.3.3 Procédure de modification.

* الخطوات :

1. تعيين النفاذ المباشر في الجذاذتين .
2. فتح الجذاذتين للقراءة أو الكتابة.
3. قراءة مفتاح البحث عن التسجيل المطلوب تغييره.
4. البحث عن التسجيل في الفهرس : إن وجد ننقل رتبته من الفهرس ثم نشير إلى التسجيل في الجذاذة القاعدة .
5. نقرأ التغيير من الدخل .
6. كتابة التغيير في المكان المشار إليه .
7. إذا أردنا تغييراً آخر نكرر الخطوات 3 إلى 6 .
8. غلق الجذاذتين .

** في الباب الموالي تجد طريقة أسرع لإحداث الفهارس على شكل قائمة خطية حركية :

Liste linéaire dynamique أنظر حل المسألة رقم 5.

الباب الرابع عشر: بنية البيانات الحركية.

Chapitre 14 : Structure de donnée dynamique. Dynamic Data structure.

1. التعريف.

1. Definition

فيما تقدم كنا نصرح بالبيانات في قسم التصريحات على شكل إعلان مسبق بأننا سوف نحتاج إلى تلك البيانات في البرنامج و الإجراءات و الدوال قبل أن يبدأ تنفيذ التعليمات من

قسم التعليمات في البرنامج الرئيسي , فكان الترجمان عند كل تصريح ببيان يحضر لنا الذاكرة المناسبة المعرفة بإسمها أي Identificateur/identifiant و محتواها sa valeur / its value غير معيّن إلى أن يبدأ تنفيذ التعليمات فتصبح لها قيم . هذه الطريقة تسمى البنية الساكنة structure statique /static structure لأنها معروفة عندما يُترجم البرنامج known at compile time فهي بني لا تنتظر التنفيذ لتُعرف . أما البنى الحركية dynamic data structure فلا تعرف كاملة إلا خلال التنفيذ at run time أي أن الترجمان لا يحدث البنية كاملة (لا يحضر الذاكرة المطلوبة) إلا عنوانها فقط الذي سوف يشير إليها.

ترجمان باسكال يستعمل تقنية لإدارة الذاكرة المركزية و إستعمالها Pascal storage management techniques نلخصها فيما يلي : التحضير الآني للذاكرات في المنطقة المسماة الكتلة المتراسة automatic - allocation in the run - time-stack والتحضير المتحكم فيه خلال التنفيذ من طرف المبرمج المسمى : manual - allocation in the Heap و هي كتلة في الذاكرة تتمدد حسب الطلب أي تسمح لك بتحضير العدد الذي تريد من الذاكرات لتنفيذ برنامجك على عكس الكتلة الأولى stack فهي غالبا ما تكون محددة مسبقا بـ: 64 kbyte .

2.الهدف من إستعمال البنية الحركية:

2.But des Structures Dynamiques.

من الفوائد المهمة في إستعمال البنى الحركية هي الجداول الحركية dynamic arrays حيث أن المبرمج لا يصرح بحدود الجدول إلا خلال التنفيذ و بالتالي لا يضع أماكن في الذاكرة كما كان يفعل عندما يصرح بالجدول في قسم التصريحات بعدد معين من العناصر و قد لا يستعملها كلها و هو ضياع للذاكرة.

الفائدة الثانية على سبيل المثال لو أردنا شحن محتوى جذاذة file إلى الذاكرة لنقوم بترتيبها أو أي معالجة أخرى بسرعة النفاذ إلى الذاكرة بدلا من سرعة النفاذ إلى القرص فالبنية

الديناميكية توفر لنا طرق عدة لشحن محتوى الجذاذة إلى الذاكرة المركزية .
 الفائدة الثالثة و ليست الأخيرة : لكتابة برامج عالية مثل الترجمات و محررا النصوص و
 الصفحات و الجداول و برامج تشغيل الحاسب ككل مثل إدارة الذاكرة المركزية و القرص لا
 بد لنا من استعمال البنية الديناميكية .

3.التصريح بالنوع الحركي : الأسهم .

3.Declaration du type dynamique:Les Pointeurs/pointers..

المتغير من نوع السهم هو عنوان لذاكرة من نوع بيان معين . A pointer type variable
 contains the memory adress of dynamic variable of a specifeid type .
 Syntax : type identifier = ^data_type ;

يعرف السهم للإشارة إلى أي نوع بيان كان . نوع البيان الذي يشير إليه السهم يسمى
 بمجال نوع السهم pointer's domain type . قيمة السهم لا يمكن أن تكون عنوانا لمتغير من
 نوع آخر . غالبا ما يكون نوع مجال السهم التسجيل record و الذي يصرح به مباشرة بعد
 التصريح بالسهم .

لنفاذ إلى القيمة التي يشير إليها السهم يجب كتابة إسم السهم متبوعة بالحركة ^ و هو
 ما يعرف بـ: dereferencing the pointer .

Semantics :

المتغير السهم يحتوي على عنوان متغير آخر في الذاكرة (الكتلة heap)

الثابت : Nil يتلائم مع جميع أنواع الأسهم . Nil Pointer type constante that does
 not point to anything. And is compatible with all pointer type .
 الكلمة الخاصة Nil تشبه القيمة الصفر (0) أو العنوان صفر , السهم الذي قيمته nil لا
 يشير إلى أي متغير .

الأسهم أعداد صحيحة ممثلة بكلمات ثنائية (binary word) لعنونة الذاكرة و بالتالي لا
 يمكن قراءتها أو كتابتها أو القيام بأي عمليات حسابية عليها .

يمكن تعيينها إلى متغير من نفس النوع ومقارنتها بالتساوي و الاختلاف فقط .

الأسهم المصرح بها مسبقا : **Predefined pointer Type** يوفر BPW

نوعان مصرح بها مسبقا و هي : Pointer و Pchar .

1: النوع Pointer: هذا السهم يعرف متغير من نوع السهم من دون نوع محدد أي مجال

تعريفه مفتوح . define an untyped pointer : pointer that does not point to any specified type.

2: النوع : Pchar و يعني :

Type Pchar = ^char ;

أمثلة : التصريح بنوع السهم :

```
Type IntegerPtr = ^ integer ;
    WordPtr = ^ word ;
    IdPtr = ^ IdRec ;
    IdRec = Record
    Nom_prenom : string[25];
    NuCompt : integer ;
    Next : IdPtr ;
End;
```

4. كيف نعين قيمة إلى المتغير السهم.

4.How to assign a value to a pointer value.

لإعطاء قيمة للمتغير السهم أي لكي يصبح يحمل عنوانا للمتغير الذي يشير إليه يوفر 7

BPW ثلاثة طرق :

* بإستعمال الإجراء NEW أو GetMem .

* والعامل @ : at .

* والدالة : function Ptr .

1.4 الإجراء الخاص بإحداث المتغير الحركي:

4.1 Procedure New.

Declaration : Procedure New (var P : Pointer [,init : constructor]);

الإجراء New يحدث متغيراً حركياً جديداً في الـ heap و ينقل عنوانه إلى المتغير السهم الذي يشير إليه . أي أنه يقوم بعملين : أولاً يحدث الذاكرة التي ستحمل البيانات في الـ heap و يعطي عنوان هذه الذاكرة إلى المتغير السهم الموجود في الـ stack إن كان المتغير السهم ساكناً أو في الـ heap: إن كان المتغير السهم حركياً , فالسهم الآن له قيمة .
 . creat a new dynamic variable and set a pointer variable to point to it .
 و قد أضيف إليه العمل على إعطاء قيمة ابتدائية للأشياء إذا قدم له الوسيط الثاني الباني للأشياء . كما أضيف إليه أن يعمل على شكل دالة ترجع قيمة من نوع سهم و يكون وسيطها نوع بيان السهم بدلا من المتغير السهم . New is extended to act as a function .
 returning a pointer value, the parameter passed to new is the type of the pointer to the object rather than the pointer variable itself.

2.4 الإجراء Getmem(var P: pointer ; size : word);

هذا الإجراء يشبه New و لكنه يضيف حجم الذاكرة التي سوف يحضرها أي يعطي للمبرمج حرية اختيار طول كلمة الذاكرة و هو أمر غير يسير لمن لا يتقن كيف تخزن البيانات في الذاكرة المركزية .
 . creat a dynamic variable of the specified size and puts the adress of the block in the pointer variable .
 لإستعمال هذا الإجراء بدون أخطاء يمكن الإستعانة بالدالة sizeof: التي ترجع عدد الكلمات الثنائية التي يتكوّن منها التسجيل (الوسيط).

SizeOf (function);

Returns the number of bytes occupied by the argument .

function SizeOf: Integer;

مثال 1:

type

CustRec = record

Name: string[3];

```

    Phone: string[14];
End;
var
    P: ^CustRec;
begin
    GetMem(P, SizeOf(CustRec) );
    Writeln ('The size of the record is ', SizeOf(CustRec) );
    FreeMem (P, SizeOf(CustRec) );
    Readln;
end .

```

مثال 2 :

```

program pointeroperations;
    uses WinCRT;
    type recPtr = ^rec;
           rec = record
               v1 : real;
v2 : string;
next : recPtr;
    end;
    var rec1,rec2,rec3 : recPtr;
    begin
        getmem( rec1 ,sizeof(rec);
        rec1^.v1 := 456;
rec1^.v2 := 'bonjour';
        rec2 := rec1 ;
        new( rec3) ;
        rec3^.v1 := 123456 ;
rec3^.V2 := 'Mouslim';
        writeln('rec1.v1:',rec1^.v1:5:2,' rec1.v2:',rec1^.v2);
        writeln('rec2.v1 :',rec2^.v1:5:2,' rec2.v2 : ',rec2^.v2);
        writeln('rec3.v1 ',rec3^.v1:15:2,' rec3.v2:',rec3^.v2) ;
        if rec1 = rec2 then writeln('rec1 = rec2');
        if rec1 <> rec3 then
            writeln('pointer rec1 <> Pointerrec3');
            new( rec1^.next ) ;
        rec3 := rec1^.next ;
    end

```

```

rec3^.v1 := 123456 ;
rec3^.V2 := 'Mouslim' ;
writeln('rec3.v1 ',rec3^.v1:15:2,' rec3.v2:',rec3^.v2');
freemem(rec1,sizeof(rec));
writeln('have disposed rec1') ;
writeln('rec1.v1:',rec1^.v1:5:2,' rec1.v2:',rec1^.v2);
writeln('rec2.v1 :',rec2^.v1:5:2,' rec2.v2 : ',rec2^.v2);
end.

```

3.4 العامل @ .

4.3The @ (at) operator

لإحداث سهم لمتغير حركي يمكن إستعمال هذا العامل الذي يلعب دور New في نقل عنوان أي ذاكرة ساكنة أو حركية إلى المتغير السهم .
مثال :

```

Program poinTop ;
Type   NumPoint = ^integer ;
      Var numberPoint : NumPoint ;
          Number : integer ;
Begin
  Writeln('here are two ways to manipulate a variable');
  Number := 5 ;{ initialize the variable }
  Writeln('Number =',number);
  NumberPoint := @ number ;{get's its adress }
  Writeln( 'numberPoint^:',numberPoint^ );
  Number := 10 ; writeln('number is now',number );
  writeln( 'numberPoint^:',numberPoint^ );
    {the stored value has changed}
  numberPoint^ := 20 ;
  writeln('we assigned a new value to NumPoint^,NumPoint^);
  {make the assignment different way }
  writeln('the stored value in Number is ',number);
    {yes the stored value has changed }
end.

```

5. كيف نتخلى عن المتغير الحركي

5. Désaffectation d'une variable dynamique : la procédure Dispose.

Procedure Dispose (var P : pointer [,destructor]) ;

هذا الإجراء يحذف المتغير السهم من الذاكرة فتصبح قيمة السهم غير معلومة . after a

call to dispose the value of P is undefined

أو الإجراء الآخر : Freemem(var P:pointer,size :word);

6. أمثلة :

كيف نحدث و نعين الأسهم :

Demonstrate pointer allocation and assignment.

Program pointers ;

Uses winCRT;

Type charPoint = ^ char ;

Var firstPtr , secondPtr , ThirdPtr : charPoint ;

Begin

{ allocate a char sized location to for each pointer to address }

new (firstPtr) ; new (secondPtr);

firstPtr^ := 'A' ; secondPtr := 'B' ;

{ give the location a value } thirdPtr := firstPtr ;

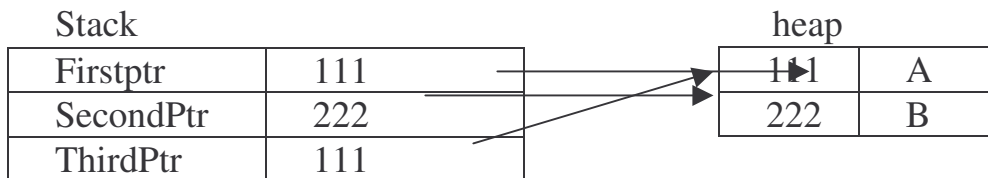
writeln(firstPtr); writeln(secondPtr); writeln(thirdPtr);

end.

لنرى كيف تتم عمليات الإحداث الحركية بـ: new . عندنا تصريح بنوع جديد من

نوع حركي charPoint بمعنى أننا سوف نحدث متغير يكون محتواه عنوان لمتغير من نوع char

. و هي التي صرحنا بها في الـ: var firstptr ,secondPtr,thirdPtr:charPoint;



التصريح بالـ: var يحضر لنا 3 ذاكرات من نوع سهم في الـ: stack و عند النداء للإجراء

new تحدث الذاكرة الحقيقية من نوع char في الـ heap و عنوانها يعطى للسهم الموجود في الـ stack لذلك نقول إننا نشير إلى الذاكرة بالسهم لأنها تملك عنوان الذاكرة . في مثالنا هنا أحدثنا الذاكرة 111 و أعطينا هذا العنوان إلى المتغير السهم firstPtr ثم أحدثنا 222 بـ new دائما و أعطينا هذا العنوان إلى secondPtr ثم عينا قيمة A للذاكرة في الـ heap بالإشارة إليها بـ firstPtr بالإضافة إلى الحركة ^ التي تسمح بالدخول إليها و كذلك إلى secondPtr عينا B ثم منحنا إلى ThirdPtr عنوان الأول و هكذا أصبح سهمان يشيران إلى ذاكرة واحدة .

كيف نحدث تسجيل حركي يحتوي على حقل سهم .

```
program pointerTocity ;{creat linked list }
  type cityLink = ^cityRecord;
    cityRecord = record
      city : string[25];
      next : cityLink ;
    end;
  var Headoflist , secondCity : cityLink ;
begin
  new ( headoflist );
  headoflist^.city := 'algers';
  new ( secondCity );
  secondCity^.city := 'Tokyo ';
  headoflist^.next := secondcity ;
  new ( secondCity^.next );
  secondCity^.next^.city := 'cairo';
  secondCity^.next^.next := nil ;
  ...
```

كيف نتخلى (نحذف) المتغير الحركي .

```
Program    nodePointer ;{how to dispose pointers}
  type NodePOINT = ^NodeTYPE;
    NodeTYPE = record
      Next: NodePOINT;
```



```

        Data: integer
    end;

procedure FreeList (var Head: NodePOINT);
var TempPtr: NodePOINT;
begin
    TempPtr := Head;    { Initialize TempPtr }
    while Head <> nil do
        begin
            Head := Head^.Next; { Advance the list head }
            dispose (TempPtr); { Dispose of the previous node }
            TempPtr := Head;    { Reinitialize TempPt }
        end
    end;
    { FreeList }

```

العمليات الممكنة على الأسهم و المتغيرات المشار إليها بالأسهم .

4.) program pointeroperations ;

```

    { global operation on pointers }
    uses Wincrt ;
    type rec = record
        v1 : real;
        v2 : string;
    end ;
    var rec1,rec2,rec3 : ^rec;
    begin
        new( rec1 );
        rec1^.v1 := 456 ;rec1^.v2 := 'bonjour ' ;
        rec2 := rec1 ;
        new( rec3 ) ;
        rec3^.v1 := 123456; rec3^.V2 := 'essalam';
        writeln(rec1^.v1,rec1^.v2,rec2^.v1:15:2,rec2^.v2);
        writeln(rec3^.v1:15,rec3^.v2);
        if rec1 = rec2 then writeln('rec1 = rec2');
        if rec1 <> rec3 then
            writeln('pointer rec1 <> Pointerrec3');
        end.

```

لنرى ماذا يحدث عند تقديم الوسيط من نوع السهم بالقيمة في إجراء أو دالة .

5.){Demonstrates some effects of passing a pointer as a value parameter. }

```

program Passage;
    uses winCRT;
type NodePOINT = ^ NodeTYPE;
    NodeTYPE = record
        Next: NodePOINT;
        Data: char;
    End;
var CurrentPtr: NodePOINT;
procedure Change (TempPtr: NodePoint);
begin
    TempPtr^.Data := 'C';           { Which of these are  }
    TempPtr := TempPtr^.Next;      { local assignments }
    TempPtr^.Data := 'D';
End;
begin
    new (CurrentPtr);
    CurrentPtr^.Data := 'A';
    new (CurrentPtr^.Next);
    CurrentPtr^.Next^.Data := 'B';
    write (CurrentPtr^.Data); write (CurrentPtr^.Next^.Data);
    Change (CurrentPtr);
    write (CurrentPtr^.Data); write (CurrentPtr^.Next^.Data);
end. {Passage}

```

7. القائمة الخطية .

7.The data type linear list

القائمة الخطية تشبه المصفوفة من بعد واحد مع الفرق أنها غير محددة بعدد معين من

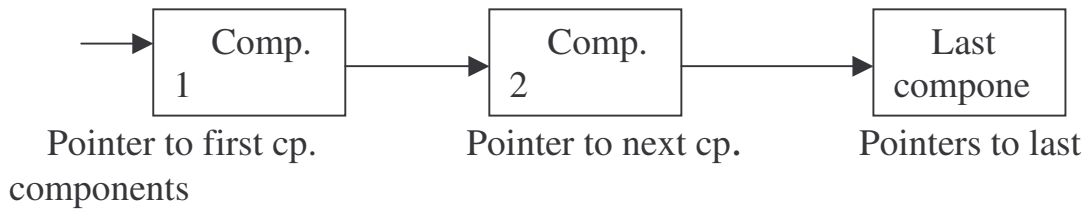
البيانات أي لا نصرح بمجالها من ماذا إلى ماذا [1..n] فهي إذا قائمة من صفر أو أكثر
عنصر و سهم إلى العنصر الأول (و قد يكون nil إذا كانت القائمة لا تحتوي على أي عنصر)
من ميزاتها كذلك أنها تسمح بالدمج insertion و الحذف deletion بسهولة . توجد أنواع
مختلفة للقائمة الخطية و لكن العامل المشترك بينها هو : يوجد ترتيب طبيعي بين عناصر القائمة

, بمعنى أن لكل عنصر تابع a next component و سابق a previous component (إلا الأخير أو الأول) .

1.7 القائمة الخطية الأحادية .

7.1 Liste liniaire simple / singly linked lists

القائمة الخطية الأحادية تشبه الجذاذة المتتابعة sequential file



ككل بنية بيانات نستطيع معالجتها بشتى الإجراءات و أهمها : إحداث القائمة الخطية creating a singly linked list , كتابتها على الشاشة أو الطابعة أو جذاذة على القرص , finding a given element البحث عن بيان ما فيها , writing it on any type of file إدماج عنصر جديد فيها inserting a given element , حذف عنصر منها deleting a given element , ترتيبها حسب مفتاح ما sorting ... إلى آخره .
و إليك مثالا لإحداث قائمة خطية , عرضها على الشاشة , حذفها .

```
Program linkedlist;
uses wincrt;
type listPtr = ^list;
list = record
    data : string;
    next : listPtr;
end;
var LP, tp, firstP : listPtr;
n : string;
```

إحداث قائمة خطية (من عنصر واحد على الأقل)

يكون النفاذ إليها بالسهم firstP الذي نجد فيه عنوان رأس القائمة

* الخوارزمية Algorithm :

1. إحداث أول عنصر بالإجراء New(FirstP) .

حيث أن () New كما قلنا تحدث الذاكرة في الـ heap و تقدم عنوانها إلى المتغير السهم firstP الموجود بالـ stack. و بالتالي فإننا نحتفظ بالعنوان الأول في firstP .

نملأ الحقول بقراءة البيانات من الدخل بالتعين إليها , في مثالنا عندنا حقل واحد : data
كما يلي : readln(firstP^.data);

إنتهينا من العنصر الأول , إذا أردنا إحداث عنصر ثاني نواصل العملية و إلا نعيّن في الحقل الثاني المسمى next من نوع سهم listPtr القيمة nil و التي تسمح لنا بغلق القائمة الخطية .

إذا أردنا إضافة العنصر الموالي الذي نوّد ربطه بالأول حيث أحدثنا له حقل لتخزين عنوانه next ما علينا إلا أن نتابع نفس الطريقة في الإحداث بالإجراء new() و تكون كالتالي :
(new(Lp^.next) بعدما إحتفظنا بعنوان أول عنصر في السهم firstP: نواصل الإحداث بسهم آخر مثل : Lp .

نواصل الخطوات بإدخال البيانات للحقول : عندنا حقل واحد : data: إذا تكون :
(readln(Lp^.next^.data) و لتجنب كتابة Lp^.next^ نستعمل lp كسهم يساوي lp^.next .
و هكذا نواصل الإحداث من الدخل , نقرأ البيانات و تكون القائمة الخطية بالذاكرة المركزية بالطول الذي نريد : إذا سنكرر الخطوات 5 و 6 إلى أن تنتهي البيانات المراد إدخالها و ذلك بطرح سؤال على المستعمل كل مرة : هل تريد إضافة عنصر : نعم أم لا ؟ فإن أجاب بنعم نكرر وإلا ننهي الإحداث و نغلق القائمة بوضع القيمة nil في آخر next :
lp^.next := nil

إليك مثال إحداث قائمة خطية في الـ heap : عندنا تصريحاً بنوع جديد اسمه LstPtr من نوع سهم List^ و هو بدوره تسجيل صرحنا به من حقلين : data و next . ثم صرحنا بمتغير سهم firstP و Lp في الـ stack حيث يحضّر لنا الترجمان ذاكرتان firstP و Lp لتخزين

العنوان الذي يسمح لنا بالإشارة إلى التسجيلات . بعدما نكتب على الشاشة تعليق : إحداث القائمة . نطلب إحداث أول تسجيل حركي dynamic record بـ: new(firstP) التي تقوم بإحداث تسجيل في الـ: heap و تعطي عنوانه للمتغير السهم firstPfirstP مثلاً عنوانها 11 سيعطى للذاكرة firstP و هو الآن عنوان أول عنصر في القائمة سنحتفظ به لحاجتنا إليه من بعد في عمليات النفاذ إلى القائمة . و نبدأ بإدخال البيانات إلى حقول التسجيل المحدث في الـ: Heap و المشار إليه بـ: firstP كالتالي : readln(firstP^data) بعدما كتبنا تعليقا على الشاشة لتنبه المستعمل لإدخال البيان ? Data .
مثلاً لندخل القيمة : "علي" في الحقل data .

