محاضرات في مادة الإعلام الالي 02

تحتوي على دروس في الخوارزميات والبرمجة

الأستاذ: عطاالله عمر

مفاهيم عامة حول برمجة الحاسوب :

البرنامج:هو مجموعة من الخطوات «الأوامر» المنطقية يكتبها المبرمج و ذلك من أجل تأدية معين و تتقسم البرامج إلى نوعين:

- **برامج تطبيقية.**
- **برامج النظم.**

1. برامج تطبيقية:

و منها برامج الكتابة و برامج الجداول و الحسابات و برامج الإنترنت و برامج الإحصاء و التصميم...إلخ.

2. برامج النظم:

تكتب من قبل مبرمجي النظم المتخصص لمساعدة مستخدمي الحاسب للاستفادة منه إلى أحسن وجه و من أهمها: لغات البرمجة، أنظمة التشغيل.

لغات البرمجة: هي وسيلة إتصال بين أجهزة الحاسب و الشخص المشغل و تنقسم الى :

أ- البرنامج المصدر:

هو عبارة عن مجموعة من البرامج والتعليمات المكتوبة بشكل منطقي و متسلسل بإحدى لغات البرمجة المتوفرة في الجهاز من قبل مبرمج أو مجموعة من المبرمجين و كل تعليمة من هذا البرنامج هي توجيه الحاسب لأداء عملية معينة من المسألة.

ب- برنامج المترجمات:

للحاسب الآلي لغة واحدة يفهمها هي لغة الآلة و لذا يجب ترجمة البرنامج المكتوب من قبل المبرمج إلى برنامج لغة الآلة و يتم ذلك باستخدام إحدى المترجمات التالية:

✓ **المترجم**: كل لغة من اللغات الراقية لها مترجم خاص بها ومهمته:

1- كشف و تسجيل الأخطاء التي تحدث في البرنامج.

2- ترجمة هذا البرنامج إلى لغة الآلة.

✓ **المفسر**: يحص و يترجم كل أم من أوامر البرنامج الأساسي أولا بأول و يبلغ المبرمج بالخطأ في نفس الوقت.

ج- برنامج الهدف: هو برنامج قابل للتنفيذ على الحاسب و لا يمكن التعديل في لأنه مكتوب بلغة الآلة.

3- لغات الحاسب الآلي: و هي:

أ. **لغة المستوى الادنى**:

- لغة الأدنى هي أول لغة وضعت للحاسب و هي عبارة عن مجموعة مصفوف من الأرقام الثنائية (0، 1).
- لغة التجميع: تستخدم مجموعة من المصطلحات العلمية add (جمع) و sub (طرح) و store (تخزين) و تحتاج إلى مترجم يترجمها إلى لغة الآلة.

ب. لغات المستوى العالي:

و هي لغات انتشرت مع انتشار الحاسب و تستخدم في شتى المجالات و هي سهلة الاستعمال و حلت المشاكل التي واجهت المبرمجين بلغة التجميع و من أهمها: لغة البيسيك، لغة فوترون، الباسكال، السي.

ج– اللغات الراقية: وهي أبسط و أيسر اللغات ومن أمثالها (Delphi, visual, Basic...الخ).

د– أنظمة التشغيل: وظيفتها التنظيم و التحكم في تشغيل إدارة الحاسب الآلي ومن أهمها (Unix, Dos,Windows).

الخوارزميات و طرق البرمجة

الخوارزمية: هي سلسة منتهية من التعليمات تبين لنا الطريقة التي بها تم تنفيذ مجموعة من التعليمات للحصول على حل لمشكل ما.

و إتقان الخوارزميات هو اتفان الصيغة أو البناء المنطقي لبرنامج ما و المشكل الوحيد المطروح هو اختلاف الأوامر بين الخوارزميات و لغات البرمجة بالنسبة لأوامر الخوارزميات باللغة الفرنسية و لكن طريقة عرضها باللغة الانجليزية.

ALGORITHME « Non algorithme » } Partie de l'en-tête

Les unites
Les étiquètes
Les constants
les types
les variables
lesprocedures
les functions

} Partie de Declaration

DEBUT

Instruction01

} Patrie de traitement

Instruction02

FIN

الخوارزمية تتمثل في ثلاثة مراحل:

1. **قسم الرأس:**

للخوارزمية رأس يعرفها ويعبر عن الكلمة المفتاحية للخوارزمية و مطابق للخوارزمية (اسم الخوارزمية).

2. **قسم التصريحات** :أنواع التصريحات في الخوارزمية مرتبة كالتالي:

1- التصريح بالوحدات les unites .....الكلمة المفتاحية (Uses).

2- التصريح بالعناوين les etiquettes.....الكلمة المفتاحية: (label).

3- التصريح بالثوابت les constants .....الكلمة المفتاحية: (const)

4- التصريح بالأنواع les types.....الكلمة المفتاحية: (type).

5- التصريح بالمتغيرات les variables..... الكلمة المفتاحية: (variable).

6- التصريح بالإجراءات les procédures..... الكلمة المفتاحية: (procedure).

7- التصريح بالدوال les fonctions .....الكلمة المفتاحية: (fonction).

1.2. الوحدات:les unitées

مثال: Les unitées : unité 01, unité 02, Dos

2.2. العناوين:les etiquettes

مثال: Etiquette : label a, b

3.2. الثوابت: les constants

التصريح بالثابت يكون كما يلي:

Constante Non constant = valeur.

مثال1: Constante p = 3.14

مثال2: Constante Val = "Omar "

4.2. الأعداد المعرفة بواسطة المستخدم: les types

مثال: type couleur = (noir, rouge, bleu, jaune, vert)

5.2. المتغيرات: les variables

نميز العديد من أنواع المتغيرات:

1- صحيح entier (1,2,3,…)

2- حقيقي reél (0.1, 2, 2.2…)

3- حرفي caractère (a, b)

4- منطقي booléenne (صح، خطأ)

2. التصريح بالمتغيرات:

التصريح بالمتغيرات يكون كما يلي :

Variable list de variable: type-variable

مثال:

Variables A, B, C: entier

التصريح بالحروف يكون كما يلي :

Variables Nom-caractère: caractère

مثال:

Variables A, B: caractère

التصريح بسلاسل الحروف يكون كما يلي:

Variables Nom-chaine de caractère: chaine de caractère [taille]

Variables Mot : chaine de caractère [10]

قسم المعالجة:

يبدأ بالكلمة المفتاحية DEBUT و ينتهي بالكلمة المفتاحية FIN.

أنواع التعليمات:

1. تعليمة الادخال و الاخراج : (les entrées et les sorties)

*** التعليمة « Saisir »**

لترجمة القيم داخل الحاسب الآلي الموجودة ي لوحة المفاتيح يمكن لمستخدم البرنامج استعمال التعليمة saisir لادخاال القيمة و التي تكون على الشكل التالي:

Saisir (liste de variables)

مثال1:

Saisir (A, B, C)

مثال2:

Saisir (X)

*** التعليمة « Affichage »**

لعرض النتائج و التعليق على الخوارزمية يجب الاستعانة بتعليمة Afficher العرض و التي تكون على النحو التالي: (Afficher (message

Message » »: ربما تكون تعليق أو ثابت أو متغير.

1. **التعليق**: يكون سلسلة حروف محصورة بين شاولتين.

مثال:

Afficher (“ Hello world”)

الشاشة تعرض Hello world

2. **المتغير**: الشاشة تعرض محتوى المتغير: Afficher (A)

مثال:

A= 120 الشاشة تعرض 120.

3. **الثابت**:

مثال:

Afficher (5) الشاشة تعرض 5

مثال:ليكن لدينا:

Afficher (” la valeur de la variable A est “)

Afficher (A)

لها نفس أثر التعليمة

Afficher (” la valeur de la variable A est “,A)

2- تعليمة التخصيص (الإسناد) l'affectation «: »

الإسناد هو عملية تسمح بتبديل قيمة المتغير و الشكل الخاص بالإسناد يكون كما يلي:

Nom variable ⟵ valeur

مثال1: A⟵ 5

مثال2: A⟵5+2

مثال3: A ⟵B

مثال: ما هي قيم المتغيرات بعد تنفيذ التعليمات التالية:

A ⟵ 1

B⟵ 2

C ⟵ 3

A ⟵ C+1

B ⟵C

تنفيذ التعليمات:

Instruction	**A**	**B**	**C**
Début			
A ⟵ 1	1		
B ⟵ 2	1	2	
C ⟵ 3	1	2	3
A ⟵C+1	4	2	3
B ⟵ C	4	3	3

تعريف المخطط الانسيابي:

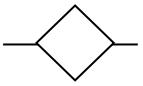
هو تمثيل على شكل رسم تخطيطي يفصل فيه جميع مراحل المعالجة و ذلك باستعمال رموز متعارف عليها.

و هو يعبر عن البداية DEBUT و النهاية FIN.

و هو يعبر عن القراءة (saisir) و العرض (Affichage).

وهو يعبر عن المعالجة Traitement.

 وهو يعبر عن الإسناد L'affectation.

 و هو يعبر عن الشرط Condition.

مثال: أكتب الخوارزمية التي تقوم بجمع عددين A و B .

التحليل:

A et b :réel

S:reel

العمليات:

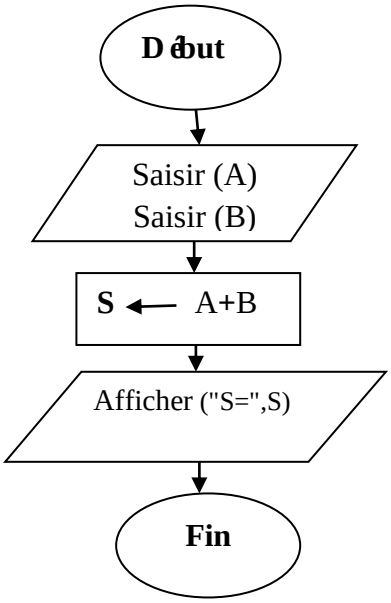
-Donner la valeur de A.

