لغات البرمجة (*Les Langages de Programmation*)

البرمجة (*Programmation*) هي عملية إعداد وتطوير البرامج من خلال تحديد الخطوات الأساسية التي تتبعها الآلة من أجل حل مشكل ما. وتتم عملية البرمجة باستخدام لغة يختارها المبرمج. لكل لغة برمجة (***Langage de Programmation***) خصائصها التي تميزها عن الأخرى وتجعلها مناسبة بدرجات متفاوتة لحل المشكل المطروح. كما أن لها بالمقابل خصائص وحدود مشتركة بحكم أنها صممت للتعامل مع الكمبيوتر.

I. تطور لغات البرمجة:

يوجد حاليا المئات من لغات البرمجة التي تختلف في أهدافها ووظائفها، وقد مرت هذه اللغات بمراحل طويلة من التطور حتى وصلت لشكلها الحالي:

1. لغـــــة الآلـــــة (*Langage Machine*):

تسمى اللغة الثنائية (*langage binaire*) لاعتمادها على سلسلة من الرقمين 0 و 1، وهي اللغة الوحيد التي يفهمها الكمبيوتر، تتميز بالصعوبة البالغة نظراً لما تتطلبه من حفظ ودقة في كتابة سلسلة طويلة من 0 و 1 بترتيب معين، مما ينتج عنه أخطاء كثيرة، كما تتطلب معارف تقنية دقيقة بتركيب الكمبيوتر الداخلي وخصائصه، ومن بين سلبيات البرمجة بلغات الآلة استحالة نقل برنامج لتنفيذه على نوع آخر من أجهزة الكمبيوتر باعتبار أن لكل جهاز لغة آلة تختلف عن الجهاز الآخر حسب النوع والتركيب.

2. لغـــات التجميع (*Les Langages Assembleur*):

ظهرت لغة التجميع بوصفها لغة ترميز، تستخدم الرموز *Symbol* للتعبير عن تعليمات لغة الآلة، وذلك لمواجهة صعوبة لغة الآلة، حيث تم تعويض سلاسل 0 و 1 باختصارات لكلمات انجليزية مثل *ADD*، *MOV* ... ورغم أن البرمجة بلغة التجميع أسهل بكثير من لغة الآلة، إلا أن لغة التجميع أقرب من لغة الآلة منها الى لغة الإنسان، حيث تتطلب معارف تقنية للتركيب الداخلي للكمبيوتر، مما يصعب عملية نقل البرامج من جهاز لآخر.

3. اللغات الراقية (*Les Langage Evaluées*):

تعبيرات اللغات الراقية شبيهة إلى درجة كبيرة باللغة الطبيعية التي يستخدمها الإنسان حيث أصبح بإمكان المبرمج بفضل لغات البرمجة الراقية كتابة البرامج دون معرفة تفاصيل كيفية قيام الكمبيوتر بتنفيذها، كمواقع التخزين وخصائص الجهاز الدقيقة.

ومن مميزات اللغات الراقية أنها غير مرتبطة بجهاز معين. أي أنه يمكن تنفيذ البرنامج على أكثر من جهاز، كما أن اكتشاف الأخطاء وتصحيحها أصبح أكثر سهولة بسبب سهولة قراءة البرامج وتتبعها وفهمها. من أمثلة لغات البرمجة الراقية:

- *(1954) FORTRAN*
- *(1959) COBOL*
- *(1964) BASIC*
- ***(1970) PASCAL***
- *(1972) C*
- *(1983) ADA*
- *(1983) C++*
- *(1991) JAVA*
- *(2000) C#*
- *(2009) GO*
- *(2011) DART*