FirstP	111	Adress	Data	next
Lp	555	111	علي	222
		222	محمد	333
		333	فاطمة	444
		444	عائشة	555
		555	عمر	nil

و نواصل إحداث القائمة بالسهم Lp الذي أصبح الآن عنده قيمة حيث نقلنا إليه قيمة firstP ليشير إلى نفس الذاكرة الأولى . الآن كيف نحدث العنصر الموالي و نربطه بالأول : بما أن الحقل next للتسجيل من نوع سهم أي أنه سيسمح لنا بإحداث عنصر جديد إذا كتبنا : new(Lp^next) حيث أن الإجراء new تحدث ذاكرة في الـ: heap من نوع تسجيل و تضع

عنوانها في الحقل next مثلا عنوانها 222 . و لكي نستعمل المتغير السهم lp كسهم للإشارة إلى هذا العنصر الجديد نحول Lp^{next} إلى Lp ليكون فيها الآن 222 بدلا من 111 . و نواصل بإدخال البيان إلى الحقل data بـ: $readln(Lp^{data})$ مثلا ندخل القيمة : محمد, و بالطبع نكرر هذه العملية حسب إختيار المستعمل . و في النهاية نغلق القائمة بوضع القيمة nil في الحقل Lp^{next} . في المثال الموالي أحدثنا قائمة من 5 عناصر : متتالية الأول يوصل إلى الثاني و الثاني يوصل إلى الثالث و الثالث يوصل إلى الرابع و الرابع مربوط بالخامس و الخامس لا يوصل إلى شيء nil . و السهم Lp في الأخير يشير إلى الأخير أي فيه العنوان 555 و بالتالي فالقائمة الخطية أحدثت و لا يمكن النفاذ إليها إلا بالسهم firstP الذي به عنوان الأول . لذلك نقول أنها قائمة FIFO: first in first out أول داخل أول خارج.

```

procedure creatlist;
begin
    writeln('Creation de la liste');
    { le pointeur sur le premier est dans firstP }
    new ( FirstP);
    write('data ?:');readln(FirstP^.data);
    و نواصل إحداث القائمة بالسهم LP
    LP := FirstP;
    writeln('Another Node [y]es or [n]o');
    while readkey In ['y','Y'] do
        begin
            new ( LP^.next);
            LP := LP^.next;
            write('data ?:');readln(LP^.data);
            writeln('Another Node [y]es or [n]o');
        end;{ while }
        lp^.next := nil;
    { le dernier est marqué par nil c.a.d ne pointe sur rien }
    end;{ creatlist }

```

إجراء عرض محتوى القائمة .

*. الخوارزمية : algo..

1. للدخول إلى القائمة الخطية في حالتنا هذه ليس لنا إلا الرأس firstP نضعه في متغير سهم

. LP

نكتب قيمة الحقل Lp^.data

نتقدم إلى العنصر الموالي بـ Lp^.next أي وضعنا العنوان الموالي في المتغير السهم

Lp لكي يشير إليه و نستطيع النفاذ إليه بسهولة

و هكذا نواصل كتابة محتوى الحقل data لكل عنصر إلى أن نصل إلى نهاية القائمة و علامت

نهايتها هي Nil: أي LP = nil

```

Procedure ecritlist;
begin
  writeln('listing of the linked list ');
  lp := FirstP;
  while lp <> Nil do
  begin
    writeln(Lp^.data);
    lp := Lp^.next;
  end;
End;

```

إجراء حذف القائمة.

*الخوارزمية :

1. نبدأ بأخذ عنوان رأس القائمة الذي خزن في السهم FirstP .

2. نضع هذا العنوان في متغير سهم Lp

3. نحتفظ بعنوان العنصر الموالي الموجود في الحقل next للعنصر الأول في firstP أي كي

يبقى عندي دائما في السهم firstP عنوان العنصر الأعلى في القائمة أي الأول .

4. نحذف الآن العنصر المشار إليه بالسهم Lp بالإجراء dispose

نضع في السهم Lp عنوان العنصر الموالي و الذي نجده في firstP
 نكرر الخطوات 3 و 4 و 5 إلى أن نصل إلى نهاية القائمة أي إلى أن يصبح Lp = nil .
 هذه الخطوة الأخيرة تقتضي إستعمال التعليمة Until...repeat و هي تعليمة تكرر على
 الأقل مرة واحدة و نحن سنستعمل تعليمة الـ: while للإحتياط المطلوب حيث أننا لا نعلم في
 البداية إن كانت القائمة فارغة أم لا لذلك وجب أن نختبر قبل البدء إن كان السهم الأول
 فارغا لا نقم بأي حذف و إن كان به عناصر نبدأ بحذف الأول .. إلى آخره...
 * الإجراء :

```

Procedure freelist;
begin
  lp := firstP;
  writeln(' disposing all the list');
  while lp <> nil do
    begin
      firstP := lp^.next;
      dispose ( lp );
      lp := firstP;
    end;
  writeln('list disposed ');
end;
```

إستعمالات القائمة الخطية كثيرة منها : دمج عنصر جديد فيها insert و ترتيبها sort و
 الإضافة من الرأس أو الخلف و البحث عن عنصر ما بمعرفة حقل ما منه و إليك الإجراءات
 التالية :

*إجراء إضافة عنصر من نهاية القائمة.

*الخوارزمية :

1. للإضافة إلى نهاية القائمة وجب أن نذهب إلى آخر عنصر و كما قلنا النفاذ في القائمة
 الخطية يكون من رأسها فقط , لذلك وجب أن نأخذ العنوان الموجود في FirstP لنذهب إلى
 الأخير بتعليمة While Tp <> Nil نتقدم في القائمة بـ: TP := TP^.next , نحتفظ بآخر عنوان

في Lp . فإذا وصلنا إلى نهاية القائمة كان عندنا في Lp عنوان الأخير , الذي في حقله next نجد nil , ما علينا الآن إلا أن نحدث العنصر الجديد بـ: new(LP^.next) و نعين البيان في حقله data , و في حقله next نضع nil . و نكون قد اظفنا عنصرا جديدا في النهاية .

```

procedure addRecord ;
begin
  Tp := firstP ;
  while Tp <> Nil do
    BEGIN
      Lp := Tp ;
      Tp := Tp^.next;
    end;
    New ( Lp^.next ); Lp := Lp^.next ;
    write(' add new data :'); readln(Lp^.data );
    Lp^.next := nil ;
  end;

```

إجراء البحث عن عنصر في القائمة

بمعرفة حقل ما وإرجاع عنوانه.

الخوارزمية :

1. نريد البحث على عنصر بمعرفة حقل ما : كرقم أو إسم مثلا عندنا حقل من نوع string يمكن أن نخزن فيه أسماء أو أرقام فإذا أردنا البحث عن عنصر ما مثل : "محمد" في القائمة نقرأ أولا البيان المطلوب البحث عنه . ثم نأخذ رأس القائمة في متغير سهم TP .
2. نقارن إذا كان البيان X = بالحقل الأول Tp^.data إذا فقد وجدنا العنصر نضع عنوانه في المتغير LP وإلا نواصل البحث بالتقدم في القائمة : Tp := Tp^.next .
3. نكرر الخطوة 2 ما دُمنا لم نصل إلى نهاية القائمة أو وجدنا البيان المطلوب , باستعمال متغير منطقي Trouve .

إن لم نجده نضع في المتغير السهم Lp القيمة nil .

يمكن تحويل الإجراء إلى دالة نرجع فيها عنوان العنصر المطلوب البحث عنه كالتالي :

function Find : ListPtr ;

```

procedure find ;
  var x : string ; trouve : boolean ;
  begin
    write('which data to find ? :');  readln( x );
    Tp := FirstP ; trouve := false ;
    while ( Tp <> nil ) and ( not trouve ) do
      begin
        if Tp^.data = x Then
          begin
            trouve := true;    Lp := Tp ;
          end
        else
          Tp := Tp^.next ;
        end;
      if not trouve then Lp := nil ;
    end;
  end;

```

إجراء دمج عنصر ما في القائمة بعد عنصر ما نجده بمعرفة حقل ما بالإجراء السابق.

*الخوارزمية :

1. نقوم بالبحث عن العنصر المطلوب الدمج بعده .

إذا وجدناه سيكون عندنا في Lp السهم الذي يشير إليه.

نحتفظ بالعنوان الموالي Lp^.next الذي يشير إلى العنصر التابع له.

نحدث العنصر الجديد الذي نريد دمجه في المكان Lp^.next حيث يصبح العنصر التابع

للعنصر الذي بحثنا عنه و عنوانه : Lp .

نقرأ حقول التسجيل عندنا واحد : readln(LP^.data);

ثم نعيّن في الحقل next العنوان السابق الذي إحتفظنا به حتى تصبح القائمة متسلسلة .

```

procedure insert ;
  begin
    find ;
    if Lp <> nil then
      begin
        Tp := Lp^.next ;

```

```

new ( Lp^.next ); Lp := Lp^.next ;
writeln('data 2 insert ? :'); readln( Lp^.data );
Lp^.next := Tp ;
end
else
writeln('element non trouve');
end;

```

يمكن إستعمال جذادة **file** دخل البيانات من نوع نص **text** لإحداث القائمة :نقرأ منها
 البيانات التي نريد إعطائها للحقل أو الحقول في التسجيل و هو عندنا الحقل Data . نصرح
 بجذادة نص للدخل **fin** و أخرى للخروج **fout** .
 الخطوات تبقى نفسها ما و نضيف إليها :
 1. ربط الجذادة المنطقية بالفيزيائية بـ: `assign(fin,'finlist.pas');`
 2. فتح الجذادة للقراءة منها حيث نكون قد خزننا فيها البيانات مسبقا بـ: `reset(fin);`
 و ما علينا إلا أن نقرأ البيانات بنفس الطريقة و لكن نضيف في تعليمة القراءة الوسيط **fin**
 كما يلي : `readln(fin ,FirstP^.dat);` للأول و `readln(fin,Lp^.data);`
 و لكن هنا نقرأ إلى نهاية الجذادة **fin** و هي قراءة دفعة واحدة **batch processing** غير
 القراءة التخاطبية **conversationnel** من لوحة المفاتيح
 و في الأخير نغلق الجذادة بـ: `close(fin)` .

```

program linkedlist ;
uses wincrt;
type listPtr = ^list ;
list = record
  data : integer ;
  next : listPtr;
end;
var LP,tp ,firstP,tt : listPtr;
n : integer ;
fin , fout : text ;

```

*إجراء الأحداث بجذادة نص .

```

procedure creatlist_fin ;
begin
  reset( fin );
  if not eof( fin ) then
    Begin
      { le pointeur sur le premier est dans firstP }
      new ( FirstP );
      readln(fin , FirstP^.data );
      LP := FirstP ;
    end
    else firstP := nil ;
  { s'il n y a pas de composant dans fin firstP = nil }
  while not eof ( fin ) do
    begin
      new ( LP^.next );
      lp := lp^.next ;
      readln(fin ,lp^.data );
    end;
    lp^.next := nil ;
    writeln('list created ');
  { le dernier est marqué par nil c.a.d ne pointe sur rien }
end;

```

* إجراء الإضافة من الرأس:

* الخوارزمية: بما أننا إحتفظنا برأس القائمة إذا سننفذ إليها من الرأس:

1. نحدث العنصر الجديد بـ: New(Tp) في الـ heap .

2. نملأ الحقول: أي قراءة الحقول data عندنا .

3. ثم الآن بقي الحقول next نعطيه العنوان firstP رأس القائمة قبل هذا الجديد . و يصبح

عنوان رأس القائمة الجديد هو الذي في Tp نعينه إلى FirstP .

```

procedure addrecATop ;
begin
  New ( Tp );
  write(' add new data :');
  readln(Tp^.data );
  Tp^.next := FirstP ;

```

```
FirstP := TP ;
end;
```

*إجراء الكتابة على جذادة في القرص:

*الخوارزمية : هنا عكس الأحداث من جذادة نص إذ قرأنا من الجذادة النص البيانات و أدخلناها في القائمة , سنقوم بتخزين تلك البيانات في جذادة نص على القرص , بمعنى سنقوم بإخراجها من الذاكرة المركزية على وسيط تخزين.

نفتح الجذادة للكتابة عليها بـ: `rewrite(fout)`

ندخل إلى القائمة من رأسها : `firstp := Lp` .

نكتب على الجذادة أول عنصر : هنا نكتب حقل واحد من التسجيل : `Lp^.data` .

نتقدم في القائمة : نعطي العنوان الموالي للسهم `Lp` الذي يشير إلى العناصر الواحد تلو الآخر .

نكرر الخطوات `repeat 3` و `4` إلى نهاية القائمة أي إذا أصبح `until Lp=nil` أو نقول نكرر

الخطوات `3` و `4` ما دام `while Lp<>nil` السهم

في النهاية نغلق الجذادة `fout`.

```
procedure ecritlist_fout ;
begin
  rewrite( fout );
  lp := FirstP ;
  while lp <> Nil do
    begin
      writeln(fout , Lp^.data);
      lp := Lp^.next ;
    end;
  close( fout );
end;
```

* إجراء حذف عنصر ما بمعرفة حقل مامثل `:data`

*الخوارزمية : هنا سنقوم بالبحث عن عنصر ما بمعرفة حقل ما , فإذا وجدنا ذلك

العنصر نقوم بحذفه إذا كان الأول وإلا سنحذف الموالي له:

نبدأ بقراءة البيان المراد البحث عنه و الذي به نجد العنصر .

الحالة الأولى إذا كان هذا البيان للعنصر الأول إذا سنقوم بتبديل عنوان العنصر الأول

بالذي يليه : $firstP := FirstP^{next}$

و إلا نأخذ العنوان الأول في سهم آخر مثل TP و نبدأ البحث في القائمة: إذا وجدناه نعطي

للحقل $next$ عنوان الموالي $TP^{next^{next}}$ و هو عنوان الموالي للموالي أي قفزنا فوق الموالي

: حذفناه من القائمة .

```

procedure delete ;
  var x : integer ;
      found : boolean ;
begin
  found := false ;
  write('give data of the component to delete :');
  readln( x );
  if firstP^.data = x then
    begin
      firstP:= firstP^.next;
      found := true ;
    end
  else
    begin
      tp := firstP ;
      while ( not found ) and ( tp <> nil ) do
        begin
          if tp^.data = x then
            begin
              found := true ;
              tp^.next := tp^.next^.next ;
            end
          else
            tp := tp^.next ;
          end;
        end;
      end;
    end; {delete}

```

*إجراء ترتيب القائمة على حقل ما مثل data

*الخوارزمية: نستعمل طريقة الترتيب بالتبادل أو المعروف بالترتيب على شكل كويرات

الصابون bubble sort التي تطلع إلى السطح الواحد تلو الأخرى.

نبدأ بأخذ عنوان العنصر الأول من FirstP في tp .

نقارن بيان الأول مع بيان الثاني و نقوم بالتبادل إذا كان الأول أكبر من الثاني .

نتقدم في القائمة بـ: $Tp := Tp^{next}$

نتابع المقارنة مثنى مثنى م عملية التبادل كي يصعد الأصغر و يتزل الأكبر ما دمنا لم نصل إلى

نهاية القائمة.

و نكرر الخطوات 1 و 2 و 3 و 4 إلى أن تصبح القائمة مرتبة و علامة ذلك إذا لم يتم أي

تبادل في الدورة السابقة .

```

procedure trlist ;
  var  trie : boolean ; ech : integer ;
  begin
    tp := firstP ;
    repeat
      trie := true;
      while tp <> Nil do
        begin
          if tp^.next <> nil then
            if tp^.data > tp^.next^.data then
              begin
                ech := tp^.data ;
                tp^.data := Tp^.next^.data ;
                Tp^.next^.data := ech ;
                trie := false ;
              end ;
            tp := tp^.next ;
          end;
        tp := firstP ;
      until trie ;
    end;
  end;

```

```

begin {prg listFIFO}
  assign( fin , 'finlist.pas');
  assign(fout , 'foutlist.pas');
  repeat
    clrscr;
    writeln('mémoire dispo :',memAvail:6);
writeln('max mem available :',maxAvail:6);
writeln('[c]réation de linked list a partir d'un fichier Text');
    writeln('[L]isting de la liste');
    writeln('[F]ree la list ');
    writeln('[e]crit sur fichier externe');
    writeln('[i],[I]nsert a new component after a given one ');
    writeln('[d]elete any component found by the data to give ');
    writeln('[a]dd new record at end');
    writeln('[T]dd new record at top') ;
    writeln('[s]ort list ');
    writeln('[B]reak ');
  case readkey of
    'c','C' : creatlist_fin ;
    'L','l' : ecritlist ;
    'f','F' : freelist ;
    'E','e' : ecritlist_fout ;
    'i','I' : insert ;
    'd','D' : delete ;
    'a','A' : addRecord ;
    'T','t' : addRecATop ;
    's','S' : trilst ;
    'b','B' : break ;
    else writeln('choix inconnu');
  end;{ case }
  writeln('autre execution [o]ui ou [n]on ');
  until readkey in ['n','N'];
  writeln('fin au revoir ');
end.

```

*الخلاصة : حاولنا عرض أغلب العمليات على القائمة الخطية لنبين أنها بنية مرنة تسمح

بكل العمليات و مفيدة جدا للمبرمج حيث أنها لا تحدد له عدد التسجيلات كم هو حال

المصفوفة array's و تتم كل العمليات بالذاكرة المركزية فتكون معالجتها أسرع من الجذاذات files و بالتالي فهي بنية المبرمج الفطن و المتقدم advanced programmer . من أجل تسهيل الفهم كتبنا جل الإجراءات بدون وسيط و على الطالب أن يعيد كتابتها بإستعمال الوسيط المقدم كقيمة value parameter و بالوسيط المقدم كمتغير variable parameter و إستعمال الدوال function بدلا من الإجراءات كذلك .

2.7 بنية البيانات : الكومة.

Data type Stack(pile)

*تعريف الكومة stack: هو قائمة خطية أين تكون جميع العمليات من دخل و إضافة و إدماج و حذف وإخراج من بداية القائمة فقط . باختصار تسمى بالأول الداخل الآخر الخارج

First In Last Out(FILO):

عند إستعمالنا لبنية الكومة يجب أن تسمح لنا بالعمليات التالية :

*الإحداث $\text{creat}(s)$: creation of a stack S

*إضافة عنصر إلى الكومة: $\text{Push}(s)$: adding an element x to a stack S

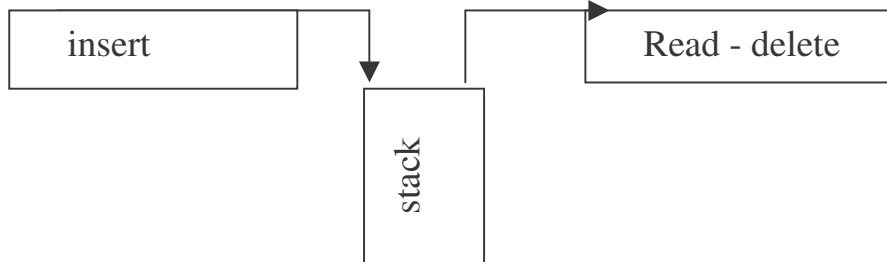
*قراءة أول عنصر من الكومة: $\text{Top}(s)$: reading the first element of a stack s

*حذف عنصر من الكومة: $\text{pop}(s)$: deleting an element a stack s

*إختبار هل الكومة فارغة: $\text{empty}(s)$: testing whether a stack is empty

هذه العمليات تتم على شكل دوال functions فالإحداث $\text{creat}(s)$ دالة و الإضافة

$\text{push}(x,s)$ دالة كذلك ... إلى آخره..



و بالتالي الخصائص التالية دائما صحيحة في حق الكومة :

must always hold :

*كل كومة نحدثها الآن تكون فارغة: $\text{creat}(s)$ is empty: $\text{Empty}(\text{creat}(s)) = \text{true}$.

*تكون الكومة غير فارغة بعد إضافة عنصر إليها: A stack to which an element is

added is not empty: $\text{Empty}(\text{push}(x, s)) = \text{false}$

*قراءة أو حذف عنصر من كومة أحدثت حالا ينتج خطأ. reading and deleting an

element from a newly created stack result in error: $\text{pop}(\text{creat}(s)) = \text{error}$ and $\text{Top}(\text{creat}(s)) = \text{error}$

*إضافة عنصر إلى الكومة و قراءته منها مباشرة تكون نتيجه ذلك العنصر. adding an

element to a stack and immediatly thereafter reading the first element yields that element : $\text{Top}(\text{push}(x, s)) = x$

Function	Implementation
$\text{Creat}(s)$	Var tops , temps : StackPt ; Tops := nil ;
$\text{Push}(x, S)$	New(temps); temps^.field := x; Tems^.last := tops ; tops := temps;
$\text{Pop}(S)$	If tops = nil then writeln('error ;stack empty ; no pop ') Else tops := tops^.last ;
$\text{Top}(s)$	If tops <> nil then writeln(Tops^.field) else writeln('error');
$\text{Empty}(s)$	Test if Top(s) = nil

في هذه الطريقة تلاحظ أن جميع العناصر تضاف قبل العنصر الأول

inserted BEFORE the first element

و إليك الآن جميع هذه العمليات مكتوبة على شكل إجراءات و ما عليك إلا إعادة كتابتها

كدوال :

```

program Stack ;
  uses wincrt;
  type idPtr = ^Id ;
  id = record
    data : string[25] ;
    last : idPtr ;
  end;
  var node ,Tp, TopS: idPtr;

```

*.خوارزمية إجراء الإضافة: Push

1.إحداثا العنصر بـ: new(node)

2.ملاً الحقول : عندنا حقل واحد data بالقراءة;readln(node^.data)

3.و في الحقل node^.last نضع العنوان السابق الموجود في tops

4. و نضع العنوان الجديد في tops حتى يصبح هو العنوان الأخير الذي يشير إلى آخر

عنصر دخل إلى الكومة.

```

procedure push ;
begin { Add Item to the stack. }
  new( node );{ Create and fill a new node... }
  write('data :'); readln( node^.data );
  node^.last := topS ;
  TopS := node;
  { ...then advance the top pointer. }
end;

```

*خوارزمية الإحداثا :

نبدأ بوضع القيمة nil في tops للدليل على أن الكومة فارغة .