- Donner la valeur de B.

- Calculer S=A+B.

- Afficher S.

* **انجاز المخطط الانسيابي:**



* **انجاز الخوارزمية:**

ALGORITHME somme deux var *iables*
var *iables* A, B, S : *r閞l*
DEBUT
| *Saisir* (A)
| *Saisir* (b)
| $S \leftarrow A + B$
| *Afficher* $("S =", S)$
FIN

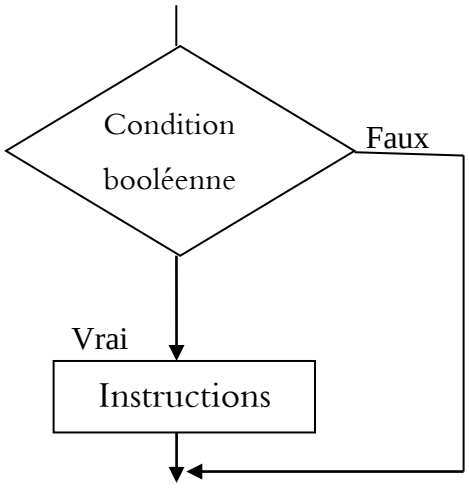
* البرنامج المكافئ للخوارزمية بلغة C++

```
#Include <iostream >
Using name space std;
Int main( )
{
Float  A , B, S ;
Cout<<"entrez les valeurs A et B";
Cin>> A;
Cin>> B;
S =A+B;
Cout<< "S="<<S;
}
```

المعالجات الشرطية: les traitements conditionnelles

1- التعليمة (SI…Alors) l'instruction

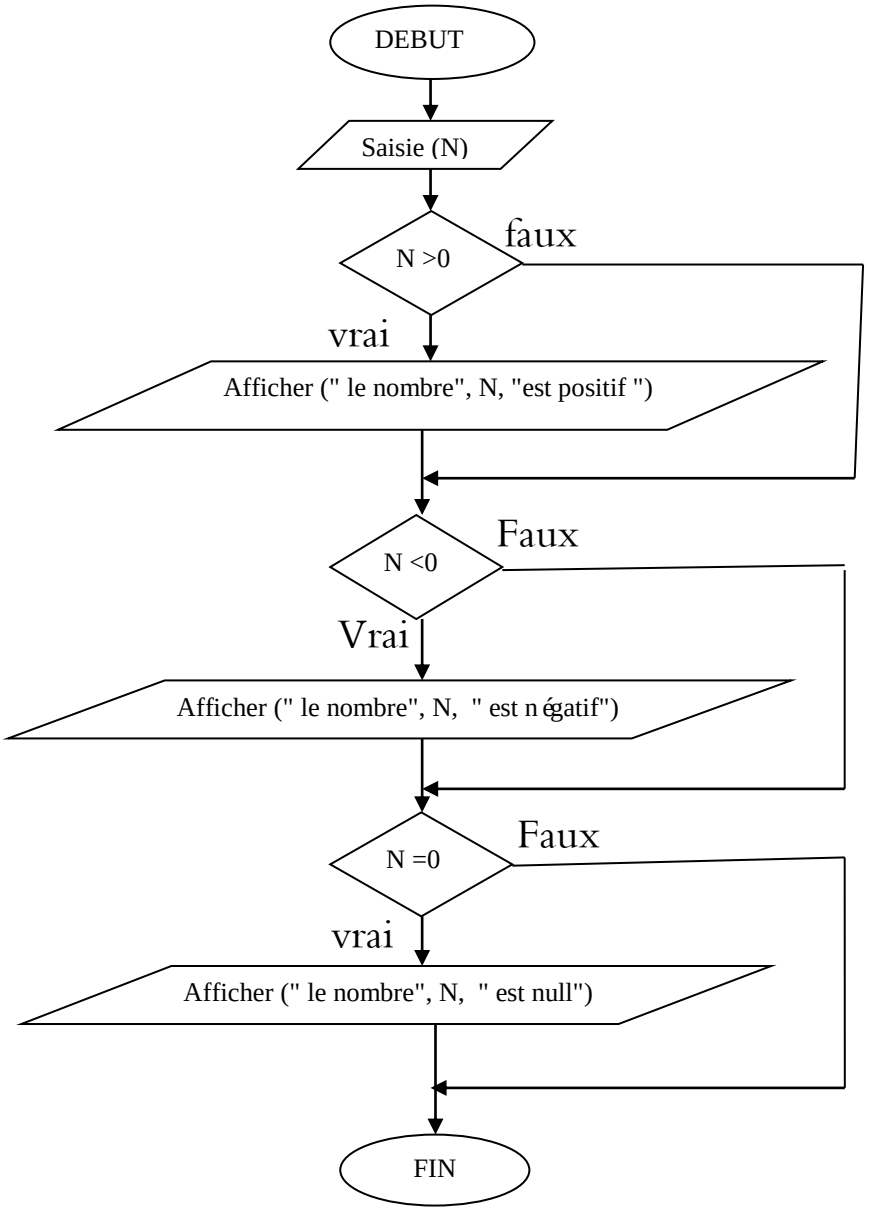
هي معالجة تسمح بتنفيذ تعليمة أو مجموعة من التعليمات و لها شرط منطقي يأخذ القيم «Vrai » أو « Faux » و تستخدم معاملات المقارنة و المعاملات المنطقية Non, ≥, >, ≤ , <, ou, et,= , ≠ وشكلها يكون كما يلي:



Si (*condition bool閑nne*) *Alors*
| *Instructio ns*
Fin Si

تمرين1: أكتب الخوارزمية التي تقرأ العدد ثم تخبر هل هو موجب أو سالب أو معدوم.

* **إنجاز المخطط الانسيابي**



* انجاز الخوارزمية:

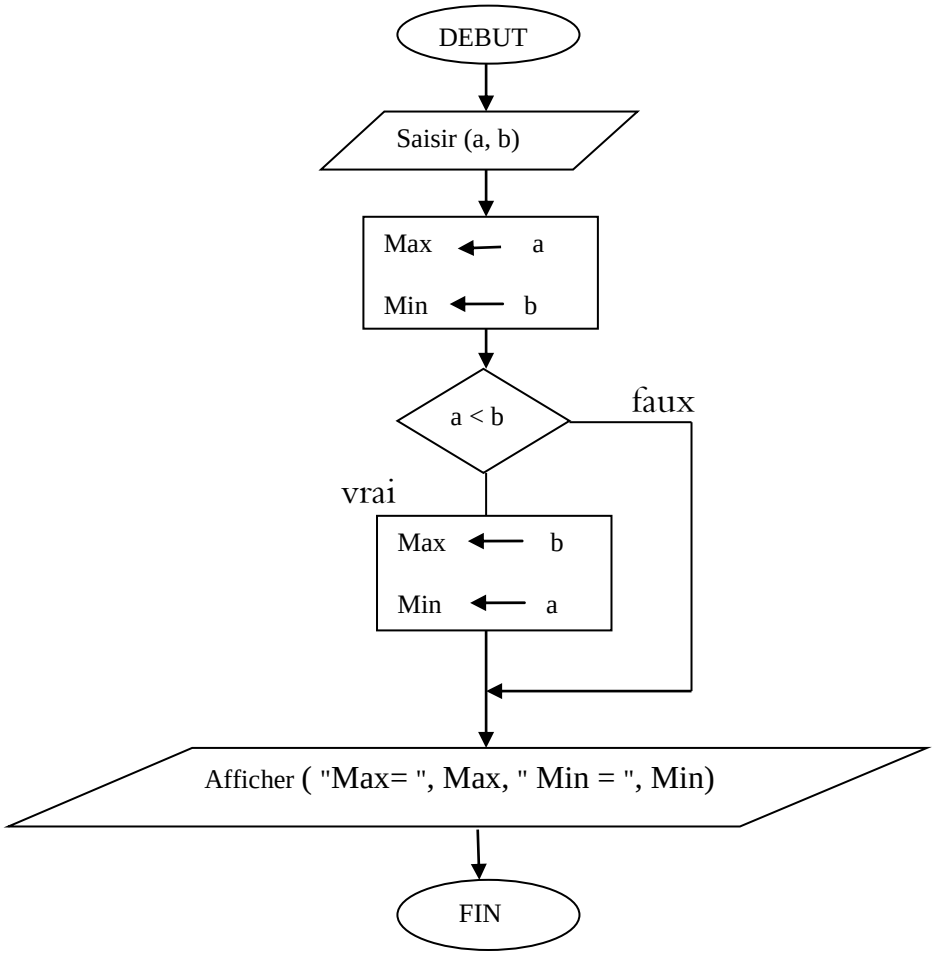
```
ALGORITHME   nombre
 var iables  N : r閑l
DEBUT
   Saisir (N)
   Si  (N > 0) alors
   | Afficher ("le nombre est", N, "est positif ")
   fin si
   Si  ( N < 0) alors
   | Afficher ("le nombre  ", N, "est negatif ")
   fin si
   Si (A = 0) alors
   | Afficher ("le nombre", N, "est null")
   fin si
FIN
```

* البرنامج المكافئ بـــ++C:

```
#Include <iostrean>
Using name space std;
Int main( )
{
Float N ;
Cin>> N;
IF (N>0)
{
Cout <<"le nombre" <<N<<"est positif" ;
}
IF (N<0)
{
Cout <<"le nombre" <<N<<"est n égatif" ;
}
IF (N==0)
{
Cout <<"le nombre" <<N<<"est null" ;
} ;}
```

تمرين2: أكتب الخوارزمية التي تخرج أكبر و أصغر قيمة بين a و b .