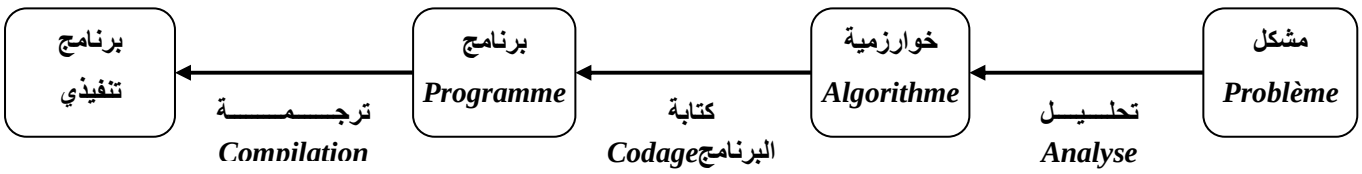
3. لغات الذكاء الاصطناعي (*LES langages d'Intelligence Artificielle*):

مع منتصف الخمسينات من القرن الماضي ظهرت موجة الاهتمام بالذكاء الاصطناعي. نشأ هذا الاهتمام الكبير بسبب رغبة اللغويين بالتعرف على معالجة اللغات الطبيعية، و علماء النفس بسبب محاولة محاكاة المعلومات الإنسانية وأخيرا علماء الرياضيات بسبب الرغبة في أتمتة اثبات النظريات. والشيء المشترك بين كل هذه التطبيقات هو الحاجة الى طريقة تسمح للكمبيوتر بمحاكاة العقل البشرى حيث تعتمد على برمجته كي تتعامل وتفكر كالإنسان وتتخذ القرارات مثله. من أمثلة لغات الذكاء الاصطناعي: ***Prolog***

ملاحظـــــة هامـــــــة: أغلب لغات البرمجة تعتمد على اللغة الانجليزية إلا أنه يمكن تسجيل بعض المجهودات لعلماء عرب طوروا لغات تعتمد كليا على اللغة العربية مثل لغة **"ج"** لمطورها **محمد عمار السَلكة (*http://www.jeemlang.com*).** ولغة **"زاي"** لمطورها الجزائري **الأستاذ زقور جمال الدين (*http://www.zegour.uuuq.com*)**

II. مراحل اعداد برنامج:

إعداد برنامج ما عملية جد معقدة تتطلب عدة مراحل، أهمها تحديد الهدف النهائي من البرنامج والتقيد به. يتم عادة تقسيم معالجة المشكل المطروح إلى مجموعة متسلسلة من العمليات أقل حجما وتعقيدا. الشكل التالي يبين مراحل كتابة برنامج:



1. التحليل: تحليل المسألة ووضع طريقة الحل هو أصعب مراحل إعداد البرنامج، حيث يتم في هذه الخطوة تحديد:

- طبيعة المخرجات (النتائج) وتنظيم كتابتها.
- المدخلات (البيانات أو المعلومات) وتحديد نوعها و تنظيم إدخالها إلى الكمبيوتر.
- طرق الحل المناسبة و تقييمها بما يتلاءم مع كيفية تنفيذها وفي ضوء ذلك يتم اختيار الحل الأمثل.

بعد اختيار طريقة الحل المثالية وتحديد كل ما تشمله من علاقات رياضية، يتم التعبير عنها على شكل خطوات متسلسلة ومترابطة منطقياً، تؤدي إلى حل المشكل المطروح. هذه الخطوات تعرف **بخوارزمية** المشكل (***Algorithme***) ويمكن تمثيلها بمخطط وصفي تسلسلي يسمى (**المخطط البرمجي *Organigramme***) وذلك باستخدام مجموعة من الأشكال الاصطلاحية الرمزية.

2. كتابة البرنامج باستعمال لغة برمجة: الخوارزميات غير معدة لتنفيذها مباشرة على الجهاز لذلك يجب كتابتها بإحدى لغات البرمجة التي يختارها المبرمج وفقا لعدة معايير أهمها قدرات المبرمج ونوع المشكل المطروح. لغة البرمجة **باسكال *Pascal*** من بين أشهر لغات البرمجة نظرا لبساطتها وقدرتها على التعامل مع أغلب المشاكل خاصة تلك المتعلقة بعلوم الإدارة والتسيير.