ثم نبدأ بإدخال الإضافات المسماة إدفع push لأننا نقوم بدفع العناصر داخل الكومة الواحد تلو الآخر : الأول فوقه الثاني و هكذا تتراكم فيها العناصر الأول من دخل يكون آخر من سيخرج .

```

procedure creatstack ;
begin

```

```

writeln('debut de creation list ');
topS := nil ;
writeln('U want to creat a stack [y]es or [n]o');
while readkey in ['y','Y'] do
  begin
    push ;
    write('another node yes or no ');
  end;
end;
end;

```

*خوارزمية : هل الكومة فارغة :

1. نختبر هل السهم tops توجد به قيمة الـ nil فإذا كانت كذلك تأخذ الدالة empty القيمة true وإلا false .

```

function empty : boolean ;
begin
  if tops = nil then
    empty := true else empty := false ;
  end;
end;

```

الخوارزمية قراءة العنصر الأول من الكومة :

و هي عملية كتابة على الشاشة للحقل أو الحقول من أول عنصر المشار إليه بـ: Tops

```

procedure tOp;
begin
  if tops <> nil then writeln('data is:',topS^.data)
    else writeln('error stack empty ');
  end;
end;

```

*خوارزمية عرض محتوى الكومة على الشاشة :

1. نأخذ عنوان أول عنصر من السهم tops في متغير آخر node

2. نكتب محتوى الحقل node^.data .

3. نتقدم في الكومة إلى العنصر الموالي تحت الذي سبق بأخذ عنوانه من : node^.last في

node لتشير إليه .

4. نكرر الخطوات 2 و 3 إلى أن يصبح السهم node=nil أو نقول نكرر 2 و 3 ما دام المتغير node <> nil ليتمكن إستعمال التعليمة التكرارية while بدلا من repeat للإحتياط في حالة ما تكون الكومة فارغة لا نقوم بأي نفاذ و نتجنب الخطأ .

```
procedure viewStack ;
begin
  writeln('Here is the stack content :');
  node := TopS ;
  while node <> nil do
  begin
    writeln('data : ',node^.data );
    node := node^.last ;
  end;
  writeln('DooooooooooooooooNnnnnnnnnneeeeeee');
end;
```

*خوارزمية حذف العنصر الأعلى :

1. نأخذ عنوان العنصر الأعلى من tops في node .
2. نأخذ عنوان الأسفل من tops في tops حيث لو حذفنا مباشرة الأول فسنقوم بحذف كل المتغير و هو يحتوي على عنوان الذي يليه في last و بعد ذلك لا يمكننا النفاذ إلى العنصر الأعلى بما أننا لا نملك عنوانه :لذلك لا بد من الإحتفاظ بالعنوان الموجود في node^.last في tops .

3. و في الآخر نحذف العنصر المشار إليه بـ: node .

```
procedure pop ;
{ Return the stack's top item, then remove its node. }
begin
  if empty then writeln('error')
  else
  begin
    writeln('data of Item popped : ', TopS^.data );
    node := tops ;
    { Save a pointer to the old top node... }
    Tops := node^.last ;
  end;
```

```

    { ...point it to the current top... }
    { ...so that it can be disposed of later. }
    dispose ( node );
end;
end;

```

*خوارزمية حذف كل الكومة :

1. و هي عملية حذف الأعلى إلى نهاية الكومة ما دام tops<>nil فهي نداء للإجراء السابق في تكرار ما دام tops<>nil .

```

procedure popAll ;
{ Remove the stack, leaving the TopPtr node nil. }
begin
    while tops <> nil do
        pop ;
        writeln('stack disposed');
    end;
begin{prg stck}
    repeat
        clrscr ;
        writeln('>>[c]reat stack');
        writeln('>>[v]iew stack');
        writeln('>>[a]dd New node');
        writeln('>>[p]}op : show data of top node and remove it ');
        writeln('>>[d]elete all nodes ');
        writeln('>>[r]eading the first element of astack');
        writeln('>>[e]mpty stack True or False ?');
        writeln('>>[B]reak ');
        case readkey of
            'c','C' : creatStack ;
            'a','A' : push ;
            'v','V' : ViewStack ;
            'p','P' : pop;
            'd','D' : PopAll ;
            'r','R' : top ;
            'e','E' : writeln('stack empty ?:',Empty );
            'b','B' : break ;
            else writeln('unknown Operation');
        end;
    until readkey = 'B';
end;

```

```

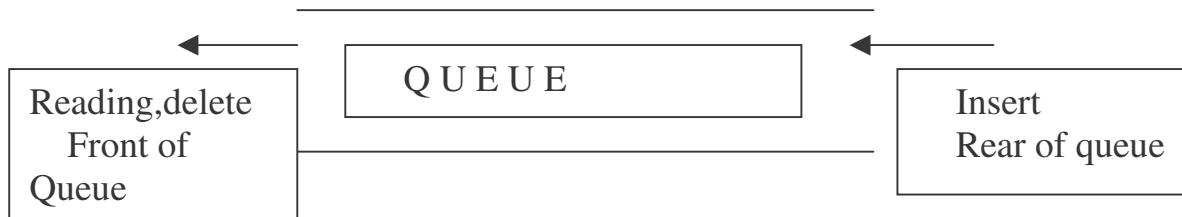
end;
writeln('another Run');
until readkey in ['n','N'];
write('FiiiiiiiiiiiiNNNNNNNNNNNNNN');
end. {fin stack operation}

```

3.7 بنية الطوابير .

The data Type Queue.(file)

1. التعريف : هذا النوع من القائمة الخطية يضع قاعدة في أن تكون جميع عمليات الدمج insertions من جهة واحدة : من الأخير و جميع عمليات الحذف و القراءة من الجهة الأخرى : من الرأس فنقول إنها من نوع أول داخل أول خارج (FiFo) First In First Out .



هذه البنية تسمح بالعمليات التالية :

1. الإحداث Queue Q : creat(q)

2. إضافة عنصر لها : Append(x,q);

3. حذف عنصر منها : Delfront(q);

4. قراءة رأس الطابور : Front(q);

5. اختبار هل هي فارغة : Empty(q);

و بالتالي للطابور الخصائص التالية:

1. كل طابور أحدثناه حالا فارغ : Empty(creat(q)) = true;

2. إذا أضفنا عنصر فهي غير فارغة: Empty(Append(x,q)) = false;

3. قراءة أو حذف عنصر من طابور محدث حالا يحدث خطأ: Front(creat(q)) = error

and Delfront(creat(Q)) = error

4. إضافة عنصر ثم مباشرة قراءة رأس الطابور يعرض ذلك العنصر إذا كانت فارغة من قبل

و إلا فسيكون رأس الطابور : If Empty(Q) Then Front(append(x,q) = x else

Front(append(x,q)=front(q).

5. وكذلك بالنسبة لحذف عنصر : If Empty(Q) Then

Empty(Delfront(Append(x,Q))) = True else Delfront(Append(x,Q)) =

Append(x,Delfront(Q)) .

و إليك الآن هذه العمليات على شكل إجراءات قم بتنفيذها ثم حولها إلى دوال.

```
program QueueOperations;
uses Wincrt ;
type DataTYPE = string;
QueuePOINT = ^NodeTYPE;
NodeTYPE = record
    next: QueuePOINT;
    Data: DataTYPE
end;
Var Front , Rear, Node : QueuePoint ;
    item : DataType ;
    foutQ : file of nodeType ;
    filename : string;
```

*خوارزمية البدء و الإضافة :

بغية تسهيل الفهم على الطالب قسمنا عملية الإضافة إلى إجراءاتين : الأول للبدء نحدث أول

عنصر من الخلف new(rear) ونحتفظ بعنوانه في رأس القائمة front و هو الإجراء initialize و

أما الإحداث فقد قمنا به في البرنامج الرئيسي بتعيين في front و rear القيمة Nil .

ثم إجراء الإضافة يعمل على إتجاهين : إن كانت القائمة فارغة ينادي الإجراء initiliaze

لإدخال أول عنصر و إلا يقوم يقوم بالإضافة دائما من الخلف: new(rear^.next) إلى

آخره... نفس الطريقة التي إتبعنا في الفقرة 1.7 القائمة الخطية أول من يدخل هو أول من

تخرج . fifo

```
procedure Initialize ;
begin
```



```

new (Rear); write('Data ? : '); readln( rear^.data );
Front := Rear ;
end;
procedure insert ;
begin
  if ( front = nil ) and ( rear = nil ) then    initialize
  else begin
    new ( rear^.next); rear := rear^.next ;
    write('Data ? : '); readln( rear^.data );
    end;
    rear^.next :=nil ;
  end;

```

3. للقيام بالإضافة ما علينا إلا نداء الإجراء insert و نكرر تنفيذه عند الطلب .

```

procedure append;
begin
  repeat
    insert ;
    writeln('another node [y]es / [n]o');
  until readkey in ['n','N'];
end;

```

*خوارزمية عرض محتوى القائمة الطابور:

1. ندخل للقائمة من الرأس front و نضع السهم في متغير آخر سهم.

2. نكتب محتوى الحقل node^.data .

3. نتقدم في القائمة بأخذ عنوان الماوي من node^.next في node .

4. نكرر الخطوات 2 و 3 ما دام السهم node يختلف عن nil .

```

procedure ViewQueue;
begin
  node := front;
  while node <> nil do begin
    writeln(node^.data );
    write('Taper CR pour continuer '); readln;
    node := node^.next ;
  end; writeln('>>>end of queue>>>');
end;

```

*خوارزمية الحذف :

1. نأخذ عنوان العنصر الذي سنحذفه من الرأس حيث لا نستطيع الدخول إلا من الرأس و لا نحذف إلا من الرأس .

2. نحفظ بعنوان العنصر الموالي front[^].next في front

3. نحذف العنصر بالإجراء dispose

```

Procedure pop ;
{ Retrieve the front value from the queue. }
var TempPtr: QueuePOINT;
begin
  TempPtr := Front;
  Front := Front^.next; { Advance the front pointer. }
  dispose (TempPtr); { Remove the old front node. }
  writeln('One component deleted');
  if front = nil then rear := front ;
end;
```

4. إذا أردنا حذف كل القائمة الطابور ننادي بالإجراء POP ما دام Front <> nil

```

procedure delete ;
{ Return each node in the queue to computer memory. }
var TempPtr: QueuePOINT;
begin
  while Front <> nil do pop ;
  front := nil ; rear := nil ;
  writeln('All queue deleted ....>>>');
end;
```

*خوارزمية التخزين :

1. نعين اسم الجذاذة التي نريد التخزين عليها بـ: assign(..)

2. نفتح الجذاذة للكتابة بـ: rewrite(foutq)

3. ننفذ إلى القائمة من الرأس بـ: node := front

4. نكتب على الجذاذة بـ: write(foutQ,node[^])

5. نتقدم في القائمة بـ: $node := node^{next}$

6. نكرر الخطوات 4 و 5 ما دامنا لم نصل إلى نهاية القائمة $node \neq nil$

```
procedure save;
begin
  write('Save as ? : '); readln( filename );
  assign( foutq ,filename);
  rewrite( foutQ ); node := front ;
  while node <> nil do
    begin
      write( foutq , node^);
      node := node^.next ;
    end;
  close ( foutQ );
  writeln('>>>> queue saved >>>> ');
end;
```

*خوارزمية شحن القائمة من القرص إلى الذاكرة المركزية :

1. نعين الجذاذة التي سنشحن منها القائمة إلى الذاكرة بـ: $assign(...)$

2. نفتح الجذاذة للقراءة منها بـ: $reset(foutQ)$

3. نبدأ بإحداث أول عنصر من القائمة الطابور و ذلك من الخلف بـ: $new(rear)$ ثم تقديم

عنوانه إلى المتغير $front$ للإحتفاظ برأس القائمة فيه . ولكن لا نقوم بهذه الخطوة إلا إذا إختبرنا

هل بالجذاذة على الأقل عنصر واحد , فإن لم يكن بها أي عنصر $eof(foutQ)$ تكون $true$ و

بالتالي لا نحدث القائمة و ستكون فارغة بـ: $rear:=front$ و $front := nil$

4. نقرأ من الجذاذة العنصر الأول بـ: $read(foutQ, rear^)$ حيث يوضع مباشرة في

التسجيل المحدث حركيا بـ: $new(rear)$.

5. نواصل إحداث القائمة بـ: $new(rear^{next})$ و نضع في $rear:=rear^{next}$ ولكن

كذلك هنا لا نطلب إحداث العنصر بالذاكرة إلا بعدما نتأكد مازال في الجذاذة تسجيل للنقل

6. نكرر الخطوات 4 و 5 ما دامنا لم نصل إلى نهاية الجذاذة بـ:

```
while not eof(foutQ) do ....
```

```

procedure load ;
begin
    write('load from ? : ');readln( filename );
    assign( foutq ,filename);
    reset( foutQ );
    if not eof ( foutQ )then
        begin    new ( rear ) ; front := rear ;    end
    else begin front := nil ; rear := front ;    end;
    while not eof( foutq ) do
        begin
            read( foutq , rear^ );
            if not eof( foutQ ) then
                begin    new ( rear^.next); rear := rear^.next ; end;
            end;
            rear^.next := nil ;
            writeln('>>>> queue loaded .....');
            close ( foutQ );
        end;
begin {queueOperation}
    front := nil ; rear := front ;
    repeat
        writeln('>>[v]iew the queue ');
        writeln('>>[A]ppend in a queue : push') ;
        writeln('>>[p]op 1 node from a queue ');
        writeln('>>[d]elete all the queue ');
        writeln('>>[i]nsert');
        writeln('>>[l]oad from file ');
        writeln('>>[s]ave in file ');
        writeln('>>>Break');
    case readkey of
        'B','b' :Break;
        'i','I' : insert;
        'v','V' : viewQueue;
        'a','A' : Append;
        'p','P' : pop ;
        'd','D' : delete ;
        's','S' : save ;
        'l','L' : load ;
    end
end

```

```

end;
writeln('another run [y]es or [n]o ');
until readkey in ['n', 'N' ];
writeln('FFFFFFFFiiiiiiiiNNNNNNNNNNNNNNN');
end.{queueOperation}
end.

```

4.7 بنية البيانات الغير خطية :بنية الأشجار.

Non-Linear complex Data Structure: Trees.

1.(التعريف : الأشجار بنية بيانات غير خطية أهمها الأشجار الثنائية Binary trees

(arbres binaires) .

2.(تعريف الأشجار الثنائية: definition of binary trees

الشجرة الثنائية بنية سلمية لعناصر تسمى العقد nodes. أهمها العقدة الأولى في المستوى

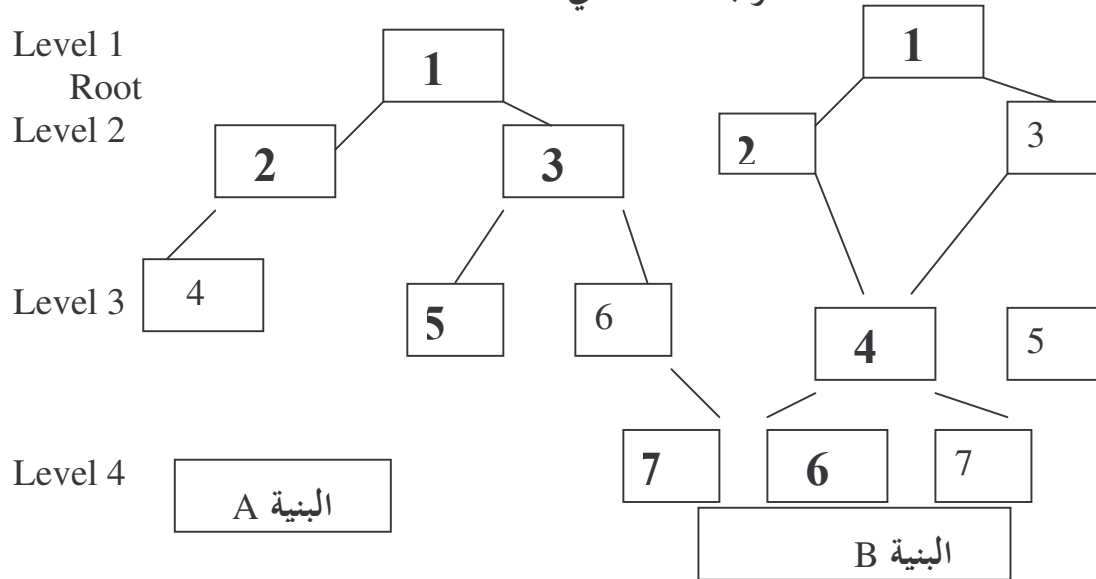
الأول و تسمى : الجذر root . A binary tree is a hierarchical structure of component .

called :nodes,with a distinguished node at the first level called root.

للشجرة الخاصة التالية : كل عقدة في المستوى i level مربوطة بعقدة واحدة فقط بالمستوى

الأعلى (i-1) و على الأكثر بعقدتين فقط بالمستوى الأسفل (i+1) علما بأن i أكبر من 1

(i>1) . لنضرب المثال التالي :



في هذا المثال يظهر جليا أن البنية A شجرة ثنائية binary tree و البنية B غير ثنائية حيث أن العقدة 4 في المستوى 3 مربوطة بعقدتين في المستوى 2 وهما العقدة 2 و 3 و هو مخالف للقاعدة أعلاه.

خواص الأشجار: لكل عقدة في المستوى i ($i > 1$) عقدة أب أو والد $\text{parent}(\text{node})$ في المستوى $i-1$ و العقد المربوطة في المستوى $i+1$ تسمى الأولاد أو الأبناء children . كل عقدة بدون ولد تسمى ورقة leaf . إذا كانت العقدة تحتوي على الأقل على ولد واحد تسمى عقدة داخلية interior node . المسلك path هو الطريق الوحيد الذي نصل به إلى العقدة من الرأس أو الجذر root . طول المسلك $\text{the length of a path}$ هو عدد العقد التي نمر عليها للوصول إلى العقدة من الجذر ناقص واحد. مثال طول المسلك من 1 إلى 7 يساوي 3. ارتفاع the height الشجرة هو أطول مسلك من الجذر إلى أي عقدة في الشجرة و قد نعرفه بعدد مستويات الشجرة ناقص واحد في مثالنا ارتفاع الشجرة $\text{height} = 3$. الشجرة الفارغة هي كل شجرة بدون عقد. $\text{the empty tree is a tree without nodes}$.

فرع الشجرة n أو الشجرة التحتية subtree لشجرة T هي العقدة n من T بالإضافة إلى كل العقد m التي تستجيب للخاصية التالية: المسلك الوحيد من الجذر T إلى m يجب أن يمر على n و بالتالي تصبح العقدة n جذر تلك الشجرة التحتية أو الفرع في مثالنا الفرع المتجذر من 3 يحتوي على العقد 3 و 5 و 6 و 7, و أما الفرع المتجذر من 4 فلا يحتوي إلا على تلك العقدة.

في أغلب التطبيقات يستحسن التفريق بين ولدين children العقدة. و نقول ولد اليسار و ولد اليمين و هو ما يدفع إلى التفكير في نوع من الترتيب ordering و تصبح الشجرة مسمات

في كثير من الأحيان بالشجرة الثنائية المرتبة **ordered binary tree**

و سنتكلم إذا عن الفرع اليسار من الشجرة left subtree و الفرع اليمين للشجرة right subtree .

* التصريح بالشجرة : implementation

```
Type BinTreePt = ^BinTComp;
  BinTComp = Record
    Field : string[25];
    Left , right : BinTreePt;
  End;
  Var root , temp : BinTreePt;
  ...
```

لنضرب مثال كيف نحدث بنية الشجرة بطريقة مباشرة :

```
new(root);root^.field := 'علي';new(root^.left);
root^.left^.field := 'الحسن';root^.left^.right := nil;
new(root^.left^.left);temp := root^.left^.left;temp^.field := 'الحسين';
  temp^.left := nil; temp^.right := nil ;
new(root^.right);temp:=root^.right;temp^.field := 'زين العبيد';
new(temp^.left);temp^.left^.field := 'أم كلثوم';
temp^.left^.left:=nil;temp^.left^.right:= nil;
new(temp^.right);temp:=temp^.right;temp^.field := 'الطاهر';temp^.left:=nil;
new(temp^.right);temp:=temp^.right;temp^.field := 'صالح';
  temp^.left:=nil;temp^.right:=nil;
```

تلاحظ أن إحداث الشجرة بهذه الطريقة سيكون متعباً و غير عملي . لذلك سنلجأ إلى

الخوارزمية التراجعية Recursive Algorithm لإحداث الشجرة.

و إليك الآن بعض العمليات على الأشجار الثنائية الخاصة بالبحث Binary search tree :

دمج عنصر أو أكثر في الشجرة inserting elements into a tree و عبور الشجرة

traversing و هي عملية عرض محتوى الشجرة .

*1. ما هي الشجرة الخاصة بالبحث Search Tree :

هي شجرة ثنائية من خواصها أن كل عقدة n تحتوي على معلومة I(n) و

بالتالي إذا n تحتوي على يسارها على شجرة تحتية أو فرع $sutree$ غير فارغة S_L فإن لكل العقد

$$I(n_L) < I(n) > S_L \text{ في } n_L$$

و كذلك إن كان لـ n فرع على اليمين غير فارغ S_r فإن جميع العقد n_r في S_r نجد : $I(n) < I(n_R)$.