*** إنجاز المخطط الانسيابي**



*** انجاز الخوارزمية:**

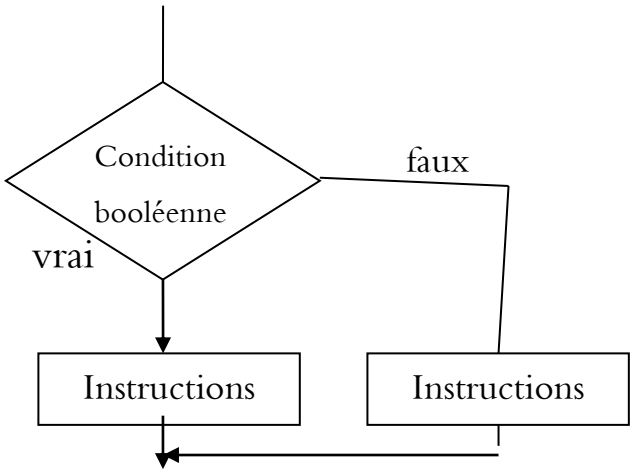
```
ALGORITHME  nombre Max et Min
 var iables a,b,Max,Min : entier
DEBUT
 |
 |Saisir (a,b)
 | Max ← a
 | Min ← b
 | Si (a < b) alors
 | | Max ← b
 | | Min ← a
 | fin si
 | Afficher("Max =",Max,"Min =",Min)
FIN
```

*** انجاز البرنامج المكافئ بـ C++:**

```
#include <iostream >
Using name space std;
Int main ( )
{
Int a, b;
Cin>> a>>b;
Max=a ;
Min=b ;
IF (a < b)
{
Max=b ;
Min=a ;
} ;
Cout <<"Max=" <<Max<<"Min=" <<Min;
}
```

2- التعليمة **l'instruction SI...Alors...Sinon**:

الشكل العام لهذه التعليمة يكون كما يلي:



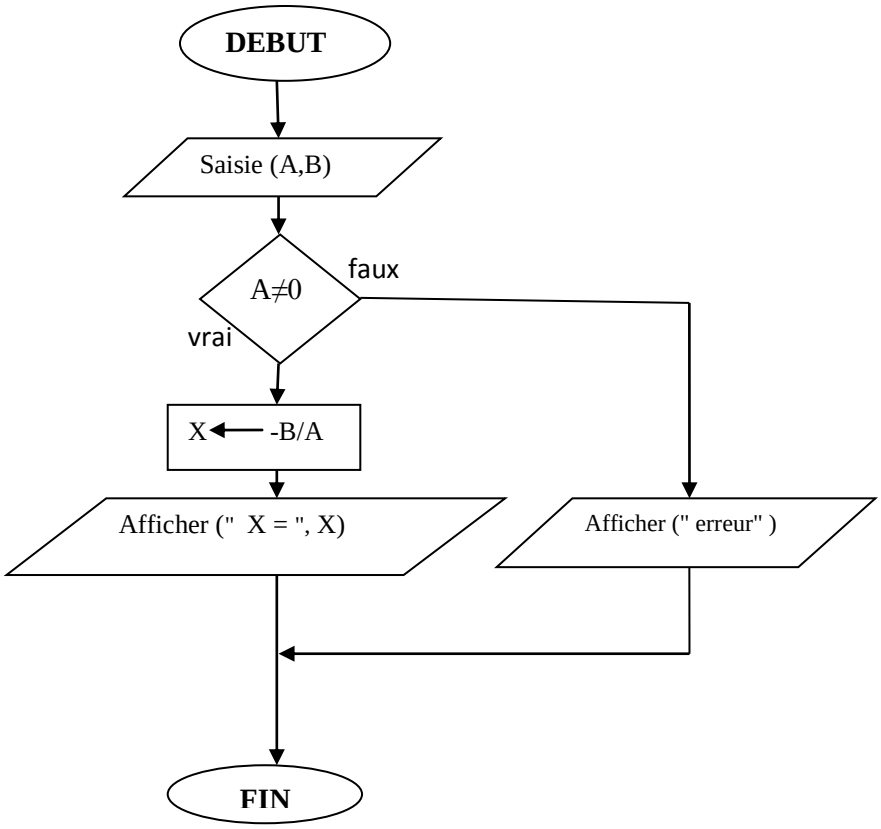
Si (*condition bool閑nne*) *Alors*
| *Instructio ns*
Sinon
| *Instructio ns*
fin si

ملاحظة: إذا كان صحيح تنفذ التعليمات الخاصة بالكتابة الأولى و إذا كان خطأ تنفذ التعليمات الخاصة بالكتابة الثانية.

مثال:

Si $(N \neq 0)$ *Alors*
| $N \leftarrow N-1$
| $I \leftarrow I+1$
Sinon
| $N \leftarrow N+1$
| $I \leftarrow I-1$
fin si

تمرين: أكتب الخوارزمية التي تقوم بحل المعادلة الخطية $AX + B = 0$



***انجاز الخوارزمية**

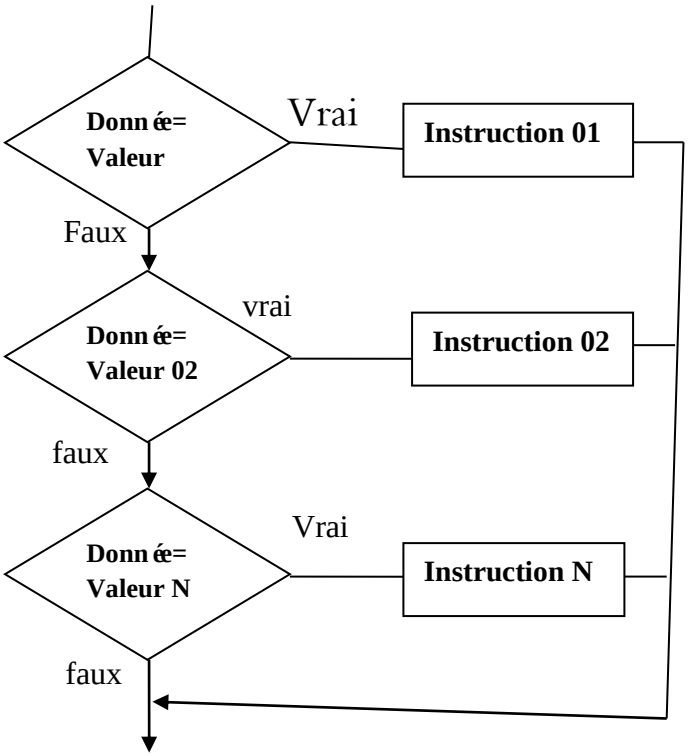
ALGORITHME Equation lin閞re
var *iables* A, B, X : *r閞l*
DEBUT
  Saisir (A, B)
  Si $(A \neq 0)$ *alors*
    $X \leftarrow -A/B$
    Afficher $("X =", X)$
  Sinon
    Afficher $("erreur")$
  fin si
FIN

*** انجاز البرنامج المكافئ بـ C++ :**

```
#include<iostream>

Using name space std;

main ()

{

Float A, B, X;

Cin>>A>>B;

IF (A ! = 0)

{

X= -B/A;

Cout <<" X="<< X;

}

ELSE

{

Cout <<"erreur";

};}
```

3– التعليمة: l'instruction Suivant cas

حينما ينبغي تنفيذ مجموعة من التعليمات بشرط منطقي بواسطة القيمة التي يأخذها المتغير ينتج عن هذا عمل تكرار للتعليمة عدة مرات، و لهذا من الأفضل استعمال الشكل التالي:



Suivant (*donn*閧) *faire*

| *cas valeur* 01 : *instructio n* 01
| *cas valeur* 02 : *instructio n* 02
| *cas valeur* 03 : *instructio n* 03
| ⋮
| ⋮