يتم الحصول في نهاية هذه المرحلة على ملف يختلف نوعه حسب لغة البرمجة المستعملة (***pas.*** في حالة لغة باسكال)، يسمى **البرنامج المصدر (*Code Source*)**، ولا يطلع عليه إلا المبرمج نظرا لاحتوائه على كل ما يتعلق بحل المشكل المطروح.

3. ترجمة البرنامج المصدر: البرنامج المصدر مكتوب بلغة راقية لا يفهمها المعالج المركزي، لذلك يجب ترجمته إلى لغة الآلة بواسطة برنامج الترجمة (*Compilateur*) الخاص بلغة البرمجة المستعملة. يتم الحصول بعد الترجمة على ملف مكتوب بلغة الآلة قابل للتنفيـذ مباشـرة نوعـه " ***exe.*** " وهو الملف الذي يحصل عليه المستعمل النهائي للبرنامج عادة (دون البرنامج المصدر الخاص بالمبرمج).

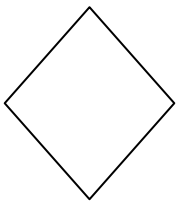
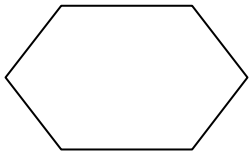
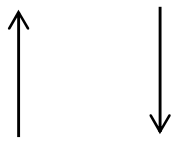
III. الخوارزميات والمخططات البرمجية (*Les Algorithmes et les Organigrammes*)

1.III. مقدمـــة عـامـــــــة

الخوارزميــة (*Algorithme*) هي مجموعة من العمليات المتسلسلة التي تؤدي إلى حل مشكل ما. وسميت الخوارزمية بهذا الاسم نسبة إلى العالم المسلم **أبو جعفر محمد بن موسى الخوارزمي** الذي ابتكرها في القرن التاسع الميلادي.

المخططات البرمجية (*Les Organigrammes*) هي وصف تصويري لخطوات الخوارزمية لتبسيط تتبع الخوارزمية وتوضيحها. والمخططات البرمجية تقوم مقام الخوارزمية ويمكن بواسطتها ملاحظة تتبع التسلسل المنطقي لحل المشكل بكل سهولة.

الجدول التالي يبين الأشكال الرمزية الاصطلاحية المستعملة لرسم المخطط البرمجي:

Description	**الوصف**	**الرمـــز**
Début / Fin de l'organigramme	بداية / نهاية المخطط البرمجي	***Debut*** ***Fin***
Entrées de données ou Sorties des résultats	ادخال المعطيات **أو** اخراج النتائج	
Instruction Simple	تعليمة بسيطة	
Condition	شرط	
Boucle	حلقة (تكرار)	
Connexion	اتصال	
Orientation	اتجاه تسلسل العمليات	

ملاحظـــــة: سيتم تفصيل رموز المخطط البرمجي بالموازاة مع تفصيل تعليمات الخوارزميات.

2.III. اصطلاحات عامـــــة

قبل التفصيل في بنية الخوارزميات سيتم ابتداء تحديد بعض المفاهيم الهامة جدا وهي:

1. التعليقات (*les Commentaire*): هي نصوص توضيحية يكتبها **المبرمج** في أي مكان من الخوارزمية للتعليق والتوضيح فقط. حيث لا يتم تنفيذها ولا حتى قراءتها من طرف منفذ الخوارزمية (المعالج المركزي في حالة البرنامج)، تعتبر هامة جدا خاصة في حالة الخوارزميات الكبيرة.