غالباً ما تكون المعلومات المراد تخزينها تتمتع بنوع من الترتيب المفهرس lexicographical ordering .

1*. خوارزمية إحداث الشجرة: Generating .

عملية الإحداث تتمثل في تكرار عملية الإدماج Inserting كالتالي:

ليكن لدينا شجرة بحث ثنائية T : binary search tree و سهم $proot$ إلى جذرها $root$
نريد دمج العنصر x في T أي نريد في الحقيقة إحداث شجرة بحث أخرى . عندنا حالتين : T
فارغة يعني $proot = nil$ و حالة T غير فارغة $proot \neq nil$.
إذا T تكون فارغة يجب إحداث عقدة : تعيين القيمة x إلى حقلها ثم جعل $proot$ يشير إلى تلك العقدة.

إذا T غير فارغة يجب أن نبحث عن المكان المناسب لدمج القيمة x : و سنتخذ الإستراتيجية التالية : نحاول البحث على x كأنها موجودة . إذا وجدناها إذا لا يوجد مكان لدمجها حيث في تعريفنا السابق للشجرة البحث تمنع ازدواجية العناصر . فإن لم نجد العنصر x حيث في مكان ما على اليسار أو اليمين سنجد السهم nil .

و إليك البرنامج و إجراءات الدمج و الإحداث و عرض محتوى الشجرة حسب الطرق

الثلاث : مرتبة و قبل الترتيب و بعد الترتيب : in-order , pre-order, post-order .

```
program TreeOperations ; { opeartions on Binary search tree }
uses wincrt ;
Type BinTreePt = ^BinTcomp ;
BinTcomp = record
    field : integer ;
    left , right : BinTreePt ;
```



```

    end;
var Proot : BinTreePt ; x : integer;
procedure Insert ( var proot : BinTreePt ; x : integer);
begin
    if Proot = nil Then
        begin
            new( proot );
            with Proot^ do
                begin
                    field := x ;
                    left := nil ;
                    right := nil ;
                end
            end
        end
    else
        if x < proot^.field then
            Insert( proot^.left , x )
        else insert( proot^.right , x );
    end;
procedure generaTree ;
begin
    Repeat
        write('data ');readln( x );
        Insert( proot , x );
        writeln('more node [y]es or [n]o ');
    until readkey in ['n'];
    writeln('Tree generated ');
end;
procedure PretTrav( var P : BintreePt );
begin
    if P <> nil then
        begin
            writeln (P^.field );
            pretTrav ( P^.left);
            pretTrav ( P^.right );
        end;
    writeln('Tree Traversed ');
end;

```

```

procedure PostTrav( var P : BintreePt );
begin
    if P <> nil then
        begin
            postTrav ( P^.left );
            postTrav ( P^.right );
            writeln (P^.field );
        end;
        writeln('Tree Traversed ');
    end;
procedure InTrav(var P : BintreePt );
begin
    if P <> nil then
        begin
            InTrav (p^.left );
            writeln (P^.field );
            InTrav (p^.right);
        end;
        writeln('Tree Traversed ');
    end;
begin { binary search tree }
    repeat    clrscr;
        writeln('>>Menu of Binary Search tree ');
        writeln('>>[G]enerating Binary Search Tree');
        writeln('>>[I]n-order Traversal of the Tree');
        writeln('>>[P]re-order Traversal ');
        writeln('>>[T]:posT - order Traversal ');
        writeln('>>[B]reak ');
        case readkey of
            'g','G' : GeneraTree ;
            'T','t' : Intrav ( proot );
            'p','P' : PretTrav ( proot );
            't','T' : PostTrav( proot );
            'b','B' : break;
            else writeln ('Unknown choice ');
        end;{case}
        writeln('>>>>>>>>>Another Run [y]es or [n]o ? ');
    until readkey in ['n','N'];

```

```
writeln('>>>> End of Binary Search Tree operations <<<<<<<');
end.{treeop}
```

الباب الخاتمة :نصوص تمارين (محلولة) وبرامج شتى. Chapitre Conclusion: Exercices (corrige) et programmes divers. كيف تبحث عن خوارزمية مسألة:

Méthode de Recherche d'un Algorithme .

(from : "How to solve it" of the Great Mathematician Polya.)

Comprendre l'énoncé du problème:

Quelles sont les inconnues ?

Quelles sont les données ?

Quelles sont les conditions ?

Est-ce qu'elles peuvent être vérifiées?

Écrire les hypothèses , les inconnues , les conditions chacun par son symbole

c.a.d son identificateur approprié.

Séparer les différentes parties de la(ou les) condition .

Peut-on l'écrire , la formuler.

Chercher l'algorithme.

Trouver la relation entre les données et l'inconnue.

As-tu déjà rencontré cette situation ?

Ou d'une autre manière , pose différemment?

Connais - tu une situation ayant une relation avec celle posée.

Connais-tu une théorie , théorème , formule pouvant t'aider.?

On peut être ramené a réfléchir a un problème équivalent, si on ne trouve pas la solution du problème directement.

Observons bien l'inconnu , et essayant de se rappeler un problème ou cette inconnue figure , ou une inconnue semblable.

Ainsi ce problème a une relation avec celui qu'on vous propose , il a été résolu.

Peut-on utiliser le même algorithme .?

Peut-on utiliser le résultat obtenue.?

Ou bien introduisons quelques changement pour pouvoir l'utiliser.

On doit arriver a l'Algorithme recherche.

Peut - on reprendre l'énoncé en le formulant nous même.?

Revenons aux définitions , propriétés pour pouvoir exprimer notre raisonnement.

Si on n'arrive pas trouver une solution au problème , alors essayons de résoudre un problème équivalent , ayant une relation avec celui pose.

Est-ce qu'on connait un problème se rapprochant mais plus facile a résoudre , ou un problème d'ordre général , ou un cas particulier ou un problème inverse.?

Peut-on résoudre une partie du problème ?

Prenons une partie de la condition et négligeons le reste.

Peut-on après ça voire la détermination de l'inconnue apparaître clairement.?

Peut-on changer l'inconnue ou les données en d'autres données et inconnue(s), s'il est nécessaire afin de les ramener plus proches les uns des autres.?

As-t-on utilisé toutes les hypothèses ?

As-t-on utilisés toutes les parties de la condition?

As-t-on pris en considérations les principes essentiels énoncés dans le problème, ou se rapportant au données , et inconnue(s)?

Exécutons l'Algorithme :

Au cours de l'écriture de l'algorithme soyons sure de ne rien oublier , suivons le raisonnement étape par étape .

Peut-on voire enfin que l'algorithme est valable .

Peut-on vérifier sa validité.?

Révisons...Vérifions :

Peut-on arriver au résultat souhaite ?

Peut-on vérifier la démarche .?

Peut - on arriver au même résultat d'une autre manière?

Peut-on arriver a concentrer l'algorithme de façon a pouvoir le résumer rapidement.?

Peut-on ce rappeler de cet algorithme sans aucune aide matérielle ,c.a.d le mémoriser?

Peut-on enfin penser a utiliser le même algorithme , ou le même résultat a un autre problème.(principe de la programmation modulaire tel que l'UNIT en Pascal)

و في الأخير أضيف إليك بعض الحكم من التراث العربي الإسلامي تنير طريقك عند البحث
عن حل أي مسألة تطرح عليك :

اليقين لا يزول بالشك . فلا تبني على الظن و الشك و لكن على ما لديك من علم ثابت
تطمئن إليه.

من استعجل الشيء قبل أوانه عوقب بحرمانه : يعني لا تتعجل في الفهم و الإحاطة بالشيء و لكن عليك بالتدرج و التبصر و الإستفادة من كل ما يمكن أن يقدم لك عوناً و علماً و الابتعاد عن الغرور و الإكتفاء بالزرر و القلة من العلم و أن تردد دائماً قول الله سبحانه و تعالى : و قل رب زدني علماً . صدق الله العظيم.

*** نصوص التمارين :

التمرين 1 : أكتب برنامجاً يبحث عن عنصراً من متتالية تراجعية suite recurente المعرفة كما يلي

$$U_{n2} = U_n + U_{n+1} :$$

(* calcul d'un terme quelconque d'une suite recurente définie par U_{n2}
 $= U_{n+1} + U_n$ *)

التمرين 2 : أكتب برنامجاً يحول أي عدد في الأساس B إلى الأساس العشري حسب طريقة

: shema de Horner

$$F(x) = (...(((a_n x^n + a_{n-1})x + a_{n-2})x + ... + a_1)x + a_0$$

Exp: $z = 52346_{base7}$

Z	0
Z	$0 * 7 + 5 = 5$
Z	$5 * 7 + 2 = 37$
Z	$37 * 7 + 3 = 262$
Z	$262 * 7 + 4 = 1838$
Z	$1838 * 7 + 6 = 12872$

التمرين 3 : أكتب برنامجاً يعرض متتالية Fibonacci المعرفة كما يلي :

$$F_{n+2} = F_{n+1} + F_n \text{ و } F_1 = f_2 = 1$$

التمرين 4 : أكتب برنامجاً يعرض عامل factorial العدد n .

التمرين 5 : أكتب برنامجاً لحل معادلة من الدرجة الثانية.

التمرين 6 : أكتب برنامجاً يقوم بترتيب قائمة أسماء بشق طرق الترتيب المعروفة : طريقة

الفقاعات و بالإختزال و بالدمج و بالعد و بالإنتخاب و الترتيب السريع مستعملاً وحدة unit

و جذاذة نص لقراءة القائمة و كتابة النتيجة في جذاذة نص fichier text .

التمرين 7 :مسألة لمراجعة قواعد البرمجة , التعليمات البنيوية و الإجراءات و الدوال .

But : le but de ces exercices est de vous initier au règles de Base algorithmique, les règle du langage Pascal et l'utilisation des Procédures sans Parametres,avec Paramètres , des Fonctions et l'utilisation des Units ,et de mettre en route le processus de la prog. Structurée en 5 étapes A,B,C,D,E.

A :Ecrire l'algorithme et le programme pour résoudre les problèmes mathématiques suivants :

a chaque question un petit algorithme et programme doit être écrit sans oublier l'exécution manuelle .

1. la surface d'un triangle par la formule suivante $S = 1/2 bh$;

b :la base ; h:hauteur

2.la surface d'un parallélogramme $S = bh$;

3.la surface du trapeze $S = 1/2(a + b) h$;

a :petite base,b:grande base,h:hauteur

4.la surface du cercle $S = \pi R^2$; le périmètre $P = 2\pi R$, R : rayon

i.e. volume d'un cylindre quelconque ou prisme avec des bases parallèles:

$V = Bh$; B : surface de la base ; h : hauteur.

6.cylindre circulaire droit : $S = 2\pi Rh$; Volume = $\pi R^2 h$.

R:rayon de la base;h:hauteur du cylindre

7. Any cone or Pyramide : Volume = $1/3 Bh$;

B:surface de la base;h:hauteur.

8.Sphere : Volume = $4/3\pi R^3$; $S = 4\pi R^2$

9.L'inverse d'un argument :inver = $1/ \text{argument}$;

10.Le logarithme d'un argument de base :B

$\text{Log}_B(\text{argument}) = \ln(\text{argument}) / \ln(B)$

11. la puissance : $X^n = \exp(n \ln (x))$;

12. La conversion des Degré en radian

13. La conversion des Radian en Degré .

14. La Tangente d'u angle .

15 . La sécante d'un angle sécante = $1 / \cos (\text{angle})$;

16. La cosecante = $1 / \sin(\text{angle})$;

17. La cotangente d'un angle

18 . L'arc sinus arcsinus = $\arctan(a / \sqrt{1-a^2})$

19.L'arc cosinus de a : arccos = $\arctan(\sqrt{1-\text{sqr}(a)} / a)$;

20.L'arc cosécante de a : arccsc = $\arctan(1/ \sqrt{\text{sqr}(a) - 1})$;

21.L'arc secante de a : $\text{arcsec} = \arctan(\sqrt{\text{sqr}(a) - 1})$;

22.L'arc cotangente de : $\text{arccot} = \arctan(1 / a)$;

23.....ajouter d'autres fonctions et calcul utiles ...

B : après avoir maîtriser les notions fondamentale d'un programme simple Transformer les petits programmes en procédures sans paramètres que Vous appellera dans votre programme une a une .

C: L'tape C consiste a les transformer en Procédures avec Paramètre passe par valeur et par adresse selon le besoin qu'on appellera dans un Programme .

D : Transformer les Procédures en Fonction selon le besoin .

E: Écrire les fonctions et Procédure les plus communes dans une UNIT pour les rendre PUBLIC et les utiliser dans votre Programme.

Votre Programme final sera un petit logiciel qui fournira les services suivant:

*calcul mathématique simple

*calcul Trigonométrie

*calcul Géométrie

*calcul Mathématique Avancé

Exercice N°8 :Proposer un programme qui teste si un caractère est Alphabétique , numérique ou autre en utilisant le type **ensemble : SET of ...**.Former l'ensemble des lettres et des chiffres en utilisant une procédure `litEnsmble (var E :...)` et le test par une procédure ou fonction.

Exercice N°9.Un éducateur trouve des difficulté a répéter les notions sur les ensembles a ces élèves vous a demande de l'aider par un programme qui enseigne ces notions aux élèves et ce répète selon le choix de l'élève ;les opérations sur les ensembles sont :1-former un ensemble .2-voir le contenu de l'ensemble .3-former un ensemble par l'ordinateur (générateur de nombre aléatoire).4-union de 2 ensembles .5-intersection de 2 ensembles .6-Différence entre 2 ensb.7-comparaison entre 2 ensb (égalité ou différence).8-inclusion ou contenance .Écrire un tel programme avec le type Set of 1..100; par exp.chaque opération comme procédure selon le choix de l'élève et se répéter selon le choix

Exercice N°10:Une compagnie aérienne vous a demandé de lui fournir un Programme Interactif qui maintient les Informations des Vols (Avions) .Les Vols (max 200)sont définies comme suit :

1.Le numéro de vol . 1..9999 2.Capacité du vol:300 au Max.

3.Reservation : 0..300.

4.Départ : contient 2 informations :

Aéroport :3 caractères et le Temps:0000..2350.

5..Arrive :

contient 2 informations : Aéroport :3 caractères et le Temps:0000..2350

*1.Ecrire la structure de donnée des Vols(200 aux Max) .

*2.Supposer que la liste des vols est déjà lus :Écrire la procédure qui indique

quels sont les vols complets.

Exercice N°11: Proposer un programme qui peut copier un fichier Text dans un autre et la fusion (ajout)de 2 fichiers a un seul .Les lignes du fichiers source sont de 80 caractères aux max.

Exercice N°12-Une entreprise de carte routière vous a demande de lui préparer un logiciel qui répond aux questions des routiers en matière de distance des villes par rapport l'une a l'autre , elle vous a définie 3 axes principaux : Alger-est,Alger-ouest,et Alger - sud .Pour chaque axe on doit citer la ville et sa distance par rapport a Alger ;comme suit :

Axe Alg-Est	Axe Alg-Ouest	Axe Alg-Sud
Const :500	Oran:450	Ghardaya:800
Annaba: 650	Tlemcen:600	Tindouf.1500

Etc..le nombre de ville pour chaque axe est de 10 .

1-ecrire les déclarations nécessaire pour une telle structure de donnée.

2-ecrire la procédure de lecture de donnée.

3-donner l'algo.et la procédure ou fonction qui cherche la ville la plus proche d'Alger pour chaque axe.

4-la même chose pour la ville la plus loin d'Alger.

5-ecrire le programme principal qui fournit ces services en utilisant les procédures internes et une 2°fois externe dans une UNIT .

6-Comment peut-on rendre ce programme un vrai logiciel utile aux routier..

proposer une Generalisation et une analyse du problème interessante..par ville.commune ..wilaya..

Exercice N°13 :Dans les gros computer systems on garde l'enregistrement des utilisateurs qui se connectent et le temps de connections .Ces informations sont enregistrées dans 2 tables parallèles ,la première array contient les User Names et la seconde contient le temps du jour quand il s'est connecté :loginTime.Les noms et temps sont gardés ordonnés selon le temps de connexion .Le premier dans la liste est le premier qui s'est connecté et la dernière position contient le dernier connecté.Par exemple :

UserName : Ahmed	KadurAicha	Naro	Farida
LoginTime : 1:05	2:25	2:55	5:27 6:31

1: Définir la structure de donnée .

2: écrire la procédure de lecture de donnée de l'input(qui est normalement automatique ,interne)

3:écrire la procédure qui a partir du userName comme paramètre nous rend les temps de connections(loginTime) du UserName .

4.ecrire le programme principal avec les procédures internes et externes.

Remarque: Déclarer le userName comme type déclaré une fois et une 2° fois comme string.

Exercice N°14: L'un des grands objectifs de l'utilisation des computers pour les années a avenir est l'enseignement a distance :Distance Learning.

Comme exercice penser a écrire un programme qui apprend au débutant votre langue Arabe ; commençons par الحروف القمرية و الشمسية .

L'enfant doit écrire un mot et vous devez lui corriger s'il a oublié Echada الشدة si le 3° caractère est Chamsi sinon vous le félicitez . الكلمة صحيحة أحسنت الحرف . القمر لا يكتب بالشدة و الشمسي يجب تشديده .

Exercice N°15: Un autre exercice par exemple consiste a lui apprendre l'écriture de el hamza . الهمزة في بداية الكلمة مثلا . la règle dit qu'il faut l'écrire sur el alif.

Faite lui écrire des mots avec el hamza et donner lui la correction pour chaque mot el hamza sur el alif au lieu d'être sur la ligne. تقول القاعدة: إذا وقعت همزة القطع في بداية الكلمة كتبت على الألف سواء كانت مفتوحة أو مضمونة أو مكسورة مثل: أخذ, أعيد, إن.

Exercice N°16 ajouter une procédure de tri de la table des ville de l'Ex.1

Que remarquer vous quand on liste les tables des userName et les tables de Temps parallèles; sont elles toujours parallèles devant chaque nom y a t'il son temps de Login ?

Comment doit on résoudre ce problème?

Exercice N°17. Les premiers éditeurs de texte étaient de simple éditeur de ligne comme EDLINE de Ms dos

Comme exercice Écrire un éditeur de ligne qui propose les services suivants :

1.creation d'un texte comme Table de ligne de 80 caractères au max. le max.de ligne est 20.

2.ajout d'une ligne s'il y a de l'espace.

3.insertion d'une ligne après le numéro de ligne x .

4.suppression d'une ligne de n° i .

5.modifier le contenu d'une ligne .

6.copier une ligne de i vers j .

Exercice N°18. Une matrice est dite symétrique si pour toutes les valeurs des éléments i, j les valeurs des éléments j, i leurs sont identiques.Par exp.si A est

une matrice carrée et si l'on a $A(3,4)=15$; alors $A(4,3)$ doit aussi valoir 15 et ainsi de suite pour les autres éléments .

Écrire le programme qui lit une matrice carrée , test si elle est symétrique renvoie la réponse et son contenu.

Exercice N°19: Soit un fichier Text qui contient une série de nombre , charger en mémoire ces données en utilisant la structure dynamique linéaire :Pile :(stack) ou File (Queue).

Problème N°1 : (Array of Record) Vous travailler dans une banque ou les clients sont défini comme suit : numéro de compte , nom prénom , adresse, type d'opération (C:crédit :virement ou D:débit:retrait), montant de l'opération , date de chaque opération.

Définir la structure de donnée pour un client et de 100 clients.

Écrire la procédure de saisi de 10 clients comme exp.

En supposant qu'un client peut faire plusieurs opérations de crédit et débit ,c.a.d est enregistré plusieurs fois ; donner lui le total des mouvements de crédit et débit et le solde :c.a.d le total_credit -total_debit , la recherche du client dans la table d'enregistrement ce fera par le numéro de compte.

En supposant que le client a oublié son numéro , trouver son numéro connaissant son nom_prenom .

Le comptable en fin de chaque journée a besoin du total des mouvements effectués pour tous les clients, donner lui le résultat :somme_debit et somme_credit

Écrire le prog.principal qui fournit ces services sous forme de menu .

Problème N°2:(Array of record) Vous travailler dans une bibliothèque qui garde les informations suivantes pour chaque livre : Titre , Auteur ,éditeur , date d'edition,Numero d'enregistrement Dewey:nombre décimal , et si le livre est sorti : la date de sortie .

Écrire la déclaration pour 1 livre , et une bibliothèque de 20 livres par exp.

Écrire la procédure qui liste tous les livres : titre et auteur des livres de numéro décimal Dewey entre 100 et 200 par exp.

Écrire la procédure de recherche d'un livre connaissant l'auteur seulement.

Même procédure connaissant le titre seulement.

Donner la liste des livres qui sont sortis .

Donner la liste des livre par éditeur :titre , date .

Écrire le programme qui fournit ces services sous forme de menu.

Array of record avec fichier text

Problème N°3 : Vous travailler dans une banque ou les clients sont défini comme suit : numéro de compte , nom prénom , adresse, type d'opération (

C:crédit :virement ou D:débit:retrait),montant de l'opération , date de chaque opération.

Les informations des clients sont enregistrés dans un fichier Text ,chaque champs par ligne ,les clients les uns a la suite des autres .

Definir la structure de donnée pour un client,et 10 clients.

Écrire la procédure de lecture de 10 clients comme exp.du fichier text

Data_In

En supposant qu'un client peut faire plusieurs opérations de crédit et débit ,c.a.d est enregistré plusieurs fois ; donner lui le total des mouvements de crédit et débit et le solde :c.a.d le total_credit -total_debit , la recherche du client dans la table d'enregistrement ce fera par le numéro de compte .Le resultat sera envoyé vers l'imprimante (qui est un fichier Text).

En supposant que le client a oublié son numéro , trouver son numéro connaissant son nom_prenom (sur fichier Ecran).

Le comptable en fin de chaque journée a besoin du total des mouvements effectués pour tous les clients, donner lui le résultat :somme_debit et somme_credit sur fichier Imprimante.

Écrire le prog.principal qui fournit ces services sous forme de menu .

Problème N°4: Vous travailler dans une bibliothèque qui garde les informations suivantes pour chaque livre : Titre , Auteur ,éditeur , date d'edition,Numero d'enregistrement Dewey:nombre décimal , et si le livre est sorti : la date de sortie .

Écrire la déclaration pour 1 livre , et une bibliothèque de 20 livres par exp.