Sinon

| *autres cas* : *instructio n N*

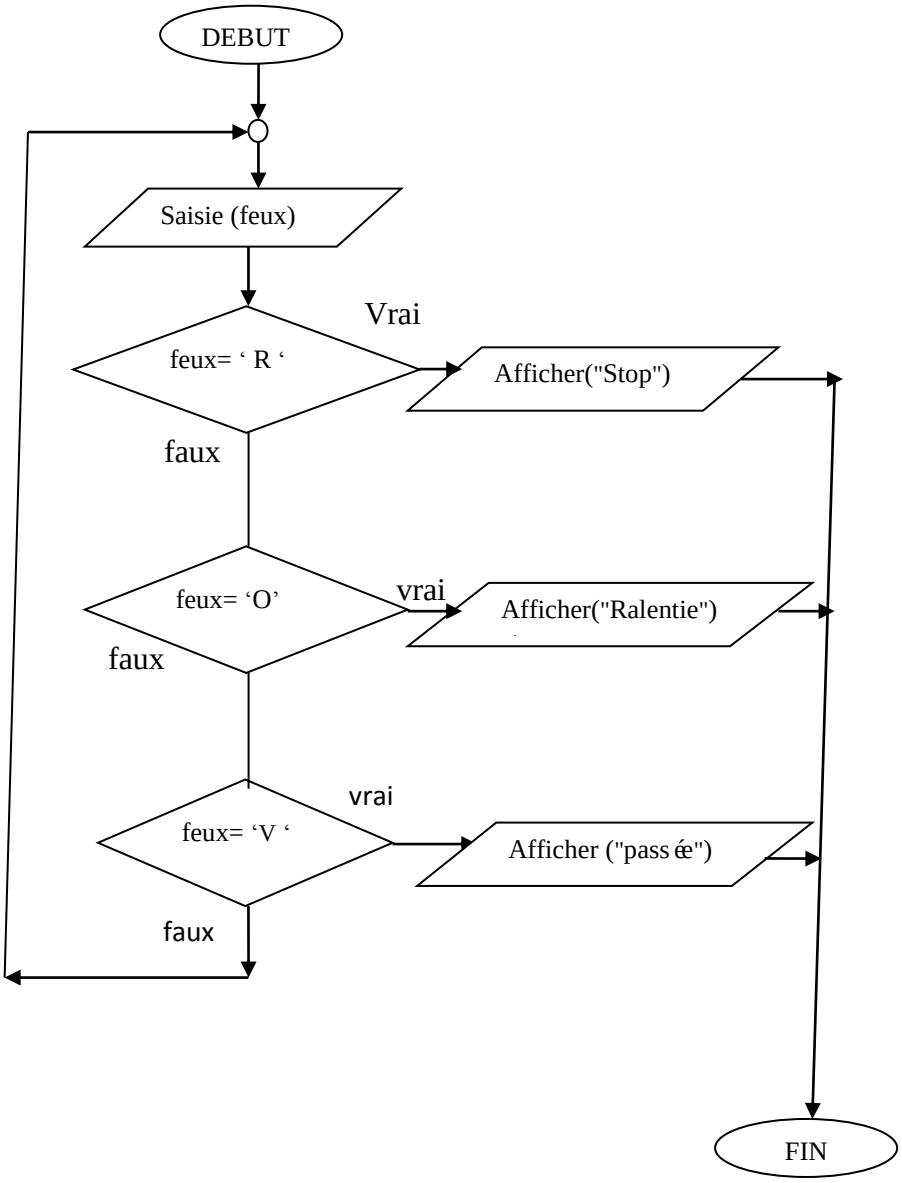
fin si

مثال:

```
DEBUT
| Afficher("donnez votre age")
| saisir(age)
| Suivant (age) faire
|   | cas 3 : Afficher("b閔e")
|   | cas 12 : Afficher("enfant")
|   | cas 18 : Afficher("adolescent")
| Sinon
|   | autres cas : afficher("adulte")
| fin Suivant
FIN
```

تمرين: أكتب الخوارزمية التي تقوم بتنظيم حركة المرور "circulation".

* **انجاز المخطط الانسيابي:**



*** انجاز الخوارزمية:**

```
ALGORITHME   Cerculatio n
 Etiqutte 10
var iables   feux :caractere
DEBUT
  10 : Saisir ( feux )
   Suivant cas ( feux ) faire
     "R": Afficher ("Stop")
     "O": Afficher ("ralentie")
     "V": Afficher (" pass閑')
   Sinon
     Allez 10
   fin Suivant cas
FIN
```

*** انجار البرنامج المكافئ بـــ C++:**

```
#include<iostream>

Using name space std;

Int main ()

{

Char feux;

10: Cin>>feux;

Swich (feux)

{

Case ‘R’: cout<<" stop";

Break;

Case ‘O’: cout<<"ralentie";

Break;
```

```
Case ‘V’:cout<<" passé";

Break;

Default; Goto 10 ;

Break ;

} ; }
```

الحلقات: les boucles

الحلقة تسمح بتنفيذ نفس التعليمات عدة مرات، و هذه المجموعة من التعليمات تسمى جسم الحلقة كل تنفيذ لجسم الحلقة يسمى تكرار و هناك ثلاث أنواع من الحلقات:

* **Tant que**.

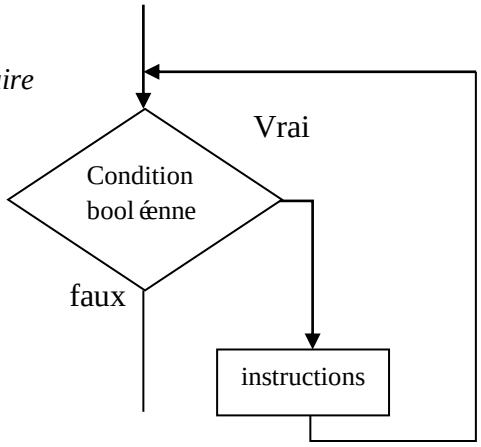
* **Répeter**.

***Pour**.

1- الحلقة Tant que:

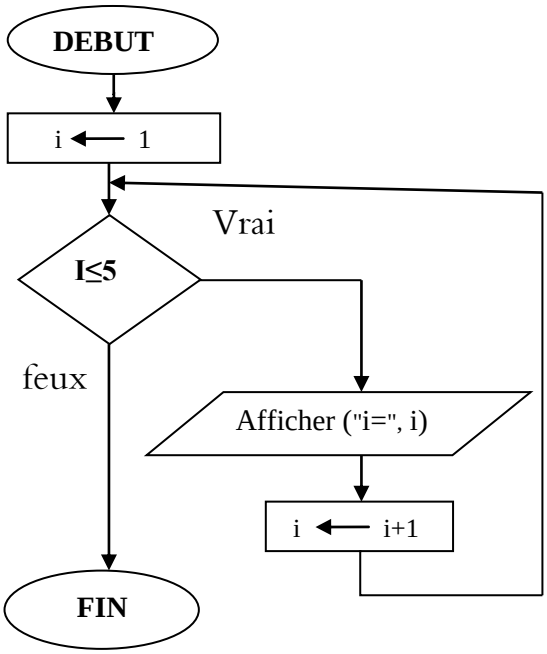
هذه الحلقة تعرف للآلة شرط الحلقة، إذا كان الشرط محقق إذن الآلة تنفذ تعليمات الحلقة للمرة الأولى ثم الحاسب الآلي يعيد اختيار الشرط مرة أخرى إذا كان الشرط محقق إذن الآلة تنفذ تعليمات الحلقة للمرة الثانية، و هكذا تنفيذ الحلقة في كل مرة عندما يكون فيها الشرط محقق، أما إذا كان الشرط غير محقق إذن تعليمات الحلقة لا تنفذ و للآلة ستنفذ باقي البرنامج، و الشكل العام للحلقة:

Tant que (*condition bool閑nne*) *faire*
| *Instructio ns*
fin Tant que



تمرين1: أكتب الخوارزمية التي تقوم بعرض كل الأرقام من 1 إلى 5.

*** انجاز المخطط الانسيابي:**



*** انجاز الخوارزمية:**

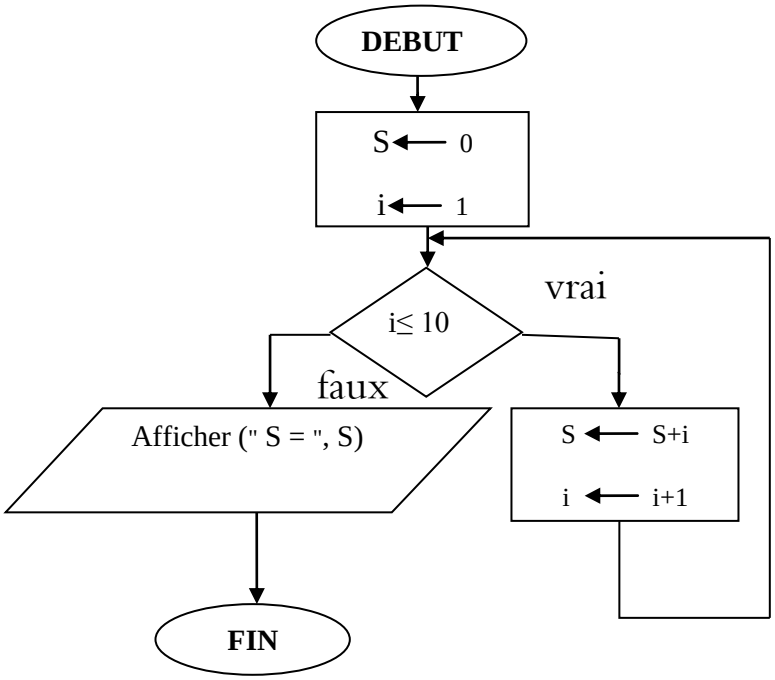
```
ALGORITHME  Affichage de 1 a 5
 var iables  i :entier
DEBUT
   i = 1
   Tant que (i ≤ 5) faire
      Afficher ("i =", i)
      i ← i + 1
   fin Tant que
FIN
```

*** انجاز البرنامج المكافئ للخوارزمية بـ ++C:**

```
#include<iostream>

Using name space std;

Int main ()

{

Int i ;

i=1;

While (i<=5)

{

Cout<<" i="<i ;

i = i+1;

};

}
```

تمرين2: أكتب الخوارزمية الذي يحسب و يعرض جمع الأعداد التالية: $S = \sum_{i=1}^{10} i$ أي S=1+2…10**.**

*** انجاز المخطط الانسيابي:**



*** انجاز الخوارزمية:**

ALGORITHME la somme de 1 *a* 10
var *iables* i, s : *entier*
DEBUT
$i \leftarrow 1$
$s \leftarrow 0$
Tant que $(i \leq 10)$ *faire*
$s \leftarrow s + i$
$i \leftarrow i + 1$
fin Tant que
Afficher $("s =", s)$
FIN