تختلف طريقة كتابة التعليقات من لغة لأخرى، في الخوارزميات نصطلح الكتابة التالية:

- **اذا كان التعليق يقل عن سطر واحد //:**

```
//ceci est un commentaire
```

- **اذا كان التعليق يزيد عن سطر واحد(* *):**

```
(*ceci est un commentaire
قد يكون التعليق مكتوبا بالعربية
وهذا مثال على ذلك*)
```

2. الكلمات المحجوزة (*les Mots Réservés*): تستعملها الخوارزمية بشكل حصري ولا يمكن استعمالها في تسمية **المُعرفات** من أمثلة الكلمات المحجوزة في الخوارزميات: *Début, Fin, Si*

3. المُعَرِف (*l'Identifiant*): هي الكلمات التي يستعملها المبرمج لتسمية مختلف العناصر التي يستعملها لكتابة الخوارزمية، مثل أسماء الخوارزميات، أسماء المتغيرات، أسماء الثوابت، ...الخ. لا توجد قواعد لاختيار اسم المعرف في الخوارزميات لكن هناك شروط يجب أن تتوفر في اسم المعرف:

- يجب ألا يكون اسم المعرف **كلمة محجوزة.**
- يجب ألا يبدأ اسم المعرف برقم بل بحرف a…z أو A…Z أو الرمز _
- يجب ألا يحتوي اسم المعرف على فراغ (***Espace***).
- يجب عدم استعمال نفس الاسم لمعرفان في نفس الخوارزمية.

3.III. بنيـــــة الخوارزميــة

الخوارزمية غير معدة لتنفيذها مباشرة على الجهاز بل يجب ترجمتها الى لغة برمجة، لذلك يمكن القول بأنها بمثابة المسودة التي يستعملها الطالب قبل نقل اجاباته على الورقة الرسمية، لذا ورغم المحاولات الجدية لتوحيد رموز الخوارزمية إلا أن اختلافا كبيرا لا يزال موجودا في كتابتها. عادة يمكن استعمال أي لغة لكتابة الخوارزمية سواء أكانت: العربية، الفرنسية، الانجليزية... لكن ولأسباب أكاديمية بحتة سيتم استعمال اللغة الفرنسية لكتابة الخوارزمية في هذه المحاضرات.

الكتابة العامة للخوارزمية تكون بالشكل التالي:

القسم الأول: اسم الخوارزمية

Algorithme Nom;

Constantes

C1=Valeur1 ***;***

C2 = Valeur2 ***;***

…..

القسم الثاني: التصريحـــــات

Variables

V1,V2 **,** …Vn ***:*** Type1 ***;***

Va **,**Vb **,** …Vz ***:*** Type2 ***;***

……

Debut

Inst1 ***;***

Inst2 ***;***

Inst3 ***;***

القسم الثالث: التعليمـــــات

……

Instn ***;***

Fin.

القسم الأول اسم الخوارزمية (*Nom de l'Algorithme*)

الكتابة العامة:

Algorithme Nom**;**

حيث:

Algorithme : كلمة محجوزة

Nom : اسم الخوارزمية يختاره المبرمج وفق قواعد تسمية المعرفات.

;: رمز نهاية اسم الخوارزمية.

أمثلـــــــة:

Algorithme Exo1 **;**

Algorithme Facturation **;**

Algorithme Moyenne **;**

القسم الثاني قسم التصريحات (*Déclarations*)

هو القسم الذي يقوم فيه المبرمج بالتصريح عن الثوابت، المتغيرات ... التي يستخدمها في الخوارزمية.

ملاحظة هامة: في هذا المستوى سنقتصر على دراسة تصريحات الثوابت والمتغيرات دون التصريحات الأخرى كالدوال والاجراءات...

الكتابة العامة:

Constantes

C1=Valeur1 ;

C2 = Valeur2 ;

…..

التصريح بالثــوابت ان وجدت

Variables

V1,V2 , …Vn : Type1 ;

Va ,Vb , …Vz : Type2 ;

……

التصريح بالمتغيرات ان وجدت

1. التصريح بالثوابت:

الثوابت: الثابت هو عنصر ذو قيمة معلومة ابتداءا (عند التصريح) لا تتغير أثناء تنفيذ الخوارزم، يتم التصريح باسمه وقيمته في قسم التصريحات:

Constantes

C1=Valeur1 ;

C2 = Valeur2 ;

…..