Écrire la procédure de lecture de donnée d'un fichier Texte DATA_IN qui contient les informations de 20 livres par exp.chaque champs dans une ligne .

Écrire la procédure qui liste tous les livres : titre et auteur des livres de numéro décimal Dewey entre 100 et 200 par exp.dans un fichier Data_Out ("listlivr.out").

Écrire la procédure de recherche d'un livre connaissant l'auteur seulement et le résultat envoyé au fichier standard output.

Même procédure connaissant le titre seulement.

Donner la liste des livres qui sont sortis et le résultat vers un fichier text "livre.out".

Donner la liste des livres par éditeur :titre , date et le résultat sur le fichier output.

Écrire le programme qui fournit ces services sous forme de menu. sur les fichiers Binaires .

Problème N°5 Reprenons le même problème N°1 sur la banque avec la

même structure de donnée mais cette fois ci on va utiliser un fichier pour enregistrer tous les clients au lieu d'un nombre défini par le programme tel que `array[1..100] of client` ; le fichier binaire vas nous permettre de stocker le nombre de client disponible sans limite du nombre (sauf celle imposé par la capacité disque) .

Definir la structure de l'enregistrement et du fichier .

Écrire l'algo. et la procédure qui crée le fichier .

Écrire l'algo. et la procédure d'ajout d'un ou plusieurs clients .

Écrire l'algo. et la procédure de modification des informations d'un ou plusieurs clients .

Écrire l'algo. et la procédure de suppression d'un ou + clients.

Écrire la procédure qui liste tous les clients .

Écrire le menu du programme principale qui fournit ces services et les services 3 , 4 , 5 demande au Pb N°1.

Problème N°6 : même chose avec le problème N°2 sur la bibliothèque ou cette fois c'est un fichier Binaire que vous aller utiliser au lieu d'une table d'enregistrement.

Donner la str. De donnée de l'enregistrement et du fichier.

Écrire la procédure Création du fichier.

Écrire la procédure d'ajout d'un ou + livres.

Écrire la procédure de modifi. D'un ou + livres .

Écrire la procédure de suppression d'un ou + livres .

Écrire la procédure de recherche d'un livre par titre ou auteur.

Écrire la procédure qui liste les livres sortent.

Écrire le Menu du prog. Principal qui fournit ces services .

***** نص امتحان :**

Problème N°1: évaluation des connaissances mathématiques élémentaires.
(Barème 60/100)

Un centre de recherche en éducation vous à chargé de lui préparer un logiciel d'évaluation des connaissances en mathématiques des élèves du 1^o cycle. Votre programme doit être interactif posant des questions à l'élève , le guidant, lisant ses réponses , lui affecter des points pour les réponses justes et les réponses fausses ; et enfin lui donnant la note finale en nombre de points .

Votre programme aura l'allure suivante :

Bienvenue au programme d'évaluation des connaissances mathématiques .

.Je vais te poser des questions et te demande d'entrer des données et les réponses aux questions ;a chaque réponse juste je te donne un +5 et un -1 pour une réponse fausse

Ton nom est ? :(entrer votre nom s.v.p):

La 1° opération est l'addition :

Entrer 2 Nombres : (par exemple il va entrer 15 25)

Le programme écrit : $15 + 25 =$ (**entrer le résultat de la somme**).....

si la réponse est juste le programme répond :

Bonne réponse . Félicitation : +5

Si la réponse est fausse le programme répond :

Réponse fausse : -1 .On te donne une deuxième chance .

Le programme revient a l'écran passé :

Une 3°chance lui est donné s'il ne réussit pas ; sinon on passe a l'opération suivante:

La 2° opération est la soustraction :

Entrer 2 Nombres :.....

Le programme écrit :..... - = (**entrer le résultat ?**):.....

si la réponse est correcte le programme répond :

Réponse correcte .Félicitation :+5

Si la réponse est fausse le programme répond :

Réponse fausse : -1 .On te donne une deuxième chance .

Le programme revient a la fenêtre passé :

Une 3°chance lui est donné s'il ne réussit pas ; on passe a l'opération suivante

La 3° opération est la multiplication:

Entrer 2 nombres :.....

..... * = ?(entrer le résultat).....

s'il réussit on le félicite et on lui donne +5 sinon on lui une 2° et 3° chance sinon on passe a l'opération suivante :

la 4° opération est la division :

la même fenêtre est présenté au candidat avec le même raisonnement

:question ? réponseetc. .

la 5° opération est de géométrie :

surface du carrée .Donner le cote d'un carrée :.....

La surface du carrée de cote= ?(donner le résultat):.....

S'il la réponse est juste on lui donne :+5 et on passe a la question suivante ; sinon on lui donne au max. 3 autres chances ; pour chaque réponse fausse , on

lui affecte : -1 ; sinon on passe a la question suivante .

Surface du rectangle . Entrer les données :.....

La surface est égale a :

La même technique est utilisé pour évaluer le candidat : +5 si juste , -1 si fausse et 3 chances.

Surface du cercle :Entrer le rayon du cercle :.....

La surface du cercle =

(+5 si juste , -1 si fausse et pas plus de 3 chances aux max. sinon Fin du Test)

K Fin du test Mr.....votre résultat est :.....:

(on doit donner la somme des points positifs (+5) mois la somme des points négatifs comme score final du Test)

Questions : 1° Dessiner l'algorithme d'une manière simple sous forme de pas a pas *خطوة خطوة* (1°..2°..3°...4°...etc. sans organigramme) , ayant comme objectif de traduire chaque pas (step) par une fonction ou procédure .

2° Déclarer les procédures ou fonctions nécessaires dans une UNIT.

3° Écrire le programme principale qui appel les procédures et fonctions de l'UNIT pour exécuter chaque partie du test, sans oublier de déclarer les paramètres effectifs des procédures et fonctions et les variables globales nécessaires .

Votre programme doit s'exécuter selon le choix de l'utilisateur et se répéter selon son choix et selon le Model qu'on vous a proposé plus haut .

Problème n°2 :(40/100) Un laboratoire de contrôle de qualité vous a demandé un programme de calcul statistique ; les calculs les plus fréquents sont la moyenne , la déviation standard **de N test** exprimée **en nombre réel** et le tri des N tests .

La moyenne de n test se calcul selon la formule suivante :

$$\text{Moyenne (x)} = \frac{1}{N} \sum_{i=1}^N x_i$$

La déviation standard est calculée selon la formule suivante :

$$\text{Devstd (x)} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \text{moyenne (x)})^2}$$

Questions : 1° : Écrire les déclarations nécessaires pour un tel programme .

2° : Donner l'algorithme et la fonction qui calcul la moyenne .

3° : Donner l'algo. et la fonction qui retourne la déviation standard .

4° : Donner l'algo. et la procédure de tri par ordre croissant des N

tests .

5° : Écrire le programme principale qui propose ces services (lecture des données et écriture des résultats inclus).

Problème Facultatif (pour gagner quelque point en plus) Donner l'algo. et la procédure qui cherche le minimum et le maximum d'une liste de N éléments de nombre réel .

/حلول بعض التمارين و برامج شتى :

*مثال لإستعمال التعليلة التكرارية . For

{ ce programme permet de faire le passage de la base B a la base décimale; on obtient la valeur z entière en effectuant le calcul du polynôme $F(x)$ pour x comme Base . Ce programme est basé sur l'algorithme appelé shema de

Horner des

mise en facteur partielles. }

Program conversionBaseDecimal;

uses wincrt;

var base, degpoly, i, z, bit : integer;

begin

repeat

z := 0;

write('Donner la base est le degre du polynôme');

readln(base , degpoly);

write('donner le nombre a convertir chaque digit sépare');

for i := 0 To degpoly do

begin

read(bit);

z := Z * base + bit ;

end;

writeln('La valeur du nombre en base ', base, 'est =', z, 'en décimal');

writeln('Autre Run o(ui) ou n(on) ');

Until readkey in ['n', 'N'];

writeln('FFFFFFFFFFFFiiiiNNNNN');

end.

*مثال لإستعمال التعليلة التكرارية . while

(*calcul d'un terme quelconque d'une suite récurrente définie par

```

    U_n2 = U_n+1 + U_n *)
program suite ;
    uses wincrt;
    var n , compte , u0,u1,u : integer;
    begin
        writeln('calcul d'un terme quelconque d'une suite récurrente définie par  $U_n = U_{n+1} + U_n$  ');
        writeln('donner les premiers éléments de la suite U0 et U1');
        readln( u0,u1);    write ('donner le numéro du terme cherche ');
    readln( n );
        compte := 2;
        while compte <= n do
            begin
                u := u0 + u1;    u0 := u1;    u1 := u ;
                compte := compte + 1;
            end;
        writeln('le ',n,'élément de la suite est égale a ',u);
    end.

```

*مثال لإستعمال التعليلة التكرارية . For

```

program suiteFibonacci;
    uses wincrt;
    var fn,fn1,fn2,i,n : integer;
    Begin
        writeln('>>>>>>>>>Suite de Fibonacci ...');
        write('Donner la limite supérieur n :');    readln( n );
        writeln('suite de Fibonacci : des ',n,' nombres');
        fn := 1;writeln('f0 = ',fn);    fn1 := 1;writeln('f1 = ',fn1);
        for i := 2 To n do
            begin
                fn2 := fn + fn1;
            fn := fn1;
                fn1 := fn2;
                writeln('f',i,' = ',fn2);
            end;
        writeln('*****fin*****');
    end.

```

*مثال لبرنامج تعليمي بالعربية يستعمل بنية String

```

program arabic;

```


[illegible]Type déclaré comme indice d'une **إستعمال النوع الإختياري في تذييل القائمة**

array.

```

program TypeDeclare;
    type MT = ( by,ca,vf);
        Lmt = array[MT] of integer;
    var T : MT;      L : LMT;
    procedure lit ( var d : LMT );
        var i : MT ;
        begin
            writeln('exp de table avec indice de type énumère');
            writeln(' entrer 3 donnée');

```

```

    for i := by To vt do readln( d[ i ]);
end;
procedure ecrit( var d : LMT );
    var i : MT;
    Begin
        writeln('les ventes sont:');
        for i := by TO vt do writeln( d[ i ]);
    end;
begin
    lit ( L );    ecrit( L );
end.

```

***برنامج يعرض رزنامة السنة الشمسية مستعملا النوع الإختياري و type déclaré و كيفية قراءة و كتابة هذا النوع.

```

program calendrier;
uses wincrt;
type mois = (janvier,fevrier,mars,avril,mai,juin,juillet,aout,
    septembre,octobre,novembre,decembre);
jours =(samedi,dimanche,lundi,mardi,mercredi,jeudi,vendredi);
var M : mois;
    premierJ,drenierj : jours;
    Annee,finMois,Date, mm , jj : Integer;
procedure litDonnee;
begin
    writeln('Lecture des Données');
    write('Donner l'annee : ');readln( annee );
    write('Donner le premier mois');
    writeln(' De 1..12 pour Janvier à Décembre'); readln( mm );
    case mm of
        1 : M := janvier ;
        2 : M := Fevrier ;
        3 : M := Mars;
        4 : M := Avril ;
        5 : M := Mai ;
        6 : M := Juin;
        7 : M := Juillet ;
        8 : M := Aout ;
        9 : M := Septembre ;

```

```

10 : M := Octobre ;
11 : M := Novembre ;
12 : M := Decembre ;
end;
write('Donner le premier Jour');
writeln('De 1 .. 7 pour Samedi à Vendredi'); readln( jj );
case jj of
1 : premierJ := samedi ;
2 : premierJ := dimanche ;
3 : premierJ := lundi ;
4 : premierJ := mardi ;
5 : premierJ := mercredi ;
6 : premierJ := jeudi ;
7 : premierJ := vendredi ;
end;
end;
procedure longMois;
begin
    case M of
        janvier,mars,mai,juillet,aout,octobre,
        decembre : Finmois := 31;
        avril,juin,septembre,novembre:
            finmois := 30 ;
        fevrier : If (annee MOD 4 = 0) then
            finmois := 29
            else finmois := 28;
        end;{ fin case M}
    end;
end;
Procedure ecritMois;
begin
    case M of
        janvier : writeln('Janvier');
        fevrier : Writeln('Fevrier');
        mars : writeln('Mars');
        avril : writeln('Avril');
        Mai : writeln('Mai');
        juin : writeln('Juin');
        juillet : writeln('Juillet');
    end;
end;

```

```

        aout : Writeln('aout');
        septembre : writeln ('septembre');
        octobre : writeln('Octobre');
        novembre : writeln('Novembre');
        decembre : writeln('Decembre ');
    end;
end;
procedure calcul;
begin
    date := 1;
    while date <= finmois do
        begin
            if premierJ <= vendredi Then
                begin
                    case premierJ of
                        samedi : write('Samedi':15);
                        dimanche :write('Dimanche');
                        lundi : Write('Lundi':15);
                        mardi : write('Mardi':15);
                        mercredi:write('Mercredi':15);
                        jeudi:write('Jeudi':15);
                        vendredi:write('Vendredi':15);
                    end;{case}
                    writeln(date:5);
                    date := succ(date);
                    premierJ :=succ(premierJ);
                end {then}
            else
                premierJ := samedi;
            end; {while}
            if premierJ = drenierj Then premierj := samedi;
        end;
Begin { programme principal}
    litDonnee; writeln('CALENDRIER DE L"ANNEE ',Annee);
    for M := janvier To decembre Do
        begin
            longmois;  ecritmois;  calcul;          ecritmois;
            writeln('CALENDRIER DE L"ANNÉE ',Annee);

```

```

        writeln('Taper sur Entrer pour Continuer ..');readln;
    end;
    writeln('Fin de L"Annee...',annee);
end. { fin prog.Prin }

```

**.وحدة تحتوي على إجراءات ترتيب قائمة .

Unité avec des Procédures de Tri.

```

unit unitext;
interface
    const n = 10 ;
    type table = array[0.. n ] of real ; {or of string}
    var V : table ; dataIn, dataOut , dataTrie : text ;
    procedure pause ;
    procedure lit( var dataIn : text ; var t : table ) ;
    procedure ecrit( var dataTrie : text ; var v : table ) ;
    procedure swap( var x , y : real );
    procedure bubble2( var v : table ) ;
    procedure bubble3( var v : table ) ;
    procedure bubble ( var v : table ) ;
    procedure bbsort( var v : table );
    procedure triSelection( var V : table ) ;
    procedure triComptage( var v : table );
    procedure triInsertion( var v : table );
    procedure triTransposition ( var v : table ) ;
    procedure quickSort( var v : table ) ;
    procedure ecritext( var data : text ) ;
implementation
    procedure lit( var dataIn : text ; var t : table ) ;
    var i : integer;
    begin
        writeln('Reading of data from text file');
        i := 1;
        while not eof( dataIn ) and (I <= 10) do
            begin
                readln( dataIn , T[I]);
                i := i + 1;
            end;
        writeln('la liste a trier est ');
    end;

```

```

        for i := 1 to n do writeln(T[i]:5);
    end;
    procedure pause;
    begin writeln(' taper entrer pour continuer ');    readln ;    end;
    procedure swap( var x , y : real);
        var z : real;
        begin            z := x ; x := y ; y := z;            end;
    procedure bubble ( var v : table);
        var i , j : integer ;            trie : boolean ;
        begin
            repeat
                trie := true ;
                for i := 1 To n-1 do
                    if v[i] > v[i+1] then
                        begin            swap(v[i],v[i+1] );            trie := false ; end
                until trie ;
            end;
    procedure bubble2( var v : table );
        var echange : boolean; i , m : integer;
        begin
            echange := true;            m := n ;
            while ( m >= 2 ) and (echange = true ) do
                begin
                    echange := false;
                    for I := 1 to (m-1) do
                        if v[i] > v[i+1] then
                            begin            swap(v[i],v[i+1]);            echange := true;
                    end;
                    m := m - 1;
                end;
            End;
    procedure bubble3( var v : table );
        var echange : boolean; i , m,w : integer;
        begin
            echange := true ;            m := n ;
            while ( m >= 2 ) and (echange = true ) do
                begin
                    echange := false;

```

```

        for I := 1 to (m-1) do
            if v[i] > v[i+1] then
                begin
                    swap(v[i],v[i+1] );
                    echange := true;
                    w := I;
                End;
            m := w;
        End;
    End;
procedure ecrit( var datatrie : text ;var v : table);
    var i : integer;
    begin
        for i := 1 to n do writeln(datatrie , V[i]:5:2 );
    end;
procedure ecrittext( var data : text );
    var r : string[20];
    begin
        reset( data );
        while not eof ( data ) do
            begin      readln ( data , r );      writeln( r );      end;
        close ( data );
    end;
procedure bbsort( var v : table);
    var i,j : integer;
    begin
        for i := 2 to n do
            for j := n downto i do
                if v[j-1] > v[j] then
                    swap(v[j-1],v[j] );
            end;
        end;
procedure triSelection( var V : table);
    var i , j ,indicePP : integer; pp : real;
    begin
        for i := 1 to ( n - 1) do
            begin
                PP := V [I];
                indicePP := I ;

```

```

                                for j := i+1 To N do
                                    if pp < v[j] then
                                        else
                                            begin
                                                pp := v[j];   indicePP := j ;
                                                V[indicepp] := V[I];   V[i] := pp;
                                            end;
                                End;
                            End;
procedure triComptage( var v : table);
    var vtrie : table; index : array[1..n] of integer;
        i , j : integer;
    begin
        for i := 1 to n do index[i] := 0;
        for i := 1 to n do
            for j := 1 to n do
                if v[i] >= v[j] then
                    index[i] := index[i]+ 1;
            for i := 1 to n do      vtrie[index[i]] := v[I];
            v := vtrie ;
        end;
procedure triInsertion( var v : table);
    var i , k : integer;
    begin
        for i := 2 to n do
            begin
                v[0] := v[I];          k := i-1;
                while ( v[k] > v[0]) and (k >= 0) do
                    begin v[k+1] := v[k]; k := k - 1;end;
                v[k+1] := v[0];
            end;
        End;
procedure triTransposition ( var v : table);
    var i,j : integer;
    begin
        i := 1;
        while i < n do
            begin
                if V[i] > V[i+1] then{permutation d element}

```



```

        begin
        swap(v[i],v[i+1]);
        j := i - 1;
        repeat {boucle de retour en arriere}
        if v[j] > v[j+1] then
        begin
        swap(v[j],v[j+1]);
        j := j - 1;
        end
        Else {si classement non altere sort}
        j := 0;
        until j < 1;
        end;{ fin du premier then}
        i := i + 1;{ continue passage}
        end; { fin du while}
        end;{fin procedure triTran}
    procedure quickSort( var v : table);
    procedure sort( l , r : integer);
        var i , j : integer;
            x,w : real;
    begin
        i := l ; j := r;
        x := V[((l + r ) div 2) ];
        Repeat
            while V[i] < x Do i := i + 1;
            while x < V[j] Do j := j - 1;
            If i <= j Then
                begin
                swap(v[i],v[j ] );
                i := i + 1;j := j - 1;
                end;
            Until i > j;
            If l < j Then Sort( l , j);
            if i < r Then Sort ( i , r );
        end;{fin sort}
    begin{quicksort}
        Sort( l , n );
    end;{quicksort}

```

```

END.{ fin unite }

program Tri;
uses winCrt , Unitext;
var T : table; dataIn ,dataOut, dataTrie : text ;choix : char;
BEGIN
    repeat
        writeln('ceci est un programme de tri de 10 noms');
        writeln('les algorithmes suivants sont utilises');
        writeln('1:tri a bulle');
        writeln('2:tri a bulle autre version');
        writeln('3:tri a bulle ameliore');
        writeln('4:tri a bulle avance');
        writeln('5:tri par selection');
        writeln('6:tri par insertion');
        writeln('7:tri par comptage');
        writeln('8:tri par transposition');
        writeln('Q:tri Rapide');
        write('Faite votre choix '); readln( choix );
        assign ( dataIn ,'dataIn.txt');
        reset ( dataIn ); LIT ( dataIn , t );
        close ( dataIn );
        case choix of
            '1' : bbsort( T );
            '2' : bubble( T );
            '3' : bubble2( T );
            '4' : bubble3( T );
            '5' : TriSelection( T );
            '6' :TriComptage( T );
            '7' : TrInsertion( T );
            '8' : TriTransposition( T );
            'q' : QuickSort( T );
        end;
        writeln('le résultat du tri est dans le fichier trie.txt');
        assign( datatrie ,'trie.txt'); rewrite ( datatrie );
        ECRIT( datatrie , T );close ( datatrie );
        writeln( 'le contenu du fichier datatrie:trie.txt ');
        writext ( datatrie );

```

```
writeln(' Autre Execution O(ui) ou N(on) ');
```

```
until READKEY in ['n','N'];
```

```
END.
```

**** بعض الدوال والإجراءات التراجعية**

Procédures et Fonctions Récursives

```
program TestFibonacci ; { $N+ }
```

```
uses wincrt ;
```

```
var n : extended;
```

```
function Fibonacci (Which: extended) : extended;
```

```
begin
```

```
if (Which=1) or (Which=2)
```

```
then Fibonacci := 1
```

```
else Fibonacci := Fibonacci(Which - 1) + Fibonacci(Which - 2) ;
```

```
end; { Fibonacci }
```

```
begin
```

```
writeln('Fibonacci series : with recursive Function ');
```

```
write('Try a number ? :');readln( N );
```

```
Writeln('Here is the Fibonacci Number :',fibonacci( n ));
```

```
end.
```

```
program Hanoi; { Recursively solves the Towers of Hanoi problem. Moves  
disks from A to C. }
```

```
uses wincrt ;
```

```
var Height: integer;
```

```
procedure Move (Height: integer; FromPeg, ToPeg, UsingPeg: char);
```

```
{ Recursive procedure for determining moves. Keep this order-from,  
to, using-in mind when you read the recursive calls. }
```

```
begin
```

```
if Height = 1
```

```
then begin write ('Move a disk from '); write (FromPeg);
```

```
write (' to ');write (ToPeg); writeln end
```

```
else begin Move (Height - 1, FromPeg, UsingPeg, ToPeg);
```

```
write ('Move a disk from ');write (FromPeg); write (' to ');
```

```
write (ToPeg); writeln;
```

```
Move (Height - 1, UsingPeg, ToPeg, FromPeg)
```

```
end { if }
```

```
end; { Move }
```

```

begin
  write ('How many disks are you going to start with? ');
  read (Height);
  Move (Height, 'A', 'C', 'B')
end. {Hanoi}
program Print;
  {Prints a sentence to demonstrate end recursion.}
  uses wincrt ;
procedure Echo;
  var TheCharacter: char;
  begin
    read (TheCharacter); write (TheCharacter);
    if TheCharacter <> '.' then Echo ;
  end; {Echo}
begin
  write ('Type in a sentence that ends with a period. ');
  writeln; Echo; writeln
end. {Print}
program Recur;
  {Uses recursion to read a line and echo it in reverse.}
  uses wincrt ;
procedure StackTheCharacters;
  var TheCharacter: char;
  begin
    read (TheCharacter);
    if not eoln then StackTheCharacters {recursive call.} ;
    write (TheCharacter)
  end; {StackTheCharacters}
begin
  writeln ('Enter a sentence that is not a palindrome. ');
  StackTheCharacters; {The first call.}
  writeln
end. {Recur}
program Summer;
  {Uses a recursive function for addition.}
  uses wincrt ;
  var Bound: integer;
function Sum (Limit: integer) : integer;