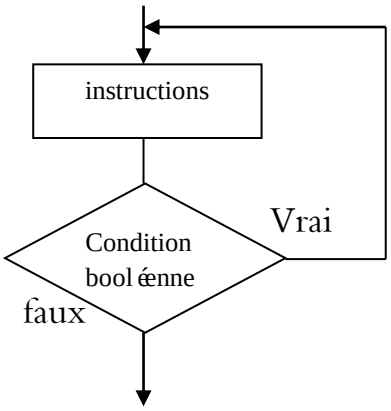
***انجاز البرنامج بـ C++:**

```
#include<iostream>

Using name space std;

Int main ()

{

Int S, i;

S=0;

i=1;

While (I<=5)

{

S = S+i;

i= i+1;

};

Cout<< "S="<<S;

}
```

2– الحلقة: Répeter……jusqu'a

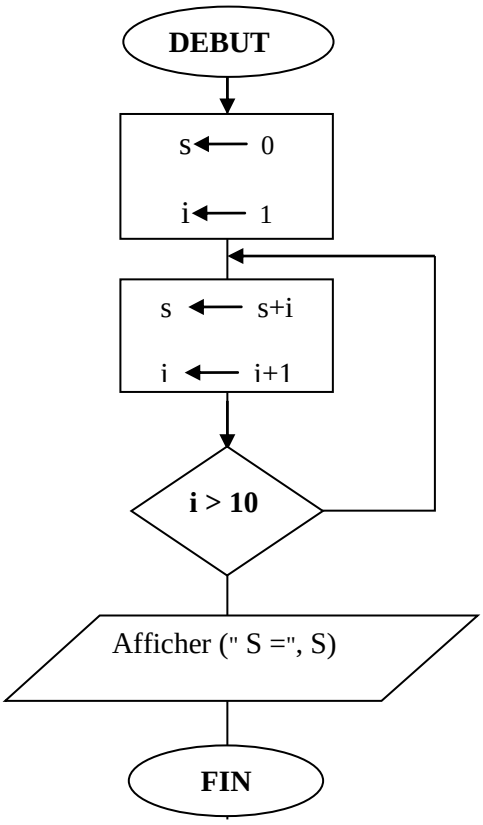
تنفذ الآلة في هذه الحلقة التعليمات الخاصة بالحلقة لأول مرة بدون شرط ثم الآلة تختبر شرط الحلقة و إذا كان شرط الحلقة غير محقق تنفذ تعليمات الحلقة للمرة الثانية، و هكذا في كل مرة عندما يكون شرط الحلقة غير محقق الآلة تنفذ تعليمات الحلقة. أما إذا كان شرط الحلقة محقق إذن تعليمات الحلقة لا تنفذ و الآلة ستنفذ بقية البرنامج، و الشكل العام كما يلي:

R閊eter
| *Instructio ns*
Jusqu'a (*condition bool閑nne*)



تمرين1: أكتب الخوارزمية التي تحسب و تعرض جمع الأعداد التالية $S = \sum_{i=1}^{10} i$ أي $S = 1 + 2 + \ldots + 10$.

*** انجاز المخطط الانسيابي:**



***انجاز البرنامج بـ C++:**

ALGORITHME la somme de 1 *a* 10
var *iables* i, s : *entier*
DEBUT
  $i \leftarrow 1$
  $s \leftarrow 0$
  R閊eter
    $s \leftarrow s + i$
    $i \leftarrow i + 1$
  Jusqu'a $(i > 10)$
  Afficher $("s =", s)$
FIN

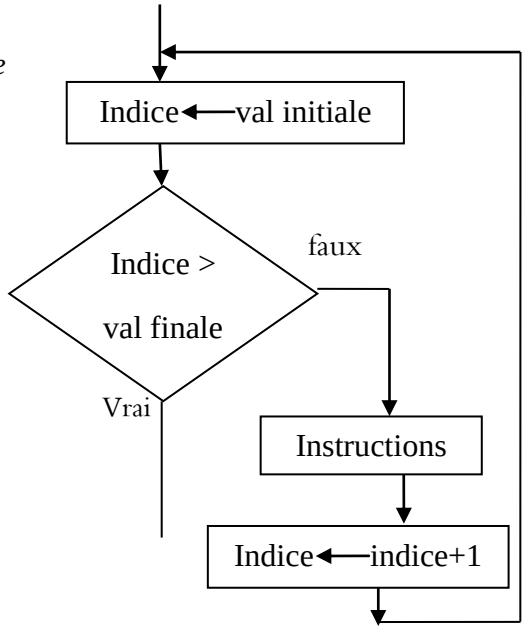
*** انجاز البرنامج:**

```
#include<iostream>

Using name space std;

Int main ()

{

Int S, i;

S=0;

i=1;

do{

S = S+i;

i= i+1;

} While (I<=5) ;

Cout<< "S="<<S;

}
```

3- الحلقة pour:

التعليمة "pour" هي تعليمة تكرارية تسمح بتنفيذ سلسلة من التعليمات عدة مرات على عكس تعليمات الحلقة "tant que" و"repéter"، التعليمة "pour" تستخدم في حالة رقم الحلقة معلوم. و شكلها يكون كما يلي:

pour i ← val initiale a val finale faire
| Instructio ns
fin pour



* اسم التغيير(indice) variable.

* القيمة الأولي valeur initiale.

* القيمة النهائية valeur finale.

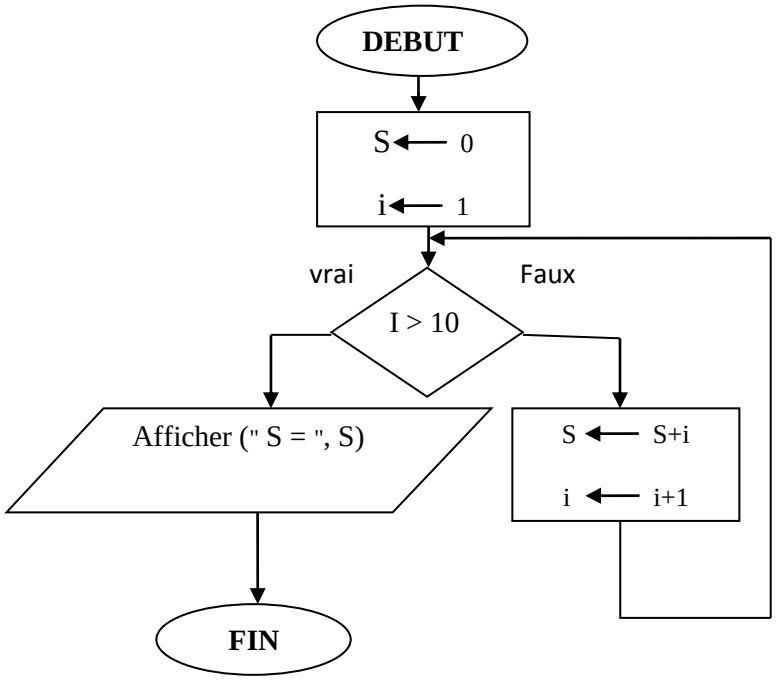
* القيمة الأولى و القيمة النهائية تكون أعداد صحيحة.

* الآلة تسند للمتغير indice « »القيمة الأولى ثم نقارنها مع القيمة النهائية، إذا كانت القيمة الأولى أقل من أو يساوي القيمة النهائية تنفذ تعليمات الحلقة « pour » و إلا تنفذ بقية البرنامج.

ملاحظة:مؤشر الحلقة « pour » يزداد آليا و هذا في نهاية كل حلقة (indice←indice+1).

تمرين1: أكتب الخوارزمية التي تحسب و تعرض جمع الأعداد التالية $S = \sum_{i=1}^{10} i$ أي $S = 1 + 2 + \ldots + 10$.

*** انجاز المخطط الانسيابي:**



*** انجاز الخوارزمية:**

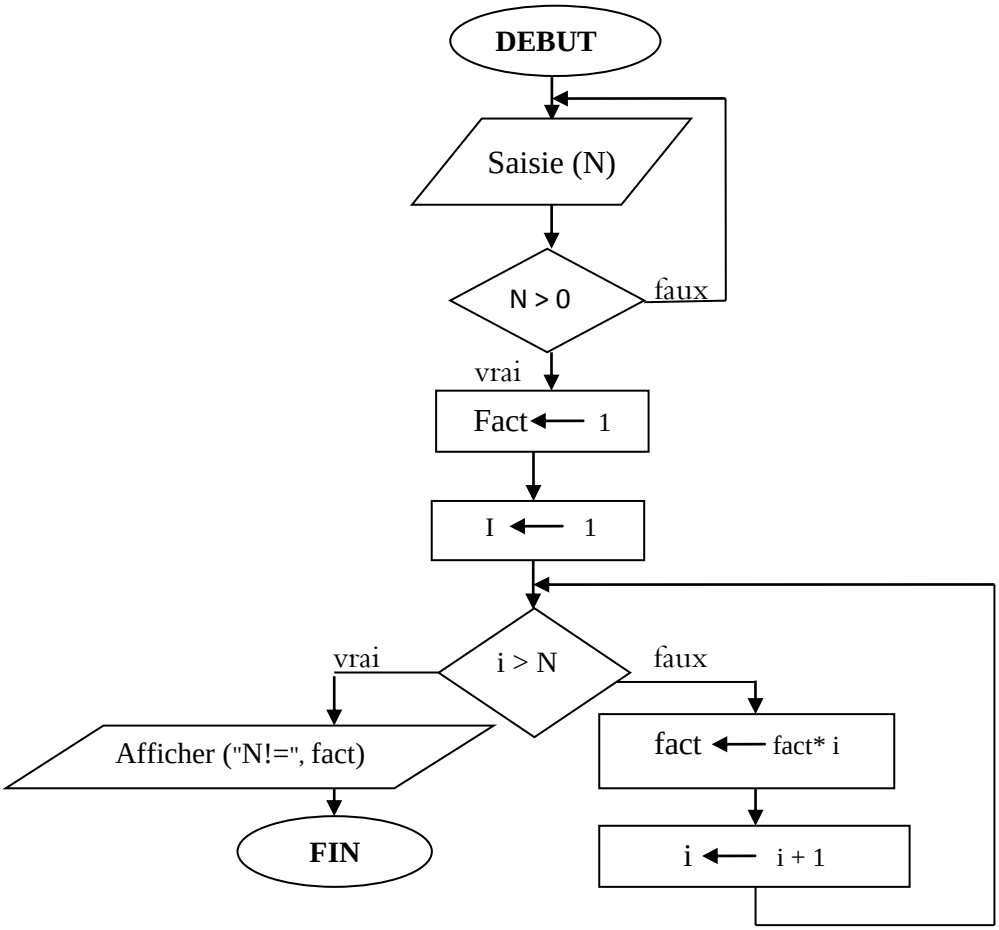
```
ALGORITHME   la somme de 1 a 10
 var iables  i, s : entier
DEBUT
   i ← 1
   s ← 0
    pour i ←1 a 10  faire
      s ← s + i
      i ← i +1
    fin  pour
    Afficher ("s =", s)
FIN
```