حيث:

Constantes : كلمة محجوزة.

C1: اسم الثابت (معرف).

Valeur1: القيمة الابتدائية والنهائية للثابت C1.

;: رمز نهاية التصريح الحالي.

مثـــال

Constantes

الثابت Pi قيمته ابتداءا 3.14 ولا يمكن تغييرها أثناء تنفيذ الخوارزمية. // Pi= 3.14;

الثابت TVA قيمته ابتداءا 0.17 ولا يمكن تغييرها أثناء تنفيذ الخوارزمية. // TVA= 0.17 ;

ملاحظة: النص الذي يلي الرمز // هي تعليق فقط (راجع فقرة **التعليقات**)

2. التصريح بالمتغيرات:

المتغير هو عنصر يمكن **لقيمته** أن تتغير أثناء تنفيذ الخوارزمي، يتم التصريح **باسمه وبنوعه** في قسم التصريحات.

Variables

V1,V2 , …Vn **:** Type1 **;**

Va ,Vb , …Vz **:** *Type2* **;**

…..

حيث:

Variables : كلمة محجوزة.

V1: اسم المتغير (معرف).

Type1: نوع المتغير.

;:رمز نهاية التصريح الحالي.

خصائص المتغير

أ. الاسم (*Nom*): تخضع تسمية المتغير لقواعد تسمية المعرف (*l'Identifiant*)

ب. النوع (*Type*): مجال تعريف المتغير (القيم التي يمكن أن يأخذها المتغير أثناء تنفيذ البرنامج)، يجب التصريح بها مسبقا في قسم التصريحات.

الجدول التالي يبين أهم الأنواع المستخدمة في الخوارزميات:

النوع	المجال	أمثلـــــــة
entier	الأعداد الصحيحة السالبة والموجبة	…-3 , -2 , -1 , 0 , 1 , 2 , 3 …
réel	الأعداد الحقيقية السالبة والموجبة	…-3…-2…-1…0…1…2, 3 …
caractère	حرف أو رمز نصي	'a'…'z' , 'A' … 'Z' , 'é'…'à' , '@', …'1'…'9'
Chaine	مجموعة من الأحرف أو الرموز النصية	'Bonjour' , 'Cours LMD 2éme'
booléen	القيمتين المنطقيتين: Vrai، Faux	Vrai ، Faux

والجدول التالي يبين أهم **العمليات الحسابية والمنطقية** التي يمكن أن تستعمل على تلك الأنواع:

النوع	رمزالعملية	الوصف	أمثلة
entire, réel	+ , -	عمليات حسابية أحادية الحد ***Unaire***	+ 5.25 = 5.25 - (+ 5) = - 5
entire, réel	+ , - , * , /	عمليات حسابية ثنائيات الحد ***Binaire***	عمليات حسابية عادية
entier	mod ,div	عمليات حسابية ثنائيات الحد ***Binaire***	- القسمة الصحيحة (Div) 14 Mod 5 = 2 - باقي القسمة (Mod) 14 Mod 5 = 4
Chaine	+	عملية حسابية ثنائية الحد ***Binaire***	تقوم + بتركيب سلسلتين حرفيتين **'LMD'+'2' = 'LMD2'**
Booléen	et	عملية منطقية ثنائية الحد **الوصل**	**A et B = VRAI** اذا كانت A و B قضيتان صحيحتان
Booléen	ou	عملية منطقية ثنائية الحد **الفصل**	**A ou B = VRAI** - إذا كانت A قضية صحيحة **أو** B قضية صحيحة
Booléen	non	عملية منطقية أحادية الحد **النفي**	**non A = VRAI** - اذا كانت A قضية خاطئة **non A = FAUX** - اذا كانت A قضية صحيحة

كما يمكن استعمال رموز المقارنة العادية:> ، < ، = ، >= ، =< (أكبر، أقل، تساوي، أكبر من أو تساوي، أقل من أو تساوي).