```

```

{Recursively sums the positive numbers 1 through Limit.}
begin
  if Limit = 1 then Sum := Limit
  else Sum := Limit + Sum(Limit - 1)
end; {Sum}
begin
  write ('I add 1..n the hard way. What's n? ');
  read (Bound);
  write ('The sum is: ');
  writeln (Sum (Bound) )
end. {Summer}
program UnDigit; ;{Recursively reverses the digits of a positive integer.}
uses wincrt;
var Data: integer;
procedure ReverseDigits (Number: integer);
begin
  write (Number mod 10); { write the `ones' digit.}
  {If there are more digits, divide away the `ones' digit and pass the result.}
  if (Number div 10) <> 0 then ReverseDigits (Number div 10)
end; {ReverseDigits}
begin
  writeln ('Please enter a positive integer. ');
  read (Data);
  ReverseDigits (Data);
  writeln
end. {UnDigit}

```

*جزأ من حل التمرين 7: حول الإجراءات و الدوال

```

program revisionGeneral;
uses wincrt,mathunit;
var x,y , z : real;
procedure calcul_arithmetique;forward;
procedure calcul_Trigonometrique;forward;
procedure calcul_Geometrique;forward;
procedure Fonction_divers ; Forward;
procedure MenuPrincipal;
  var c : char;
  begin
    clrscr;

```

```

        writeln('Menu Principal');
        writeln('1:calcul Arithmetique ');
        writeln('2:calcul Trigonometrique ');
        writeln('3:calcul Geometrique');
        writeln('4:Fonction Divers');
        writeln('0:Sortir du Menu Principal');
        Readln( c );
        case c of
            '1':calcul_Arithmetique;
            '2':calcul_Trigonometrique;
            '3':calcul_Geometrique ;
            '4':Fonction_Divers;
            '0':exit;
            else writeln ('choix errone');
        end;
    end;
procedure litxy;
    begin write('entrer 2 Nb :'); readln( x ,y ); end;
procedure subtract;
    begin writeln(x:5:2,'-',y:5:2,'=',x-y:8:3); end;
procedure add;
    begin writeln(x:5:2,'+',y:5:2,'=',x+y:5:2); end;
procedure calcul_Arithmetique;
    var c : char;
    begin
        writeln('Menu Calcul Arithmetique ');
        writeln('+:addition ');
        writeln('-:soustraction');
        writeln('0:exit');
        readln( c );
        case c of
            '+':begin litxy ;add; end;
            '-':begin litxy ;subtract;end;
            'x':menuPrincipal;
            '0':exit;
            else writeln('choix inconnu ');
        end;
    end;
end;

```

```

    procedure litx;
    begin write('entrer un Nb:');readln( x );    end;
procedure calcul_Trigonometrique;
var c : char;
begin
    writeln(' Menu Calcul Geometrique ');
    writeln('s: sinus d"angle ');
    writeln('c: cosinus .....');
    writeln('x:retour au Menu Principal ');
    readln( c );
    case c of
    's' :begin litx;
        writeln( 'le sinus de l"angle est ',sin(x));
        end;
    'c':begin litx;
        writeln('cos de ',x,'est=',cos(x):5:2);
        end;
    'x':menuPrincipal;
    else writeln('operation inconnue');
    end;
end;
procedure Scarre;
begin
    writeln('la surface du carree est',sqr( x ):5);
end;
procedure Srect;
begin
    writeln('la surface de rectangle est:',x * y:8:3);
end;
procedure Scercle;
begin
    writeln('la surface du cercle est =',pi*sqr( x ));
end;
procedure calcul_Geometrique ;
var c : char;
begin
    writeln('Menu Calcul Géométrie ');
    writeln('1:surface du carre ');

```

```

        writeln('2:.....rectangle ');
        writeln('3:.....du cercle');
        writeln('x:retour au menu Principal');
        readln( c );
        case c of
            '1':begin litx ;Scarre; end;
            '2':begin litxy ;Srect; end;
            '3':begin litx ;Scercle; end;
            'x':MenuPrincipal;
            else writeln(' choix inconnu ');
            end;
        end;
    procedure powerxn ;
        var x , n : integer ;
        begin
            write('donner x et n 2 nb entier positif ');
            readln( x , n );
            writeln('la puissance de x a n est=',exp(n*ln(x)));
        end;
    Procedure Fonction_Divers ;
        var c : char ;
        begin
            writeln(' Menu Fonction Divers ');
            writeln('1:Power de x a n comme procedure');
            writeln('2:power of x to n as function ');
            writeln('3:reciproque of argument as function');
            writeln('x:Retour au menu Principal');
            readln( c );
            case c of
                '1':PowerXn;
                '2':begin litxy ;
                    writeln('x a la puissance y=',power( x,y));end;
                '3':begin litx ;
                    writeln('la recip. de ',x:5:2,'est=',recip(x));end;
                'x':menuPrincipal;
                else writeln(' operation inconnu ');
            end;
        end;
end;

```



```

function arret:boolean;
  var c : char;
  begin
    write(' une autre exécution o(ui) ou n(on)');
    readln( c );
    arret := c in ['n','N'];
  end;
Begin { prog. principal}
  repeat
    menuPrincipal;
  until arret;          clrscr;
  writeln('fin du programme : au revoir ');
end.{ fin du prog }

```

*أمثلة لدوال رياضية مصرح بها في وحدة .

```

unit math;
  interface
  function    power (x,n:real):real;
  function    log (base, argument:real):real;
  function    recip (argument:real):real;
  function    deg2rad (angle:real):real;
  function    rad2deg (angle:real):real;
  function    tan (angle:real):real;
  function    sec (angle:real):real;
  function    csc (angle:real):real;
  function    cot (angle:real):real;
  function    arcsin (a:real):real;
  function    arccos (a:real):real;
  function    arcsec (a:real):real;
  function    arccsc (a:real):real;
  function    arccot (a:real):real;
  implementation
  function recip (argument:real):real;
  begin
    if argument = 0 then
      writeln ('Can not take the reciprocal of 0') else
      recip:=(1/argument);
    end;
  function log (base,argument:real):real;

```

```

begin
  if (base <= 0) or (argument <= 0) then
    writeln ('Can not take the log of a non-positive number') else
      log:= ln (argument) / ln (base);
  end;
function power (x,n:real):real;
  begin
    if x<=0 then
      writeln ('Base must be positive') else
        power:= exp (n * ln (x));
    end;
function deg2rad (angle:real):real;
  begin      deg2rad:= angle * pi/180; end;
function rad2deg (angle:real):real;
  begin      rad2deg:= angle * 180/pi;end;
function other (a:real):real;
  begin      other:= sqrt (1- sqr (a));end;
function others (a:real):real;
  begin      others:= sqrt (sqr (a) - 1);end;
function tan (angle:real):real;
  begin
    if (angle = pi/2) or (angle = 3*pi/2) then
      writeln ('Tangent is undefined at 90 and 270 degrees') else
        tan:= (sin (angle)/cos (angle));
    end;
function sec (angle:real):real;
  begin
    if angle = 90 then  writeln ('Secant is undefined at 90')
    else  sec:= recip (cos (angle));
  end;
function csc (angle:real):real;
  begin
    if angle = 0 then  writeln ('Cosecant is undefined at 0')
    else  csc:= recip (sin (angle));
  end;
function cot (angle:real):real;
  begin
    if (angle = 0) or (angle = pi) then

```

```

        writeln ('Cotangent is undefined at 0 and 180')
        else   cot:= recip (tan (angle));
    end;
function arcsin (a:real):real;
begin
    if a > 1 then writeln ('Value out of range')
        else   arcsin:= arctan (a/other (a));
    end;
function arccos (a:real):real;
begin
    if a > 1 then   writeln ('Value out of range')
        else   arccos:= arctan (other (a)/a);
    end;
function arccsc (a:real):real;
begin
    if a < 1 then   writeln ('Value out of range')
        else   arccsc:= arctan (1/others (a));
    end;
function arcsec (a:real):real;
begin
    if a < 1 then   writeln ('Value out of range')
        else   arcsec:= arctan (others(a));
    end;
function arccot (a:real):real;
begin
    arccot:= arctan (recip (a));end;
end.{ fin Unit}

```

****حل تمارين و مسألة الإمتحان حول الإجراءات و الجداول**

:Ex.sur les Procédures et Fonctions et les array's

```

program test_evaluation ;
    uses wincrt, mathlibr;
    var x , y , z , total : real ; nom : string[30];
        repOk : boolean; n : integer ;
    begin {debut test_evaluation}

        writeln('مرحبا بك إلى برنامج تقييم معلوماتك في الرياضيات');

        writeln('سوف أطرح عليك الأسئلة و أطلب منك إدخال البيانات و الأجوبة');
    end

```

```

writeln('و أرد عليك إذا كان جوابك صحيحا أمتحك +5 أو -1 إذا كان خطأ');
writeln('و في الأخير أعطيك النتيجة النهائية');
write('votre nom est :');readln( nom );
writeln('هل أنت مستعد للبدأ إضرب على ن لنعم و إلا على أي حرف آخر');
while readkey in [ 'ن','و','O'] do
    begin { debut du while}
    total := 0; {initialiser la variable total pour enregistrer les resultats}
    {dedut de l'opération addition}
    n := 1 ;{initialiser le compteur des 3 chances} clrscr;
    repeat { debut de +}
        writeln('العملية الأولى هي الجمع');
        write('أدخل عددين entrer 2 nombres :');
        lixy( x , y );
        write( x:5:1,'+',y:5:1 , '=...?');liz( z );
        if z = som ( x , y ) then
            begin
                writeln('أحسنت جوابك صحيح +5');
                repOk := true ;      total := total + 5;
            end
        else
            begin
                writeln('معدرة جوابك خطأ -1');
                total := total -1 ;   repOk := false;
                n := n + 1 ;
                if n <= 3 then writeln(n,'°chance');
            end;
    until ( repOk ) or ( n > 3 ) ; {fin de +}
    writeln('fin de l'operation d"addition');
    {debut operation soustraction } ;
    n := 1 ;{initialiser le compteur des 3 chances }
    repeat { début de -}
        writeln; writeln('2° opération la soustraction ');

```

```

write('أدخل عددين:');  lixy( x , y );
write( x:5:1,'-',y:5:1 ,'=...?');liz( z );
if z = sub ( x , y ) then
    begin
        writeln('reponse juste +5');    repOk := true ;
        total := total + 5;
    end
else
    begin
        writeln('je regrette ta réponse est fausse -1');
        total := total -1 ;    repOk := false;
        n := n + 1 ;
        if n <= 3 then writeln(n,'°chance');
    end;
until ( repOk ) or ( n > 3 ) ; { fin du - }
writeln('fin de l'operation de soustraction');
n := 1 ;{initialiser le compteur des 3 chances }
Repeat { debut de *}
    writeln('3° opération :la multiplication ');
    write('أدخل 2 Nombres:');  lixy( x , y );
    write( x:5:1,'*',y:5:1 ,'=...?');liz( z );
    if z = mul ( x , y ) then
        begin
            writeln('reponse juste : +5');
            repOk := true ;    total := total + 5;
        end
    else
        begin
            writeln('je regrette ta reponse est fausse :-1');
            total := total -1 ;    repOk := false;    n := n + 1 ;
            if n <= 3 then writeln(n,'°chance');
        end;
until ( repOk ) or ( n > 3 ) ; { fin du *}
writeln('fin de l'operation :Multiplication');
n := 1 ;{initialiser le compteur des 3 chances }
Repeat { debout de /}
    writeln;    writeln('4° opération :la Division ');

```

```

write('أدخل عددين entrer 2 Nombres :');
lixxy( x , y ); write( x:5:1,'/',y:5:1 , '=...?');liz( z );
if z = DIVIS ( x , y ) then
    begin
        writeln('reponse juste : +5');
        repOk := true ; total := total + 5;
    end
else
    begin
        writeln('je regrette ta reponse est fausse :-1');
        total := total -1 ; repOk := false;
        n := n + 1 ;
        if n <= 3 then writeln(n,'°chance');
    end;
until ( repOk ) or ( n > 3 ) ; { fin du /}
writeln('fin de l'operation :division');
n := 1 ;{initialiser le compteur des 3 chances }
Repeat { debout de surface du carrée}
    writeln('5° opération :la surface du carrée ');
    write('entrer le coté :'); readln( x );
    write( 'la surface du carrée =...?');liz( z );
    if z = scar ( x ) then
        begin
            writeln('reponse juste : +5'); repOk := true ;
            total := total + 5;
        end
    else
        begin
            writeln('je regrette ta réponse est fausse :-1');
            total := total -1 ; repOk := false;
            n := n + 1 ;
            if n <= 3 then writeln(n,'°chance');
        end;
until ( repOk ) or ( n > 3 ) ; { fin du scar}
writeln('fin de l'operation :surf. car');
n := 1 ;{initialiser le compteur des 3 chances }
Repeat { début de surface du rectangle}
    writeln('6° opération :la surface du rectangle ');

```

```

write('entrer les données :');
lixxy( x , y );
write( 'la surface du rectangle =...?');liz( z );
if z = srec ( x , y ) then
begin
    writeln('reponse juste : +5'); repOk := true ;
    total := total + 5;
end
else
begin
    writeln('je regrette ta réponse est fausse :-1');
    total := total -1 ; repOk := false;
    n := n + 1 ;
    if n <= 3 then writeln(n,'°chance');
end;
until ( repOk ) or ( n > 3 ) ; { fin du srec }
writeln('fin de l'operation :surf. rectangle');
n := 1 ;{initialiser le compteur des 3 chances }
Repeat { début de surface du cercle }
    writeln('7° opération :la surface du cercle ');
    write('entrer le rayonقيمة الشعاع:'); readln( x );
    write( 'la surface du cercle =...?');liz( z );
    if z = scer ( x ) then
    begin
        writeln('reponse juste : +5');
        repOk := true ; total := total + 5;
    end
    else
    begin
        writeln('je regrette ta reponse est fausse :-1');
        total := total -1 ;
        repOk := false;
        n := n + 1 ;
        if n <= 3 then writeln(n,'°chance');
    end;
until ( repOk ) or ( n > 3 ) ; { fin du scercle }
writeln('fin de l'operation :surf. cercle');
clrscr;cursor( 30,5);

```

```

writeln('إنتهى تنفيذ ميمك');
writeln('Mr.',nom);
writeln( 'Votre resultat final est النتيجة النهائية هي ',total:3:1 );
writeln('fin du prog ن لنتفيذ البرنامج مرة أخرى إضرب على ');
end;{ fin du while }      clrscr;
writeln('fin du programme au Revoir إنتهى البرنامج مع السلامة');
end.{ fin du prog.}
unit mathlibr ;
interface
    const N = 10 ;
    type list = array[1..N] of real;
    var L : list;
    Function  Moyenne( var L : list ):real ;{Pb N°2}
    Function  dev_std ( var L : list ):real;

    {declaration du Pb. N°1}
    function  som ( x , y : real ) : real;
    function  sub ( x, y : real ): real;
    function  mul ( x , y : real ) : real;
    function  divis( x , y : real ) : real;
    function  scar ( x : real ) : real;
    function  srec ( x , y : real ) : real ;
    function  scer ( x : real ) : real ;
    procedure lixy ( var x , y : real ) ;
    procedure liz( var z : real ) ;
    procedure lilist( var l : list );
    procedure ecritlist( var l : list);
    procedure minMax ( var L : list );
implementation
    Function Moyenne ( var L : list ):real;
        var i : integer; Som : real;
        begin
            som := 0 ;
            for i := 1 to N do      som := L[i] + som ;
            moyenne := som / N ;
        end;

```



```

Function dev_std ( var L : list ):real;
    var i : integer ; Dv : real;;
    begin
        dv := 0;
        for i := 1 to n do
            dv := dv + Sqr( L[i] - moyenne(L);
            dv := Sqrt( dv / ( n - 1));
        dev_std := dv;
    end;
procedure lilist( var l : list );
    var i : integer ;
    begin
        randomize;{générateur de Nombre aléatoire}
        for i := 1 to N do L[i] := random(100);
        {met un nombre quelconque dans L[i] : Façon pratique pour tester}
    end;
procedure ecritlist( var l : list);
    var i : integer;
    begin
        writeln('contenu de la liste :');
        for i := 1 to N do
            begin
                write(L[i]:10:1);
                if i mod 5 = 0 then writeln ;
            end;
        end;
procedure minMax ( var L : list );
    var i : integer; min, max : real ;
    begin
        min := L[1];max := min ;
        for i := 2 to N do
            begin
                if min < L[i] then {vide}      else min := L[i];
                if max > L[i] then {vide}      else max := L[i];
            end;
        writeln('le minimum de la liste est : ',min:10:1);
        writeln('le maximum de la liste est : ',max:10:1);
    end;

```

```

end;
procedure lixy ( var x , y : real );
begin      readln( x , y );      end;
procedure liz ( var z : real );
begin      readln( z );      end;
function som ( x , y : real ) : real ;
begin      som := x + y ;      end;
function sub ( x , y : real ) : real ;
begin      sub := x - y ;      end;
function mul ( x , y : real ) : real ;
begin      mul := x * y ;      end;
function divis ( x , y : real ) : real;
begin      if y = 0 then      begin
                writeln('القسمة على صفر مستحيلة'); end
            else divis := x / y ;
        end;
function scar ( x : real ) : real;
begin      scar := sqr ( x ) ;      end;
function srec ( x , y : real ) : real ;
begin      srec := x * y ;      end;
function scer ( x : real ) : real ;
begin      scer := 3.14 * sqr ( x ) ;      end;
end.{ fin de l'unite mathlibrary}
program PbN2;
uses wincrt, mathlibr;
var L : list ;
begin      lilist( L );      writeln('La liste est :');      ecritlist( L );
            writeln('La moyenne des ',N,'Nombres est',Moyenne( L ):10:1 );
            writeln( ' La déviation Standard est = a ', dev_std( L ):10:1 );
            writeln('Fin Pb. N°2' );
            writeln('Taper sur entrer pour voir la solution du Pb.Facultatif ');
            readln;      clrscr;
            Writeln('Solution du Problème Facultatif : Procédure MinMax d'une
liste');
            lilist( L );
            writeln('La liste est ');
            ecritlist ( L );
            minmax( L );

```

end.