***انجاز البرنامج بـ C++:**

```
#include<iostream>

Using name space std;

Int  main ()

{

Int S, i;

S=0;

for (i=1;I <= 10;i++)

{

S = S+i;

} ;

Cout<< "S="<<S ;

}
```

تمرين1: أكتب الخوارزمية التي تحسب N عاملي.

*** انجاز المخطط الانسيابي:**



*** انجاز الخوارزمية:**

```
ALGORITHME  factoriele
Variables  N,i, fact  :Entier
DEBUT
   R閊eter
   |Saisir( N)
   Jusqu 'a  (N < 0)
   i ← 1
   fact ← 1
   pour  i ←1 a  N  Faire
   |  fact ←  fact * i
   |  i ← i +1
   Fin  pour
   Afficher(" fact =", fact)
FIN
```

***انجاز البرنامج بـ C++:**

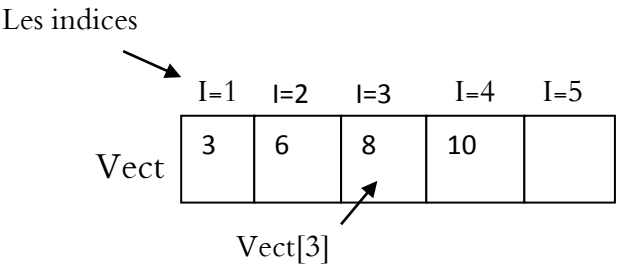
```
#include<iostream>

Using name space std;

Int main ()

{

Int N, i, fact;

do{

cin>>N;

} While (N<0) ;

Fact=1;

 for (i=1; I ≤ N; i++)
```

```
{

fact = fact*i;

} ;

Cout<< "N!= "<<fact;

}
```

الجداول les tableaux : الجدول هو تجميع للمتغيرات من نفس النوع وله اسم خاص به، كل متغيرات الجدول مرقمة، هذا الرقم يسمى المؤشر، كل متغير للجدول محدد له اسم خاص به ومؤشر له.

بشكل عام فإن الجداول نوعان (هناك جداول متعددة المستويات هي خارج نطاق المحاضرة) و جداول أحادية البعد و جداول ثنائية البعد.

1- الجداول أحادية البعد (les vecteurs):المصفوفة أحادية البعد هي شعاع أو شكل من الخانات المرتبة قرب بعضها البعض ذات مؤشرات متتالية كما في الشكل التالي:



Vect: اسم المصفوفة.

I: مؤشر العنصر في الجدول.

Les valeurs: كل قيمة في الجدول تتبع اسم المصفوفة و مؤشر المصفوفة.

مثال:

Vect [1] ⟵ 3

Vect [2] ⟵ 6

Vect [3] ⟵ 8

ملاحظات:

1- المؤشر (L'indice) لا يكون أبدا عدد حقيقيا.

2- المؤشر ربما يكون عبارة

Si i=3 alors T [i] ⟵T [3]

Mais T [3 * 10 + 3]⟵T[39]

3- الجدول ربما يخصص للجدول الآخر من نفس النوع لتخصيص أحادي

V [i] =T [i]

التصريح بالجدول أحادي البعد:

var *iables* < *nom tableau* >:*tableau* [*taille*]< *type el開ent* >

مثال: ليكن الجدول ذو 20 عنصرا، و لهذا التصريح يكون كما يلي:

var *iables Vect* : *tableau*[20]*de re開le*

* الكتابة (التعبئة) في الجدول الأحادي البعد:

pour i ← 1*a* 20 *faire*
| *saisir* (*Vect*[*I*])
fin pour

العرض في الجدول الأحادي البعد:

pour i ← 1*a* 20 *faire*
| *Afficher* (*Vect*[*I*])
fin pour

تمرين1:ليكن الشعاع vect الذي يحتوي على نقاط 15 طالبا في مادة معطاة.

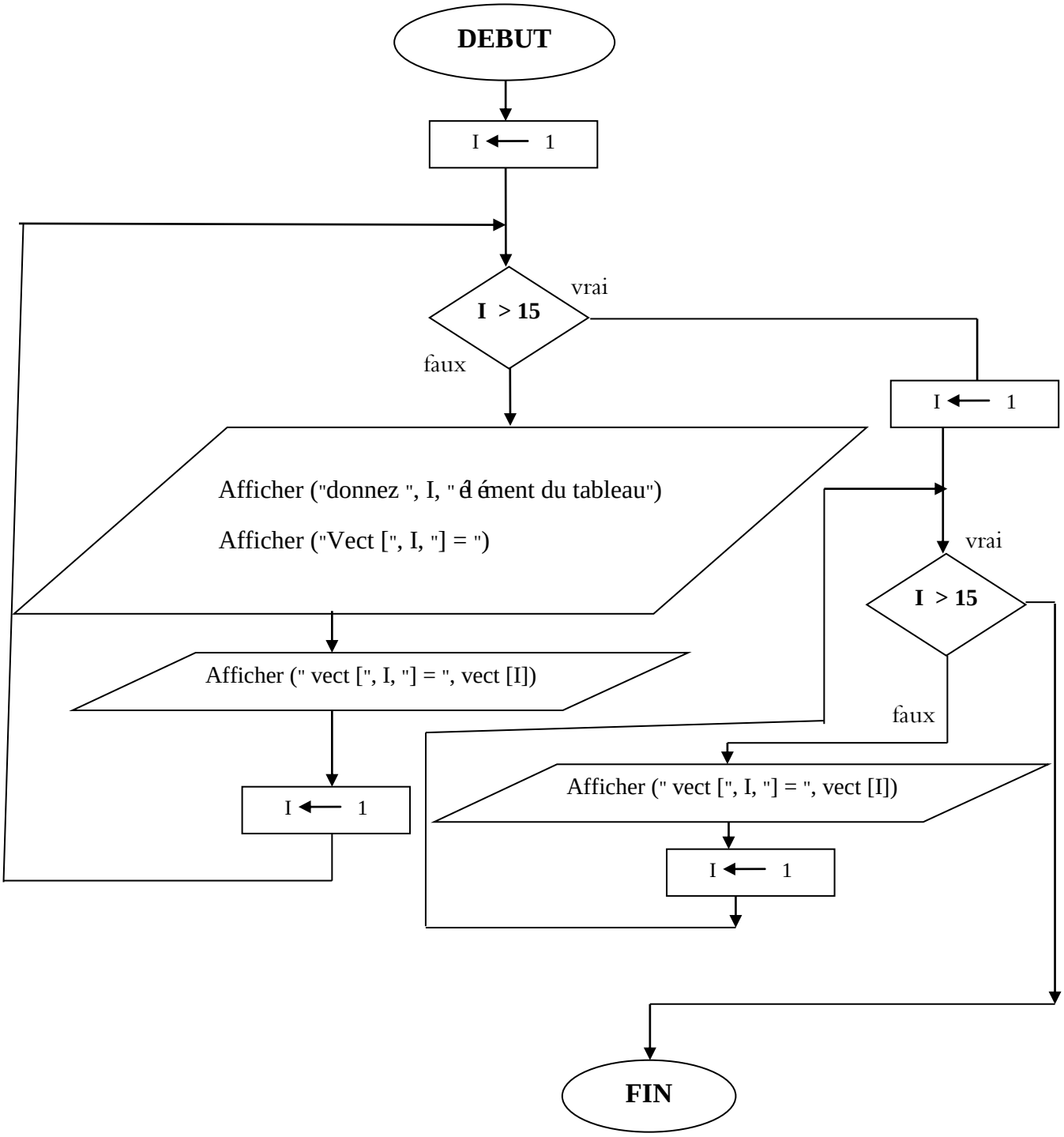
المطلوب: أكتب الخوارزمية التي تملأ ثم تعرض نقاط الطلبة لهذا الشعاع.

المعطيات:

1. حجم الشعاع 15 .
2. نوع المعطيات حقيقي
3. بعد الشعاع واحد.