ج. القيمة (Valeur): لكل متغير قيمة تنتمي إلى مجال تعريفه (النوع) وقد تتغير أثناء تنفيذ الخوارزمية، عندما يتم التصريح بالمتغير تكون قيمته الابتدائية (*Valeur Initiale*) **مجهولة** (عادة) لذلك يجب عدم استعمال متغير ما بدون معرفة قيمته الابتدائية بل يجب أولا إعطاء المتغير قيمة ابتدائية قبل استعماله.

ملاحظة هامـــة: استعمـال متغير قيمتـه الابتدائيـة غير معلومـة خطر. (*L'utilisation d'une variable non initialisée est un danger!*)

مثال عن التصريح بالمتغيرات:

```
Variables
    nombre1, n2 : réel ;          // التصريح بمتغيرين حقيقين قيمتهما مجهولة
    nb2: entier ;                 // التصريح بمتغير صحيح قيمته مجهولة
    v1 : caractère ;              // التصريح بمتغير حرفي قيمته مجهولة
    v2 : chaine;                  // التصريح بمتغير نصي قيمته مجهولة
    v3 : booléen ;                // التصريح بمتغير منطقي قيمته مجهولة
```

القسم الثالث قسم التعليمات (*Instructions*)

التعليمـــة (*Instruction*): هي الأوامر التي تنفذ من طرف الجهاز يقوم المبرمج بكتابتها بين الكلمتين المحجوزتين ***Debut*** و ***Fin.*** تنتهي التعليمة بالرمز **;**

ملاحظـــــة: يجب عدم الخلط بين التعليمة والعملية. **التعليمة** تنتهي بنقطة فاصلة **;** وتستعمل عادة ما يسمى **بالعملية** (***Opération***) التي قد تكون عملية حسابية (***Arithmétique***) (… / , * , - , +) أو منطقية (***Logique***) (…non , ou , et).

الكتابة العامـــة

```
---------------------------------------------------------------------------
Debut                   بداية قسم التعليمات//
        Inst1 ;         التعليمة الأولى //
        Inst2 ;         التعليمة الثانية //
        Inst3 ;         التعليمة الثالثة //
        ……
        Instn ;         التعليمة الأخيرة //
Fin.                    نهاية قسم التعليمات//
---------------------------------------------------------------------------
```

أهم التعليمات المستعملة في الخوارزميات

1. التعليمة الأولى: تعليمـــــة الاسنـــاد (*Affectation*)

الكتابـــــة العامـــــة:

V←expression

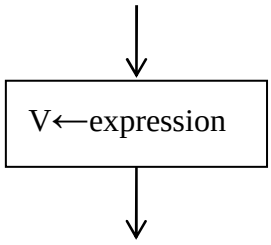
حيث:

V: متغير (مصرح به طبعا في قسم التصريحات).

expression: عبارة قد تكون:

- ثابت
- متغير
- قيمـــة
- عملية (حسابية أو منطقية).
- ...

في المخطط البرمجـــي:



تنفيـــــــذ التعليمـــة: يقوم المنفذ (المعالج المركزي في حالة البرنامج):

- **أولا** بحساب قيمة expression الموجودة على يمين رمز الاسناد (←).
- **ثانيا** يقوم بإسناد تلك القيمة الى المتغير (**V**) الموجود على يسار (←).

ملاحظة هامة: يجب أن يكون نوع قيمة expression (على اليمين) **متوافق** مع نوع المتغير (على اليسار).

أمثلة عن عملية الاسناد

Algorithme exemple_Affectation;

Constantes

a1=20;

Variables

x1,x2 ***:*** réel ***;***

Debut ---------- **0**

x2 ← 5***;***

---------- **1**

x1 ← a1***;***

---------- **2**

x1 ← x2***;***

---------- **3**

x1 ← 20,12;

---------- **4**

x2 ← x2 *3;

---------- **5**

Fin.