****حل التمرين رقم 09: حول المجموعات**

Ex. sur les set(ensembles)

```

program setProbleme;
  uses wincrt;
  type ensb = set of 1..100 ;
  var E : ensb ;    c : char ;
  procedure litset ( var E : ensb );
    var c : integer ;
    begin      E := [ ] ;
      repeat
        writeln('entrer un élément de l'ensemble : un nombre de 1 a
100');
        readln( c ) ;    E := E + [ c ] ;
        writeln(' autre élément o(ui) ou n(on)');
        until readkey in ['N','n'];
      end;
  procedure ecritset ( var e : ensb );
    var i : integer;
    begin
      writeln('les éléments de l'ensemble sont :');
      for i := 1 to 100 do   if i in E then write(i:5);  writeln;
    end;
  procedure litsetin ( var e1 : ensb );
    var i : integer;
    begin
      writeln('تكوين المجموعة داخليا');  randomize;
      e1 := [ ];
      for i := 1 to 10 do
        E1 := E1 + [random(100)] ;
      end;
  procedure union ;
    var e,f,t,u : ensb ;
    begin
      writeln('Union de 2 ensembles عملية اتحاد مجموعتين');
      writeln(' المجموعة الأولى هي '); litset( e ); ecritset ( e );
      writeln(' المجموعة الثانية هي : '); litset( f ); ecritset( f );

```

```

writeln('أدخل عناصر المجموعة المتحدتين الأولى والثانية');
t := e + f ;      litset( u);
if u = t then writeln('أحسنست جوابك صحيح')
    else writeln('أخطأت جوابك غير صحيح');
writeln('L'union des 2 ensb. est :');ecritset( T );
end;
procedure union2 ;
var e,f,t,u : ensb ;
begin
    writeln('Union de 2 ensembles عملية اتحاد مجموعتين');
    writeln('الجمموعة الأولى هي ');litsetin( e );ecritset ( e );
    writeln(' : الجمموعة الثانية هي ');litsetin( f );ecritset( f );
    writeln('أدخل عناصر المجموعة المتحدتين الأولى والثانية');
    t := e + f ;      litsetin( u);
    if u = t then writeln('أحسنست جوابك صحيح')
        else writeln('أخطأت جوابك غير صحيح');
    writeln('L'union des 2 ensb. est :');ecritset( T );
end;
procedure inter ;
var e , f , T , I : ensb ;
begin
    writeln('intersection de 2 ensb. تقاطع مجموعتين');
    writeln('الجمموعة الأولى هي ');litset( e );ecritset ( e );
    writeln(' : الجمموعة الثانية هي '); litset( f );ecritset( f );
    writeln('entrer les éléments de l'ensb.intersection');
    litset ( I ) ;  T := E * F ;
    if I = T then writeln('أحسنست جوابك صحيح')
        else writeln('أخطأت جوابك غير صحيح');
    writeln('L'intersection des 2 ensb = ');ecritset ( T );
end;
procedure dif ;
var e , f , T , I : ensb ;
begin
    writeln('difference entre 2 ensembles الفرق بين مجموعتين');

```

```

writeln('المجموعة الأولى هي '); litset( e ); ecritset ( e );
writeln('المجموعة الثانية هي '); litset( f ); ecritset( f );
writeln('entrer les éléments de l'ensb.apres différence de e1-e2');
litset ( I ) ; T := E - F ;
if I = T then writeln('أحسننت جوابك صحيح')
else writeln('أخطأت جوابك غير صحيح');
writeln('La différence des 2 ensb est : '); ecritset ( T );
end;
procedure equal_dif ;
var rep : char ; E , F : ensb;
begin
writeln('المقارنة بين مجموعتين بالتساوي أو الاختلاف');
writeln('المجموعة الأولى هي '); litset( e ); ecritset ( e );
writeln('المجموعة الثانية هي '); litset( f ); ecritset( f );
writeln('Le 1° ensb est e(gal) ou d(iffere) du 2° ensb ');
readln( rep );
case rep of
'e' : if E = F then writeln('Ok rep.juste')
else writeln('Rep fausse');
'd' : if E <> F then writeln('5/5 أحسننت ')
else writeln('1-أخطأت');
else writeln('Choix errone');
end;
end;
procedure sub_super;
var rep : char ; E , F : ensb;
begin
writeln('subset or supersetالمجموعة الأولى هل هي أكبر أو تحت المجموعة الثانية');
writeln('المجموعة الأولى هي '); litset( e ); ecritset ( e );
writeln('المجموعة الثانية هي '); litset( f ); ecritset( f );
writeln('Le 1° ensb est I(nclu) ou C(ontient) le 2 ensb ');
readln( rep );
case rep of
'I','i' : if E <= F then
begin
writeln('Ok rep.juste');

```

```

        writeln('E1 est inclu dans E2');
        end
        else writeln('Rep fausse');
    'C','c' : if E >= F then
        begin
            writeln('5/5 أحسنت '); writeln('E1 contient E2 ');
        end
        else writeln('1- أخطاء');
    else writeln('Choix errone');
end;
end;

begin{ prog setprobleme}
    repeat
        clrscr;      writeln('Problème sur les set ');
        writeln('ce pb.vous propose les opérations suivantes : ');
        writeln('1 : former vous même un ens');
        writeln('2 : voir les éléments de l'ens');
        writeln('3 : former un ensb par un générateur de nb.aleatoire interne');
        writeln('4 : Union de 2 ensb.الإتحاد بين مجموعتين');
        writeln('5 : intersection de 2 ensb.تقاطع مجموعتين');
        writeln('6 : Difference entre 2 ensb.الفرق بين مجموعتين');
        writeln('7 : Egalite ou difference entre 2 Ensب.التساوي أو الاختلاف بين مجموعتين');
        writeln('8 : superset or subset المجموعة التحتية و العليا');
        writeln('9 : former un ensb par un générateur de nb.aleatoire interne');
        writeln('a : Union de 2 ensb.interne الإتحاد بين مجموعتين');
        writeln('0 : to exit ');
        writeln('Faites votre choix إختار العملية التي تريد ');
        readln( c ); clrscr;
        case c of
            '1' :begin litset( e ); ecritset( e ); end;
            '2' : ecritset( e );
            '3' : begin litsetin( e ); ecritset ( e ); end;
            '4' : union;
            '5' : inter ;
            '6' : dif ;
            '7' : equal_dif ;

```

```

      '8' : sub_super ;
      '9' : begin litsetin( e ); ecritset ( e ); end;
      'a' : union2;
      '0' : break;
      else writeln('choix errone');
      end;
      writeln('Another run [y]es or [N]o');
      until readkey in ['n','N'];
      writeln('FFFFFFFFFFFFFFFFiiiiiiiiiiNNNNNNNNNNNNNNNNNN');
end.

```

*مثال لكيفية استعمال الجذاذة النصية Utilisation des Fichiers Textes :إحداث جذاذة

نصية ,الإضافة فيها , عرضها وطبعها. Creation ,Ajout,Ecriture et Impression de
fichier Text.

```

unit unittext;
interface
  type livre = text ;
  var page : livre ; lst : text;
  procedure liText ;
  FUNCTION ARRET : boolean;
  procedure ajout;
  procedure edit;
  procedure imprime;
implementation
  procedure liText ;
  var x : string;
  begin
    rewrite ( page );  writeln(' entrer ligne / ligne ');
    repeat
      readln(x );  writeln(page , x );
    until arret;
    close( page );
  end;
  procedure edit;
  var x : string;
  begin
    reset( page );

```

```

        while not eof( page ) do
            begin
                readln(page , x );   writeln( x );
            end;
        close ( page );
        end;
    procedure imprime;
    var x : string;
    begin
        reset( page );rewrite(lst);
        while not eof( page ) do
begin readln(page , x );   writeln(lst, x ); end;
            close ( page );close( lst);
        end;

    procedure ajout;
        var x : string;
        begin
            reset( page );   writeln(' procédure d"ajout :ajouter ligne/ligne');
            repeat
                { to go to the end of the file }append(page );
                readln( x );
                writeln(page , x );
            until arret;
            close ( page );
        end;
    function arret ;
        var c : char ;
        begin
            writeln('autre fois o(ui) ou n(on)'); readln ( c );
            arret := c in ['n','N'];
        end;
    end.
program fichText;
    uses wincrt, unittext,winPrn ;
    var c : char ;
        x : string[10];
begin

```



```

repeat
  writeln(' Mise a joue de fichier text ');
writeln('Nom de fichier :');readln( x );
  assign( page ,x ); assign( lst ,'lpt');
  writeln(' c:creation');   writeln(' a : ajout');   writeln(' i : impression');
  writeln(' e : edition' );
readln( c );
  case c of
    'c' : litext;
    'a' : ajout;
    'e' : edit;
    'i' : imprime ;
  end;
until arret;
writeln('end of up dating of text file ');
end.

```

****برنامج ينقل جذاذة نصية و يدمج جذاذتين في واحدة نصية .**

Copie et fusion de fichiers Text.

```

program fileText;
uses wincrt;
var f1 ,f2 : text;   fn,fn2 : string[15];  x : string[80];
procedure copy (var f1,f2 : text );
begin
  reset( f1 );  rewrite( f2 );
  while not eof ( f1 ) do
    begin
      readln( f1 , x );  writeln( f2 , x );
    end;
  close ( f1 ); close ( f2 );
end;
procedure fusion( var f1 , f2 : text );
begin
  reset( f1 ); reset( f2 );  append( f1 );
  while not eof( f2 ) do
    begin
      readln( f2 ,x );  writeln( f1 , x );
    end;
  close ( f1 ); close ( f2 );
end;

```

```

        end;
begin
    repeat
        writeln('C:opy');    writeln('F:usion');
        case readkey of
            'c' : begin
                writeln('name of f1 :');readln( fn );assign( f1 , fn );
                writeln('name of f2 :');readln( fn2 ); assign( f2 , fn2 );
                copy( f1 , f2 );    end;
            'f' : begin
                writeln('name of f1 :');readln( fn );assign( f1 , fn );
                writeln('name of f2 :');readln( fn ); assign( f2 , fn );
                fusion(f1,f2 );    end;
        end;
        writeln(' run yes or No ');
        until readkey in ['n' , 'N'];
    end.

```

***حل المسألة رقم 5 بإضافة إحداث فهرس للجداذة index on file في الذاكرة المركزية

على شكل قائمة خطية حركية as a linear list .

```

program procFunfile;
    uses wincrt;
    const messagerror =' cumpte  number or record not found';
    type client = record
        numc : integer;
        nomP : string[20];
        adr : string[30];
        op : char;
        mt : real;
        date :string[10];
    end;
    indexPtr = ^index ;
    index = record
        numc : integer ;
        pos : integer ;
        next : indexPtr ;
    end;
    bank = file of client;

```

```

var b : bank ; choix : char; Id ,frontId : indexPtr;
  s : longint; nomFichier : string[25]; c : client ;
procedure index_File( var id : indexPtr );
  BEGIN
    reset( b );
    if not eof( b ) then
      begin
        new ( id );  frontId := id ;
        read( b , c );
        Id^.Numc := c.Numc ;
        Id^.pos := filepos( b );
        Id^.next := nil ;
      end;
    while not eof ( b ) do
      begin
        new ( id^.next ); Id := id^.next;
        read( b , c );
        id^.numC := C.Numc ;
        Id^.pos := filepos( b );
      end;
    Id^.next := nil ;
    close( b );
  end;
procedure see_Index( var Id : indexPtr );
  begin
    id := frontId ;
    while id <> nil do
      begin
        write('num. compte = ',Id^.numc );
        writeln('position dans le fihier = ',Id^.pos );
        id := id^.next ;
      end;
    end;
function cherche_in_index( var id : indexPtr ): integer ;
  var x : integer ; trouve : boolean;
  begin
    writeln('donner le numéro de cpt. a chercher :'); readln( x );
    id := frontId ; trouve := false ;

```

```

while (id <> nil) and ( not trouve ) do
begin
    if x = id^.numc then
    begin
        cherche_in_index := id^.pos ;
        trouve := true ;
        writeln('Num. cpt Trouve dans index = ',Id^.pos);
        end;
        id := id^.next ;
    end;
    if not trouve then
    begin
        cherche_in_index := 0 ;
        writeln('Attention num. compte non trouve ');
        end;
    end;
end;
procedure tailleF ( var b : bank );
begin
    writeln('la taille du fichier2 =', filesize(b ));
end;
procedure lit1client ( var c : client );
begin
    with c do
    begin
        write('num compte:'); readln(numC);
        write('nom prenom:'); readln(nomP);
        write('adresse :');readln( adr );
        write('type d"operation v(irement),r(ertait)'); readln( op );
        write('montant :'); readln( mt );
        write('date :');readln( date );
        writeln(' end data of 1 record' );
    end;
end;
procedure ecrit1client_f( var c:client ; var f : bank );
begin    write(f , c );        end;
procedure creat( var f : bank );
var c : client ;
begin

```

```

        writeln('creation du fichier banque.dat'); rewrite ( f );
    repeat
        lit1client( c ); ecrit1client_f( c , f );
        writeln('another record y(es) or n(o) : ');
    until readkey in ['n','N'];
    writeln('Finnnnn creation du fichier banque.dat');
    close( f );
end;
procedure ecrit1client( var a : client );
begin
    with a do
        begin
            writeln('file pos:',filepos( b ));
            write( numc:10 , nomp:10,adr:10, op:10 , mt:10:1, date:10 );
        end;
end;
procedure ecritbank( var f : bank );
    var c : client;
begin
    reset( f ); writeln('debut du fichier =',filepos( f ));
    while not eof ( f ) do
        begin
            read(f ,c ); ecrit1client( c );
            writeln('taper sur Entrer pour voir le reste'); readln ;
        end;
        writeln('Fin de FFFFFFFFiiiiiiiCCCCCCCChhhhhier ');
        close( b );
    end;
procedure ajout( var f : bank );
    var c : client;
begin
    reset ( f ); seek( b , filesize(b));
    writeln('ajout de 1 ou + clients');
    repeat
        lit1CLIENT( c ); ECRIT1CLIENT_f(c, f );
        writeln('autre client o(ui) ou N(on): ');
    until readkey in ['n','N'];
    close ( f );
end;

```

```

procedure modify( var f : bank );
  var c : client ; NC : integer;  trouve : boolean ;
  begin
    writeln('Procédure de Modification d"1 ou + clients');
    repeat
      reset( f );  write('Give the Compte Number :');
      readln( NC ); trouve := false ;
      while ( Not eof( f )) and ( not trouve )do
        begin
          read( f , c ) ;
          if c.NumC = NC then
            begin
              trouve := true ;
              writeln('le client est a la position :',filepos( f));
              writeln('le client est :');  ecrit1client( c );
              writeln('give the new data');  lit1client( c );
              seek( f , filepos( f )-1);  ecrit1client_f( c , f );
            end;
          end;
        if not trouve then  writeln(messagerror);
        close ( f );
      writeln(' Modify more y(es) or N(o) ? ');
      until readkey in ['n','N'];
    end;
  procedure delete( var f : bank );
    var c : client ;n : integer; trouve : boolean ;
    begin
      writeln('suppression logique d"1 ou + clients');
      repeat
        reset ( f );  trouve := false ;
        write('give the cumpte number :');  readln( n );
        while (not eof ( f )) and (not trouve) do
          begin  read( f , c );
            if n = c.numC then
              begin  trouve := true ; c.NumC := -C.NumC;
                seek ( f , filepos( f ) -1 );  ecrit1client_f( c , f );
                writeln('record deleted by -1');
              end;
            end;
          end;
        end;
      end;
    end;
  end;

```

```

        end;
        if not trouve then writeln( messagerror);
        close ( f ); writeln('another delete ? y(es) or N(o)');
        until readkey in ['n','N'];
    end;
PROCEDURE DELETEALL ( var f : bank );
    var temp : bank ; c : client ;
begin
    assign(temp , 'temp'); rewrite( temp ); reset( f );
    writeln('are u sure U want to delete all marked record Y(es) or N(o)');
    if readkey in ['y','Y'] Then
        begin
            while not eof ( f ) do
                begin
                    read( f ,c );
                    if c.Numc > 0 then write( temp , c );
                end;
            close( f ); close( temp );
            writeln('copy of temp file back to initial file');
            rewrite( f ); reset( temp );
            while not eof( temp ) do
                begin
                    read( temp , c ); write( f , c );
                end;
            close( f ); close( temp ); writeln('end of copy ');
        end;
    end;
procedure total_Mv( var f : bank );
    var c : client; nc : integer ;
    Tmvv,Tmvr,sld : real; trouve : boolean;
begin
    repeat
        reset ( f ); Tmvv := 0;TmvR := 0;sld := 0;
        write('donner le numéro du client :');
        readln( nc ); trouve := false ;
        while not eof( f ) do
            begin
                read ( f , c );
                if nc = c.numc then

```

```

begin    trouve := true;
        case c.op of
            'v' : Tmvv := Tmvv + c.mt;
            'r' : TmvR := TmvR + c.mt ;
        end;
    end;
end;
end;
close ( f );
sld := Tmvv - TmvR ;
if trouve then
begin
    writeln('Résultat du client :',nc );
    writeln('le total des Mvt de Virement est = ',Tmvv);
    writeln('total des Mvt. Retrait est =',TmvR);
    writeln('Le solde est = ',sld:5:1);
end
else writeln( messagererror );
writeln(' another search yes or no ');
until readkey in ['n','N'];
end;
procedure cherche_client( var F : bank );
var NP : string[20];c : client;  trouve : boolean ; i : integer ;
begin
    repeat
        reset( f ); i := cherche_in_index(id) ;
        if i <> 0 then
            begin
                seek ( f,i-1); read( f , c ); ecrit1client ( c );
            end else writeln('erreur client non trouve');
            close ( f ); writeln('autre recherche o(ui) ou n(on)');
        until readkey in ['n','N'];
    end;
procedure cherche_Nc( var F : bank );
var NP : string[20];c : client;  trouve : boolean ;
begin
    repeat
        reset( f );
        writeln('recherche du numéro de compte d'un client');

```



```

write('donner le Nom prénom :'); readln( NP ); trouve := false ;
while (not eof( f )) and ( not trouve )do
begin
    read( f , c );
    if c.NomP = NP then
    begin
        writeln('le Numero de cpt =',c.NumC );
        trouve := true ;
    end ;
end;
if not trouve then writeln('erreur client non trouve');
close ( f ); writeln('autre recherche o(ui) ou n(on)');
until readkey in ['n','N'];
end;
procedure Total_Jour( var f : bank );
var c : client ; SD ,SC : real ;
begin
    writeln('Total des virements et retraits du jour');
    reset( f ); SD := 0 ; SC := 0 ;
    while not eof ( f ) do
    begin
        read( f , c );
        case c.op of
            'v' : sc := sc + c.MT ;
            'r' : SD := SD + c.MT ;
        end;
    end;
    writeln('Total mouvement des virements du jour =',Sc:8:2);
    writeln('Total mvt. des retraits du jour =',SD:8:2);
    close ( f );
end;
begin {prog.principal}
    writeln('donner le nom du fichier :?');readln( nomFichier );
    assign(b,nomfichier);
    repeat
        clrscr;
        writeln('Programme de gestion des clients d"une Banque');
        writeln('C:creation');

```

```

writeln('E:ecrit fichier et taille du fichier') ;
writeln('A:ajout ');
writeln('T: total des mvts retraits/virements et solde /client');
writeln('N: cherche Numéro de compte des clients');
writeln('J: total_jour pour tous les clients' );
writeln('M : modification des data d"1 ou + clients ');
writeln('S : suppression d" 1 ou + record' );
writeln('D : delete all marked records in S procedure ');
writeln('I : index file on num compte ');
writeln('v : iew index fie ');
writeln('H : cherche client ');
readln( choix );
case choix of
    'C','c' : creat( b );
    'E','e' : ecritbank( b );
    'A','a' : ajout ( b );
    'T','t' : Total_Mv ( b );
    'N','n' : cherche_Nc( b );
    'J','j' : Total_jour ( b );
    'M','m' : Modify ( b );
    'S','s' : delete ( b );
    'D','d' : deleteall( b );
    'i','I' : index_file( id );
    'v','V' : see_index( id );
    'h','H' : cherche_client ( b );
    else writeln('choix non disponible');
end;
writeln('another run ?');
until readkey in ['n','N'];
writeln('ffffffffffffffffiiiiiiiiii au RevoiiiiiiiiiiRrrrrr');
end.

```

المراجع :

Bibliography

Bibliographie:

أولا : ترجمات باسكال .

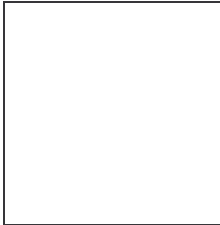
Pascal Compilers and Tools

Compilateurs et outils Pascal .

إليك بعض المراجع لترجمات لغة باسكال المتوفرة الآن مجانا على مواقع قواعد البيانات
Web sites data base يمكنك شحن download منها ما تحتاج , و المهم أن اليوم مع شبكة
Internet ما عليك إلا المشاهدة و البحث لتتعلم و تستفيد . و على سبيل المثال ما

<http://www.freebyte.com/>

يلي :



Freebyte's Guide to free
Pascal programming

Free Pascal compilers

Free Pascal

Freeware 32bit Pascal compiler. Open Source (GPL).
Available for i386+ and 680x0 processors. Supports
Linux, DOS, AmigaOS, OS/2, Win32. The Pascal
language is compatible with TP 7.0 (including classes)
and has some extensions used by **Delphi** (rtti,

compiler itself is written entirely in Pascal.

Virtual Pascal

Free Pascal compiler for 32 bit cross-platform development. Currently OS/2 and Win32 are supported. Compatible with Borland Pascal and Delphi, including the Run-Time Library (RTL), optimizing compiler, integrated debugger, online documentation.

Dev-Pascal

IDE with which one can create Win32 programs, dll's and console-applications using the Free Pascal compiler (included). Features: multi-window editor, setup creator, insight Debugger, customizable syntax highlighting, setup creator, resource file editor, tool manager, project templates, etc.

Alice

A free Pascal compiler by Brad Templeton for x86 computers. Syntax directed editor, integrated, visual programming environment with debugger, 700 help screens , Turbo Pascal compatible, works on as little as an 8088 PC with 384K of RAM, also in DOS box Windows 95/NT, etc.
Compiler source available!

Free Pascal tools

PascalToHTM

This program converts pascal source files to colored (syntax highlighted) html web pages.

Pascal Tools

Useful page specialized in pascal tools for various platforms (Windows, Linux, Mac).

Pascal resources

Pascal Central

"The intent of Pascal Central is to provide the Pascal community one place to obtain Pascal technical information, Pascal source code, and Pascal-related internet links."

ثانيا : الكتب و الوثائق .**Livres et documents**

1. البرمجة بلغة الباسكال لـ: آرثر م كيلر . مترجم عن A first cours in computer programming using Pascal .الدار الدولية للنشر و التوزيع.القاهرة . 1988 .
2. Lire Pascal . D.Piot .CEDIC .1982.
- Introduction au Pascal .P.Le Beux .Sybex .1980
3. initiation Pascal J.C.Guillomet .Ed.Radio .
4. Pascal : Norme ISO et Extensions P.Lignelet.Masson.1983.
5. Logique de Programmation .Trembley Bunt Sorenson.Mc.Graw Hill . 1985
6. Introduction a la Programmation avec Pascal .R.B.Kierbutz.Eyrolles .1983.
7. Pascal par l'exemple . J.A.Hernandez .Eyrolles .1983..
8. Programmation: Structure Language Pascal.R.Lortal.Masson .1983,.
9. Mathématique élémentaire du point de vue algorithmique.Arthur Engel .CEDIC .
10. User Manuel and Report .K.Jensen and N.Wirth.Spring Verlag ,New York .1974
11. Programming for People / Pascal .David G. Kay .Mayfield Pub.co.1985..
12. Ms Pascal Reference and user Manuel under MsDos .

13. Algorithms + Data structures = Programs .Nicklaus Wirth .Printice - Hall .
Englewood Cliffs,N.J,1976

14. مصطلحات الإعلامية - المنظمة العربية للتربية و الثقافة و العلوم - جامعة الدول

العربية ..

15. معجم مصطلحات الحاسبات . إنجليزي. فرنسي. عربي . للمؤلف. 1987 .

16. و أحسن الكتب في المادة سلسلة الأستاذ :نوت د. : فن برمجة الحاسب .