*

* **انجاز المخطط الانسيابي:**



*** انجاز الخوارزمية:**

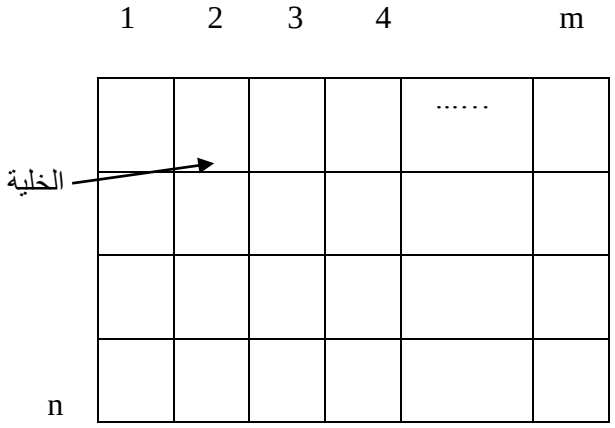
```
ALGORITHME  vecteur
Variables  Vect : tableau [15] : Réel
                          i : Entier
DEBUT
   pour  i ← 1 a 15  faire
      Afficher ("donnez le", i, "element du tableau")
      Afficher ("Vect[", i, "] =")
      Saisir (Vect[i]
   fin pour
   pour  i ← 1 a 15  faire
      Afficher ("Vect[", i, "] =", Vect[i])
   Fin  pour
FIN
```

***انجاز البرنامج بـ C++:**

```
# include <iostream >

Using name space std;

Int Main ()

{

float Vect [15];

Int  I;

For (I=1; I<=15; I++)

{

Cout << «Donnez le  «<<I<< élément »;

Cout<< «vect [ «<<I<<"]= «;

Cin>>vect [I];
```

```
}

For (I=1; I<=15; I++)

{

Cout<<"vect ["<<I<<"] =" <<vect [I];

}

}
```

2- الجدول ذو بعدين: الجدول ذو بعين هو مصفوفة مركبة من أسطر و أعمدة في ذات الوقت، و لهذا تقاطع الأسطر والأعمدة يعطي مكان الخلية، و الشكل التالي يوضح ذلك:



ملاحظات:

* الجدول ذو بعدين (المصفوفة) يحتوي على اسم و يسجل كما يلي LM [n,m]

* LM: تمثل عدد المصفوفة.

* n :تمثل عدد أسطر المصفوفة.

* m :تمثل عدد أعمدة المصفوفة .

* عناصر المصفوفة تكون كلها من نفس النوع.

* عناصر المصفوفة تعين بواسطة مؤشرات:

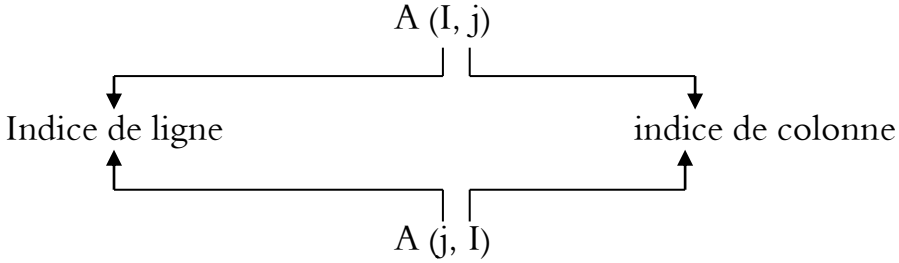
* مؤشر السطر(I).

* مؤشر العمود(J).

مثال:

LM (I, J): تمثل عنصر كائن لتقاطع السطر Iو العمودJ.

LM (2,5): تمثل عنصر كائن لتقاطع السطر 2 و العمود5.



4- المؤشرات Iو Jعددان صحيحان و يمكن أن تكون المؤشرات عبارات نتيجتها أعداد صحيحة.

مثال:

Si I=2 et j=3 Alors M (I+1, J-1)<=> M (3, 2)

5- في الحالة التي تكون عدد الأسطر يساوي عدد الأعمدة تسمى مصفوفة مربعة و تسجل كما يلي M(n,n).

* **التصريح بالمصفوفة:** التصريح بالمصفوفة بالطريقة التالية:

var *iables* < *nom tableau* >:*tableau* [*n* , *m*]< *type el開ent* >

Variable Nom.tableau : tableau [n, m] de type-élément

مثال: لتكن المصفوفة A لها 10 أسطر و 20 عمودا لها عناصر من نوع حقيقي، يصرح بهذه المصفوفة كما يلي:

var *iables A*:*tableau* [10 , 20]*de R閧*

* **قراءة المصفوفة (تعبئة)**:تعبئة المصفوفة تكون بالشكل التالي:

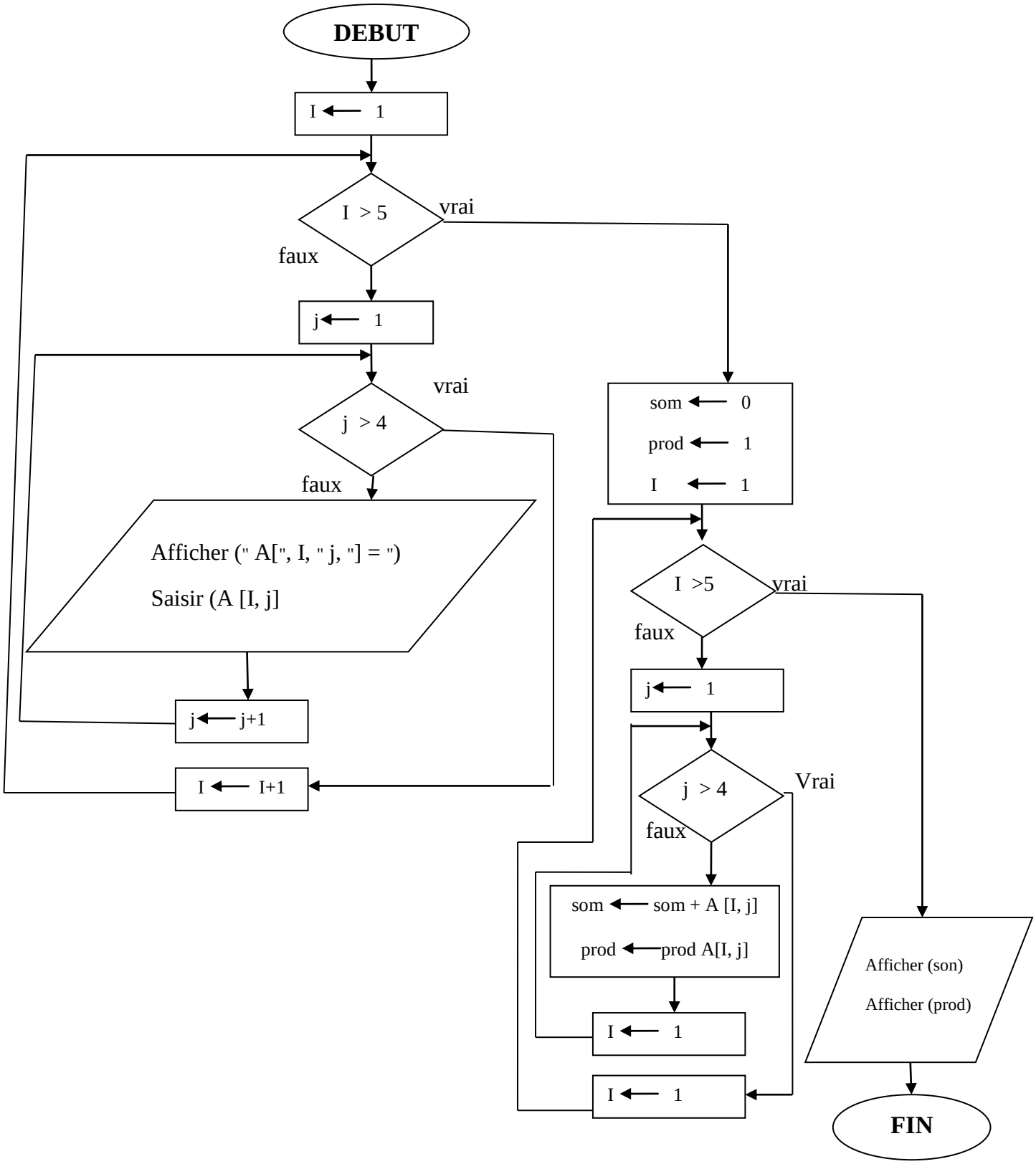
```
pour I ←n a m faire
    pour J ←n a m faire
        Afficher ("A[",I,"]=")
        Saisir (A[I,J])
    fin pour
fin pour
```

* **كتابة (عرض) المصفوفة**: لعرض عناصر المصفوفة، نتبع الطريقة التالية:

```
pour I ←1 a m faire
    pour J ←1 a m faire
        Afficher ("A[",I,"]=",A[I,J])
    fin pour
fin pour
```

تمرين1:لتكن المصفوفة A(n, m) حيث (m=4, n=5) أعداد حقيقية، أكتب الخوارزمية التي تحسب جمع و ضرب عناصر هذه المصفوفة.

* **انجاز المخطط الانسيابي:**



* انجاز الخوارزمية:

```
ALGORITHME  som et prod
Variables  A : tableau[5,4] : Réel
           som, prod : Réel
           I, J : Entier
DEBUT
  pour I ← 1 a 5 faire
    pour J ← 1 a 4 faire
      Afficher ("[", I, J, "] =")
      Saisir (A[I, J])
    fin pour
  fin pour
  som ← 0
  prod ← 1
  pour I ← 1 a 5 faire
    pour J ← 1 a 4 faire
      som ← som + A[I, J]
      prod ← prod * A[I, J]
    fin pour
  Fin pour
  Afficher ("la somme des éléments est =", som)
  Afficher ("le produit des éléments est =", prod
FIN
```

***انجاز البرنامج بـ C++:**

```
# include <iostream >

Using name space std;

Int Main ()

{

float A [5] [4];

Int   I,J;

For (I=1; I<=5; I++)

  For (J=1; I<=4; J++)

{

Cin>>A [I] [J];

}

Som=0;prod=0;

For (I=1; I<=5; I++)

  For (J=1; I<=4; J++)

{

Som=som+A [I] [J];

Prod=prod* A [I] [J];

}

Cout<<"som="<<som;

Cout<<"prod="<<prod;

}
```

البرامج الفرعية:

تعريف البرنامج: هو عبارة عن مجموعة من التعليمات و الأنشطة المكتوبة داخل البرنامج الفرعي، و يتم استدعائها من الداخل البرنامج الرئيسي في الخوارزميات يوجد نوعين من البرامج الفرعية و هما الاجراءات و الدوال.

الهدف من كتابة و استخدام البرامج الفرعية:

1- تقليل حجم البرنامج.

2- التقليل من مخاطر الأخطاء.

3- جودة سهولة قراءة البرنامج.

4- توفير لنسخ لنفس التعليمات المتسلسلة.

5- سهولة المراقبة و التحديث.

- مجال المتغير في البرنامج الفرعي:

*- المتغير المعرف على مستوى البرنامج يسمى المتغير المحلي، و مجال المتغير المعملي فقط على مستوى البرنامج الفرعي الذي يصرح فيه (أي أن هذه التعليمات الخاصة بالبرنامج الفرعي تستخدم هذا المتغير في حدود البرنامج الفرعي).

*- المتغير المعرف على مستوى البرنامج الرئيسي يسمى المتغير العام، مجاله كلي و كل برنامج فرعي من البرنامج الرئيسي يمكنه استخدام هذا المتغير و كذلك استخدامه في البرنامج الرئيسي.

*– عندما يكون اسم المتغير المحلي له نفس الاسم مع المتغير العام، فالمتغير العام لا يظهر استخدامه على مستوى البرنامج الفرعي.

شكل الخوارزمية: الخوارزمية يصبح شكلها كما يلي:

ALGORITHME « Nom d'algorithme »

Définition de constances

Définition de type

Définition de variables globales

Définition de sous-programme

DEBUT

تعليمات البرنامج الرئيسي

FIN

وسائط البرنامج الفرعي:

وسيط البرنامج الفرعي هو متغير محلي خاص مرتبط بالمتغير أو الثابت للبرنامج الرئيسي المستدعي.

– لأن الوسيط هو متغير محلي و الوسيط لديه نوع.

– عندما يستدعي البرنامج الرئيسي البرنامج الفرعي يجب تحديد المتغير أو الثابت و يكون من نوع النوع للمتغير المرتبط به.

تمرير الوسائط:

يوجد ثلاثة أنواع من الروابط التي تربط بين الوسيط و المتغير أو الثابت للبرنامج الرئيسي المستدعي.

- تمرير وسيط الإدخال) التمرير بواسطة القيمة).
- تمرير وسيط الإخراج.
- تمرير وسيط الإدخال و الإخراج (التمرير بواسطة المرجع).

1- تمرير وسيط الإدخال:

تعليمات البرنامج الفرعي لا يمكن أن يتغير متغيرات أو ثوابت للبرنامج الرئيسي المستدعي.

* عندما يمرر متغير البرنامج الرئيسي المستدعي يم نسخ قيمته و إعطائها مكان لتخزين مؤقت ضمن البرنامج الفرعي على الوسائط. و أن أي تغير في الوسيط ضمن البرنامج الفرعي عليه المتغير سيبقى محافظا على قيمته السابقة للبرنامج الرئيسي قبل الاستدعاء.

* هذا النوع من التمرير يسمح بتمرير ثابت.

مثال:

- البرنامج الفرعي Sqr تسمح بحساب الجذر التربيعي لوسيط وحيد.

- البرنامج الفرعي Afficher يسمح بعرض المعلومات ولديها N وسيط إدخاله.

2. تمرير وسيط الإخراج:

تعليمات البرنامج أسند قيمة لوسيط البرنامج الرئيسي المستدعي

* يوجد ارتباط قوي بين الوسيط و كيان البرنامج الرئيسي المستدعي، و لهذا لا يمكن استخدام الثابت كنوع للوسيط.

* قيمة المتغير المرتبط بالبرنامج الرئيسي المستدعي لا تستخدم آي البرنامج الفرعي.

3. تمرير وسائط الإدخال و الإخراج:

تمرير هذا النوع من الوسيط يجمع بين النوعين السابقين.

* حينما يستخدم البرنامج اللفرعي يجب استخدام و تغيير قيمة المتغير للبرنامج الرئيسي المستدعي، و هذا مثل تمرير وسيط الاخراج.

* هذا النوع من التمرير لا يسمح باستخدام الثابت.

1. الإجراءات:الإجراءات هي برامج فرعية و التي لا تعيد أية نتيجة، و تكون عملية تمرير الوسائط كما يلي:

* المدخلات تكون مسبوقة بـ Entrée أو E.

* المخرجات تكون مسبوقة بـ Sorties أوS.

* المدخلات/المخرجات تكون مسبوقة بـ Entrée/Sortie أو E/S .

***– عملية التصريح تكون كما يلي:**

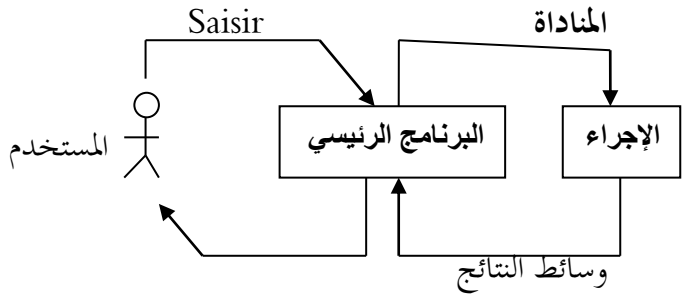
Procédure Nom de procédure (E paramètres en entée, S paramètres en sortie, E/S paramètre en entre/sortie)

Déclaration variables locales

Début

تعليمات الإجراءات

Fin



ملاحظة: الاجراء يمكن أن يقبل الوسائط من نوع واحد من التمرير أو نوعين من التمرير أو ثلاثة أنواع من التمرير.

تمرين: أكتب الخوارزمية التي تحتوي على إجراء échange الذي يسمح بتبديل قيم متغيرين.

*** انجاز الخوارزمية:**

```
ALGORITHME  échanger deux variables
variables a,b : Entier
procedurre  échanger (E / S  val01 : entier , E / S val02 : entirt )
 variables  temp : Entier
debur
   temp ← val01
   val01 ← val02
   val02 ← temp
fin  procedure
DEBUT
   Afficher ("entrez deux variables")
   échanger (a,b)
   Afficher ("a =",a,"b =",b)
FIN
```

*انجاز البرنامج بـ C++:

```
# include <iostream>

Using name Space Std;

Void échange (int val1, int val2)

{

Int temp;

Temp = val 1;

Val 1 = val 2;

Val 2 = temp;

}

Int main ( )

{

Int a, b;

Cin>>a>> b;

Echanger (a, b)

Cout << " a = " <<a<< " b " <<b ;

}
```

2- الدوال: الدوال هي برامج فرعية تسمح بوجود وسائط و ترجع نتيجة واحدة فقط(مثل الدوال الرياضية ($Y=f(x,y...)$))، الدوال لها مجموعة من الخصائص:

* الدالة لها نوع واحد فقط، و الذي هو من نفس القيمة المسترجعة.

* تمرير الوسيط في الدوال يكون إلا من وسيط الإدخال فقط.

* القيمة المسترجعة خاصة بالتعليمات المسترجعة.

* الدالة لها اسم .

***- التصريح بالدوال يكون كما يلي:**

Fonction nom de la fonction (paramètre de la fonction): type de la valeur retrouvée

Déclaration variables locales

Début

| تعليمات الدالة

Fin fonction

تمرين: باستخدام الدوال في الخوارزمية، أكتب الخوارزمية التي تقوم بحساب القيمة المطلقة لعدد صحيح.

* انجاز الخوارزمية:

```
ALGORITHME absolue01
var iables a, b : Entier
Fonction abs (unEntier : entier) : Entier
 var iables valeurabsolue : Entier
debur
   si (unEntier ≥ 0) Alors
   | vaeurabsolue ← unEntier
  sin on
  | valeurabsolue ← − unEntier
   fin si
  retourner valeurabsolue
 fin fonction
DEBUT
   Saisir (a)
  b ← abs (a)
  Afficher ("lavaleur absolue de =", a", b)
FIN
```

***انجاز البرنامج بـ C++:**

```
# include <iostream>

Using name S pace Std;

Int abS (int unEntrier)

{

Int valeurabsolue;

IF (unEntier>=0)

{

Valeur absolue = un Entier;

}

ELSE

{

Valeurabsolue = - unEntier;

}

Retourne valeur absolue;

}

Int main ( )

{

Int   a,b;

cin >> a;

b = abs (a);

Cout << "la valeur absolue de"<<a<< " est  "<<b ;}
```

قائمة المراجع

قائمة المراجع:

1- لخضر لعلام ،الخوارزميات وتقنية البرمجة بلغة GW.BASIC،ديوان المطبوعات الجامعية ،1998.

2- زيان جيلاني ،100 تمرين في الخوارزميات ،دار هومة للطباعة والنشر - الجزائر،.2007

3- Mc BELAID ،Initiation à L'Algorithmique Exercices corrigés et comment ،pages Bleues internationales ،2012.

4- Moussa KHIEDER ،Méthode Pratique pour Maitriser la programmation ،2008.

5- BAGHDALI-OURBIH Latifa ،ALGORITHMIQUE cours et exercices corrigés ،2010.

6- Chahid KHICHANE ،le deader de l'algorithmique Cours et exercices avec leur corrigés ،el MAARIFA Editions ،2004.

7- Said BELLOUT ،d'algorithmique & exercices avec solutions ،2008.