Knuth ,D.E.,*The Art of Computer Programming* (Reading , Mass.:Addison-
Wesley , 1973)

17. Data Structures and Algorithm Analysis in C

By Mark Allen Weiss

TABLE DES MATIERES

فهرس الكتاب

..... الباب الأول : نظرة عامة عن لغة باسكال 3

CHAPITRE 1:VUE GENERALE SUR LE
LANGUAGE..... 3

PASCAL 3

1..... لماذا نريد تعلم هذه اللغة. 3

1.APPRENDRE PASCAL ? 3

2..... ملخص عن اللغة. 5

2.SUMMARY OF THE LANGUAGE..... 5

..... الباب الثاني : قواعد تكوين الكلمات في باسكال. 6

CHAPITRE 2:REGLE DE FORMATION
DES MOTS EN PASCAL..... 6

1. 1. BUT..... الهدف. 6

2. التعاريف ..	6
2.LES IDENTIFICATEURS / IDENTIFIERS ..	6
3. الأعداد ..	9
3.LES NOMBRES ..	9
1.3. الأعداد الصحيحة ..	9
3.1 <i>Les nombres entiers/Integer numbers</i> ..	9
2.3. الأعداد الحقيقية ..	10
3.2 <i>Real numbers/nombres réels</i> ..	10
4. سلاسل الحركات ..	13
4.LES CHAINES DE CARACTERES ..	13
5. تمارين و أمثلة ..	14
5.EXEMPLE ET EXERCICES ..	14
الباب الثالث : قواعد البرمجة في باسكال ..	14

CHAPITRE 3:REGLES DE PROGRAMMATION EN PASCAL..... 14

1.1.BUT.الهدف ..	14
2.. رأس البرنامج و بنيته العامة ..	14
2.EN TETE DU PROGRAMME ET STRUCTURE GENERALE.	14
3.. التصريحات ..	15
3.DECLARATIONS ..	15
1.3. التصريح بالعلامة ..	16
3.1 <i>Déclaration d'étiquette</i> ..	16
2.3. بالثابت التصريح ..	17
3.2 <i>declaration de constante</i> ..	17

3.3. التصريح بالنوع	19
3.3 Declaring data type/ declaration de type	19
4.3. التصريح بالمتغير	22
3.4 Déclaration de variable	22
5.3 تمارين :	24
3.5 EXERCICES	24
04 التعليمات المنفذة	25
04: STATEMENTS / LES INSTRUCTIONS	
EXECUTABLE	25
1.4 هدف الفقرة :	25
4.1 BUT	25
2.4 القاعدة النحوية	25
4.2 Règle syntaxique	25
3.4 التعيين	25
4.3 Assignment statement .l'affectation	25
4.4 التعليمات الحسابية	26
4.4 Les instructions arithmétiques	26
5.4 التعليمات المنطقية	29
4.5 Les instructions logiques	29
6.4 إجراءات القراءة و الكتابة	32
4.6 Procedure read , readln , write , writeln	32
5. الدوال القياسية.	34
5.FONCTIONS STANDARDS	34
6. تمارين :	36
6.EXERCICES	36
الباب الرابع : الإجراءات و الدوال	39

CHAPITRE 4:LES FONCTIONS ET PROCEDURES	39
1.الهدف	39
1.BUT.....	39
2.الإجراءات و الدوال وسيلة بنبوية :	39
2.LES PROCEDURES ET FONCTIONS = METHODE	
STRUCTUREE	39
3.الإجراءات	40
3.LES PROCEDURES	40
1.3 القاعدة النحوية	40
3.1 Règle syntaxique	40
2.3 الإجراءات بدون وسيط	41
3.2 Procédure sans paramètre	41
3.3 الإجراءات بوسيط	45
3.3 Procédure avec paramètre	45
4.الدوال	52
4.FUNCTIONS.LES FONCTIONS	52
1.4 تعريف الدالة :	52
4.1 Définition d'une fonction	52
2.4 القاعدة النحوية	53
4.2 Syntax	53
5.الدوال و الإجراءات الأمامية	55
5.FONCTION ET PROCEDURE FORWARD	55
6.الإجراءات والدوال التراجعية	56
6.FONCTIONS ET PROCEDURES RECURSIVES	56

7. الدوال و الإجراءات الخارجية ..	58
7. PROCEDURES ET FONCTIONS EXTERNES	58
8. تمارين ..	60
8. EXERCICES	60
الباب الخامس : التعليمات البنوية .	61

CHAPITRE 5: LES INSTRUCTIONS STRUCTUREES 61

1. المقدمة :	61
1. INTRODUCTION	61
2. التتابع .	62
2. SEQUENCE D'INSTRUCTION	62
3. الاختيار ضمن حالتين على الأكثر ..	62
3. CHOIX PARMIS 2 CAS AU PLUS : IF .. THEN .. ELSE	62
1.3 قاعدة التعليم الشرطية	62
3.1 Syntaxe	62
2.3 أمثلة :	63
3.2 Exemples	63
4. الاختيار من مجموعة حالات ..	67
4. INSTRUCTION DE CHOIX MULTIPLE: CASE .. OF .. END	67
5. التعليمات التكرارية ..	70
5. LES INSTRUCTIONS REPETITIVES	70
1.5 التعليم : أعد من .. إلى ... الفعل	70
5.1 For .. to do	70

2.5.....تعلیمه ما دام الشرط صحيحا إفعال ...	72
5.2 While (condition: true) do	72
3.5: . تعلیمه أعد ... إلى أن یصبح الشرط صحيح	75
5.3 Repeatuntil (condition: true)	75
6.: تمارین حول الباب	77
6.EXERCICES	77
الباب السادس : تطبيقات على أنواع البيانات البسيطة .	79

CHAPITRE 6:DETAILS SUR LES TYPES DE DONNEES.....79

1. صحيح: النوع	79
1.TYPE ENTIER/INTEGER TYPE	79
2. النوع الحقيقي .	80
2.TYPE RÉEL / REAL TYPE	80
3. نوع الحركات.	81
3.CHAR AND STRING TYPE	81
4. النوع المنطقي .	82
4.BOOLEAN TYPE	82
5. النوع المصرح به إختياريا أو التعدادي ..	83
5.TYPE SCALAIRE DECLARE OU ENUMERE	83
6. نوع المجال ..	85
6.TYPE INTERVALLE /SUBRANGE TYPE	85
الباب السابع : بنية القائمة أو المصفوفة من بعد واحد.	86

CHAPITRE 7:STRUCTURE DE LISTE OU TABLEAU A UNE DIMENSION.....86

1. الهدف :	86
1.BUT	86
2. التعريف :	86
2.DEFINITION	86
3. التصريح بالقائمة :	86
3.DECLARATION DE LISTE / ARRAY	86
4. العمليات الشاملة على القائمة :	87
OPERATIONS GLOBALES SUR LES LISTES	87
1.4 التعيين :	87
4.1 L'affectation	87
2.4 المقارنة :	87
4.2 L'évaluation logique	87
5. المتغير أو العنصر المذيل ..	88
5.VARIABLE INDICE	88
6. التصريح بالوسيط الشكلي من نوع قائمة في إجراء أو دالة.	88
6.PARAMETRE DE TYPE ARRAY DECLARE FORMEL DANS UNE PROCEDURE OU FONCTION.	88
7. قراءة القائمة :	88
7.LECTURE D'UNE TABLE/ARRAY	88
8. كتابة القائمة :	89
8.ECRITURE D'UNE LISTE	89
9. التعيين بين و إلى عناصر القائمة :	89
9.AFFECTATION ENTRE ELEMENT D'UNE LISTE	89
10. النفاذ إلى عنصر من القائمة :	90
10 ACCES A UN ELEMENT DE LA LISTE	90

11. : تمارين.....	91
11.EXÉRCICES	91
الباب الثامن : خوارزميات الترتيب ..	93
CHAPITRE 8:ALGORITHMES DE TRI.....	93
SORT ALGORITHMS.....	93
1.الترتيب بطريقة التبادل.....	93
1.SORTING BY STRAIGHT EXCHANGE OR BUBBLE SORT	93
TRI A BULLE.....	93
2.الترتيب بالانتخاب.....	97
2.TRI PAR SELECTION.....	97
3.الترتيب بالدمج ..	98
3.TRI PAR INSERTION.....	98
4.الترتيب بالعد ..	100
4.TRI PAR ENUMERATION	100
5.الترتيب بالتبادل أو الاختزال ..	101
5.TRI PAR TRANSPOSITION.....	101
6.الترتيب السريع.....	102
6.TRI RAPIDE / QUICK SORT	102
الباب التاسع : المجموعات ..	104
CHAPITRE 9:LES ENSEMBLES / SETS ...	104
1.الهدف	104
1.BUT.....	104

2. تعريف المجموعات	105
2.DEFINITION DES ENSEMBLES.	105
3. تعريف باسكال للمجموعات	105
3.LES ENSEMBLES EN PASCAL	105
4. العمليات الممكنة على المجموعات	107
4.OPERATIONS GLOBALES SUR LES ENSEMBLES ...	107
1.4 .الإتحاد	107
4.1 <i>L'union</i>	107
2.4 .التقاطع	107
4.2 <i>L'intersection</i>	107
3.4 .الفرق	108
4.3 <i>La différence</i>	108
4.4 .التساوي و الإختلاف	108
4.4 <i>Égalité et inégalité</i>	108
5.4 .العنصر عضو من مجموعة	108
4.5 <i>L'élément appartient ou In</i>	108
6.4 .الإحتواء أو المجموعة العليا و المجموعة التحتية أو السفلى	109
4.6 <i>Super set and Subset</i>	109
الباب العاشر: المصفوفة من نون بعد	109

CHAPITRE 10:TABLEAUX A N DIMENSIONS..... 109

1. تعريف المصفوفة	109
1.DEFINITION DE LA MATRICE OU TABLEAU	109
2. التصريح بالمصفوفة	111
2.DECLARATION DE MATRICE.	111

3..المصفوفة الثنائية أو الجدول.....	112
3.MATRICE A 2 DIMENSIONS OU TABLEAU	112
1.3 المصفوفة الوحدة.....	114
3.1 <i>Matrice unité</i>	114
2.3 جمع مصفوفتين.....	115
3.2 <i>Addition de 2 matrices</i>	115
3.3 قلب السطر إلى عمود.....	115
3.3 <i>Transposée d'une matrice</i>	115
4.3 تمارين	116
3.4 <i>Exercises</i>	116
4..المصفوفة من 3 أبعاد.....	116
4.MATRICE TRIDIMENSIONNELLE.....	116
1.4 قراءة مصفوفة من ثلاث أبعاد	117
4.1 <i>Lecture d'une matrice à 3 dimensions</i>	117
2.4 البحث عن عنصر من مصفوفة	117
4.2 <i>Recherche d'un élément d'une matrice</i>	117
5..المصفوفة من 4 أبعاد	118
5.MATRICE A 4 DIMENSIONS.....	118
6..تمارين.....	119
6.EXERCICES	119
الباب الحادي عشر : بنية بيانات معالجة النصوص ..	120

CHAPITRE 11:TRAITEMENT DE TEXTE /DATA STRUCTURE : TEXT PROCESSING.

.....	120
1..الهدف	120
1.BUT.....	120

2.إحداث النوع :سلسلة الحركات في باسكال القياسي..2	120
2.IMPLEMENTING THE STRING PACKAGE IN STANDARD PASCAL	120
3..مسألة.....	122
3. PROJET.....	122
الباب الثاني عشر : التسجيل .	123
CHAPITRE 12 : L'ENREGISTREMENT / RECORD	123
1.. الهدف	123
1.BUT.....	123
2..تعريف التسجيل.....	123
2.DEFINITION DE L'ENREGISTREMENT.....	123
3..التصريح بالتسجيل	124
3.DECLARATION DE L'ENREGISTREMENT.....	124
4..العمليات الممكنة الشاملة على التسجيل.....	125
4.OPERATIONS GLOBALES SUR LES ENREGISTREMENTS	125
5..النفاز إلى الحقول.....	126
5.ACCES AU CHAMPS	126
6..التعليمة WITH	128
6.L'INSTRUCTION WITH.....	128
7..تسجيل داخل تسجيل	129
7.ENREGISTREMENTS IMBRIQUES	129
8..التسجيل المتغير.....	131

8.ENREGISTREMENT VARIANT / VARIANT RECORD	131
9..قائمة التسجيلات.....	134
9.TABLEAU D'ENREGISTREMENTS / ARRAY OF RECORD	134
10.. قائمة كحقل بتسجيل	135
10.TABLEAU CHAMPS DANS UN ENREGISTREMENT	135
11..المجموعات داخل التسجيل.....	137
11.ENSEMBLES DANS UN ENREGISTREMENT / SETS IN RECORD.....	137
12.. الوسيط من نوع التسجيل في الإجراء و الدالة.....	137
12.L'ENREGISTREMENT COMME PARAMETRE DANS UNE FONCTION ET PROCEDURE	137
13..تمارين.....	138
13.EXERCICES	138
الباب الثالث عشر : الجذاذات	140
CHAPITRE 13:LES FICHIERS / FILES	140
1. الهدف	140
1.BUT.....	140
2..تعريف الجذاذة	141
2.DEFINITION DU FICHIER / FILES.....	141
3..التصريح بالجذاذة	141
3.DECLARATION DE FICHIER	141
4..أنواع الجذاذات	143

4. TYPE DE FICHIER.....	143
5. الجذاذة المنطقية و الفيزيائية ..	143
5. FICHIER LOGIQUE ET PHYSIQUE.....	143
6. تنظيم الجذاذات ..	144
6. ORGANISATION DES FICHIERS.	144
1.6 التنظيم التتابعي ..	145
6.1 <i>L'organisation Séquentiel</i> .	145
2.6 التنظيم المباشر ..	146
6.2 <i>L'organisation Directe</i>	146
3.6 التنظيم التتابعي المفهرس ..	146
6.3 <i>Le séquentiel indexé</i>	146
7. الإجراءات و الدوال الموفرة لمعالجة الجذاذات .	146
7. PROCEDURES ET FONCTIONS DISPONIBLES POUR LES FICHIERS.	146
1.7 الربط بين الجذاذة المنطقية و الفيزيائية ..	146
7.1 <i>Liaison fichier logique - physique :</i>	146
Assign(..);	146
2.7 إجراء فتح الجذاذة للكتابة فيها ..	147
7.2 <i>Procédure d'ouverture pour écrire : Rewrite</i>	147
3.7 إجراء فتح الجذاذة للقراءة ..	148
7.3 <i>Procédure Reset(var f);</i>	148
4.7 الدالة : لمراقبة نهاية الجذاذة ..	149
7.4 <i>Function Eof (var f) : boolean;</i>	149
5.7 الدالة : نهاية السطر في الجذاذة ..	149
7.5 <i>Function eoln(var f):boolean ;</i>	149
6.7 : <i>get</i> .الإجراء ..	149
7.6 <i>Procédure get (var F);</i>	149

7.7	<i>put</i> إجراء الكتابة في الجذابة :	150
7.7	<i>Procedure Put (var F);</i>	150
8.7	<i>read , readln</i> الإجراءات الخاصة بالقراءة :	150
7.8	<i>Procedures Read et readln</i>	150
9.7	<i>Write , writeln</i> الإجراءات الخاصة بالكتابة :	151
7.9	<i>Les procedures Write , writeln</i>	151
10.7	إجراء غلق الجذابة.	152
7.10	<i>Procedure Close (var F);</i>	152
11.7	إجراء النفاذ المباشر إلى أي تسجيل .	152
7.11	<i>Procedure Seek (var F ; N : integer);</i>	152
12.7	إجراء حذف الجذابة فيزيائياً .	153
7.12	<i>Procedure de suppression de fichier:Erase</i>	153
13.7	إجراء الإضافة في الجذابة النصية .	153
7.13	<i>Procédure d'ajout dans un fichier text :Append</i>	153
14.7	إلك أين هي نافذة الجذابة . دالة ترجع	154
7.14	<i>Fonction :Position dans le fichier:filePos</i>	154
15.7	دالة ترجع عدد التسجيلات في الجذابة .	155
7.15	<i>Fonction : Taille du fichier:fileSize.</i>	155
16.7	دالة ترجع ما قيمة نهاية الجذابة .	155
7.16	<i>Fonction : donne la valeur de fin de fichier:seekEof.</i>	155
17.7	دالة ترجع قيمة نهاية السطر .	156
7.17	<i>Fonction : donne la valeur de fin de ligne:SeekEoln.</i>	156
8.	الجذابات النص ..	156
8.	LES FICHIERS TEXTES./ TEXT FILES.....	156

1.8	التعريف	156
8.1	Definition.	156
2.8	إحداث الجذاذة	157
8.2	Procédure de Création de fichier .	157
	الخوارزمية :	157
3.8	نقل جذاذة إلى أخرى.	157
8.3	Copie de Fichiers.	157
4.8	طبع الجذاذة	157
8.4	Imprimer un Fichier .	157
5.8	إضافة تسجيل في النظام التتابعي	158
8.5	Ajout en mode séquentiel .	158
5.8	إستحداث الجذاذة في النظام التتابعي	158
8.5	Mise a jour en mode séquentiel .	158
6.8	الإستحداث في النظام المباشر	159
8.6	Mise à jour en mode directe.	159
9.	الجذاذات الثنائية التتابعية	165
9.	LES FICHIERS BINAIRES SEQUENTIELS.	165
1.9	إحداث الجذاذة	166
9.1	Création du fichier.	166
2.9	الإضافة في النظام التتابعي	168
9.2	L'ajout en mode séquentiel.	168
3.9	التغيير في النفاذ التتابعي	169
9.3	Modification en mode séquentiel.	169
4.9	الحذف في النظام التتابعي	170
9.4	Suppression en Mode Séquentiel.	170
5.9	عرض محتوى الجذاذة	170
9.5	Consultation du fichier .	170
6.9	الخلاصة :	171

9.6 Conclusion	171
7.9 التصريح بالجدادة المتحولة	172
9.7 Déclaration du fichier mouvement.	172
8.9 إحداث الجدادة المتحولة	173
9.8 Création du fichier mouvement.	173
9.9 إحداث الجدادة الدائمة	175
9.9 Création du fichier Permanent	175
10.9 خوارزمية الإستحداث التتابعي	176
9.10 Algorithme de mise à jour.	176
11.9 خلاصة النفاذ التتابعي	186
9.11 Résumé sur l'accès séquentielle.	186
10 التنظيم المباشر للجدادات	186
10 ACCES DIRECT AUX FICHIERS/ DIRECT FILE	
ACCESS	186
1.10 الهدف	186
10.1 But.	186
2.10 التصريح بالنفاذ المباشر	187
10.2 Déclarer le mode direct.....	187
3.10 الإجراء الخاص بالنفاذ المباشر	187
10.3 Procedure Seek(var f;n:integer);	187
4.10 الإضافة في النظام المباشر	187
10.4 L'ajout en mode direct.	187
10.5 التغيير في النفاذ المباشر	188
10.5 Modification en mode directe.	188
6.10 الحذف في النفاذ المباشر	189
10.6 La suppression en mode Directe.	189
11. التنظيم التتابعي المفهرس	195

11.L'ORGANISATION SEQUENTIEL INDEXEE.....	195
1.11 الهدف.....	195
11.1.But.....	195
2.11 طرق الفهرسة.....	195
11.2 Méthodes d'indexages.....	195
3.11 إجراءات الإستحداث في النظام التتابعي المفهرس.....	198
11.3 Mise a jour en séquentiel indexé.....	198
الباب الرابع عشر :بنية البيانات الحركية.....	200

CHAPITRE 14 :STRUCTURE DE DONNEE DYNAMIQUE.DYNAMIC DATA STRUCTURE..... 200

1.التعريف.....	200
1.DEFINITION.....	200
2. الهدف من إستعمال البنية الحركية:.....	201
3. التصريح بالنوع الحركي : الأسهم ..	202
3.DECLARATION DU TYPE DYNAMIQUE:LES POINTEURS/POINTERS.....	202
4. كيف نعين قيمة إلى المتغير السهم ..	203
4.HOW TO ASSIGN A VALUE TO A POINTER VALUE.	203
1.4 الإجراءات الخاص بإحداث المتغير الحركي:.....	203
4.1 Procedure New.....	203
2.4 Getmem(var P: pointer ; size : word);.....	204
3.4 @العامل ..	206
4.3The @ (at) operator.....	206

5. كيف نتخلى عن المتغير الحركي.....	207
5.DESAFFECTATION D'UNE VARIABLE DYNAMIQUE	
: LA PROCEDURE DISPOSE.	207
6. أمثلة	207
7. القائمة الخطية	210
7.THE DATA TYPE LINEAR LIST.....	210
1.7 القائمة الخطية الأحادية	211
7.1 <i>Liste liniaire simple / singly linked lists</i>	211
2.7 بنية البيانات : الكومة	225
<i>Data type Stack(pile)</i>	225
3.7 بنية الطوابير	231
<i>The data Type Queue.(file)</i>	231
4.7 بنية البيانات الغير خطية :بنية الأشجار	237
<i>Non-Linear complex Data Structure: Trees</i>	237
الباب الخاتمة :نصوص تمارين (محلولة) وبرامج شتى.....	243
CHAPITRE CONCLUSION:EXERCICES	
(CORRIGES) ET PROGRAMMES DIVERS.	
.....	243
كيف تبحث عن خوارزمية مسألة:.....	243
***نصوص التمارين:.....	245
*** : إمتحان نص	252
بعض الدوال والإجراءات التراجعية***.....	267
PROCÉDURES ET FONCTIONS RECURSIVES.....	267
المراجع :.....	299
BIBLIOGRAPHY	299
BIBLIOGRAPHIE:	299

.....باسكال . ترجمانات : أولا	299
PASCAL COMPILERS AND TOOLS	299
COMPILATEURS ET OUTILS PASCAL	299
.....ثانيا : الكتب و الوثائق .	301
LIVRES ET DOCUMENTS	301

في تلمسان (الجزائر) 19 جوان 2000

هذا ما وفقنا لكتابته و نرجوا المزيد إن شاء الله , فمن رغب مشاركتنا و تقديم النصيحة و تشجيعنا لنشر كتب أخرى بالعربية في المعلوماتية عموما و لغة Delphi خاصة فها هو عنواننا على البريد الإلكتروني : bouklikhayahia@yahoo.fr . كما إننا أنجزنا كتيباً صغيراً في حكم سيدي بومدين و ابن عطاء الله السكندري نوفره مجاناً لمن أراد نسخة على قرص مرن ما عليه إلا مراسلتنا على العنوان الإلكتروني و صلى الله على سيدنا محمد و على آل بيته الصادقين و صحبه الكرام و من تبعهم بإحسان إلى يوم الدين .

قال ولي الله أبو مدين شعيب : من اكتفى بالعلم دون الإتصاف بحقيقته تزندق و انقطع و من اكتفى بالتعبد دون فقه خرج و ابتدع و من اكتفى بالفقه دون ورع اغترّ و انخدع و من قام بما يجب عليه من الأحكام تخلص و ارتفع .