المرحلة	قيمة x1	قيمة x2	قيمة a1
0	؟	؟	20
1	؟	5	20
2	20	5	20
3	5	5	20
4	20,12	5	20
5	20,12	15	20

2. التعليمة الثانية: تعليمـــــة الادخال (*Lecture*)

الكتابـــــة العامـــــة:

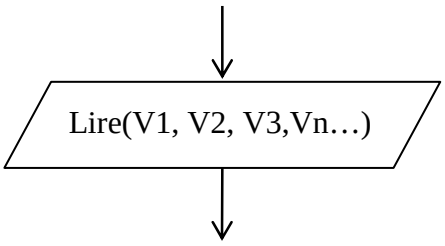
***Lire*(V1, V2, V3…) ;**

حيث:

Lire : كلمة محجوزة (رمز التعليمة).

V1 : متغير.

في المخطط البرمجـــي:



تنفيـــــذ التعليمـــة: يقوم المنفذ (المعالج المركزي في حالة البرنامج):

- **أولا** بالتوقف عن تنفيذ الخوارزمية وانتظار ادخال **n** قيمة من طرف مستخدم الخوارزمية.
- **ثانيا** يقوم بإسناد القيم التي يدخلها المستخدم للمتغيرات V1، V2 ، ...Vn على الترتيب.

ملاحظة هامة: قد يُدخل المستخدم قيمة لا تتوافق مع نوع المتغير، مما يؤدي الى خلل في تنفيذ الخوارزمية.

3. التعليمة الثالثة: تعليمـــــة الاخراج (*Ecriture*)

الكتابـــــة العامـــــة:

***Ecrire*(P1, P2, P3…) ;**

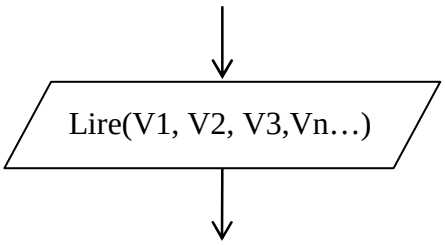
حيث:

Ecrire : كلمة محجوزة (رمز التعليمة).

P1 : قد تكون:

- عبارة (***expression***) (ثابت، متغير، قيمة، عملية حسابية أو منطقية...)
- نص مهما كان موجود بين عاكفتين **' '**

في المخطط البرمجـــي:



تنفيـــــذ التعليمـــة: يقوم المنفذ (المعالج المركزي في حالة البرنامج):

- بعرض ما بين قوسين على الشاشة:

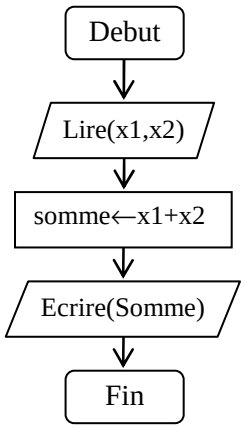
فإذا كان ما بين قوسين **عبارة** ما يقوم بعرض قيمتها مهما كانت.

وإذا كان ما بين قوسين موجود بين عاكفتين **''**يقوم بعرضه كما هو على أساس أنه نص.

أمثلة عن تعليمتي الادخال والإخراج:

1. الخوارزمية التالية تقوم بعرض مجموع عددين يدخلهما المستخدم:

```
Algorithme Exemple_Lecture_Ecriture;
Variables
        x1,x2, somme :réel ;
Debut
        Lire (x1, x2);
        somme←x1+x2
        Ecrire (somme);
Fin.
```



4. التعليمة الرابعة: تعليمـــــة الشرط (Condition): هناك نوعان من تعليمات الشرط في الخوارزميات: تعليمة الشرط البسيط، وتعليمة الشرط المتعدد.

أ. تعليمة الشرط البسيـط

الكتابـــــة العامـــــة:

```
Si( condition ) Alors
        Inst1 ;
        Inst2 ;
        Inst3 ;
        ……
FinSi ;
```

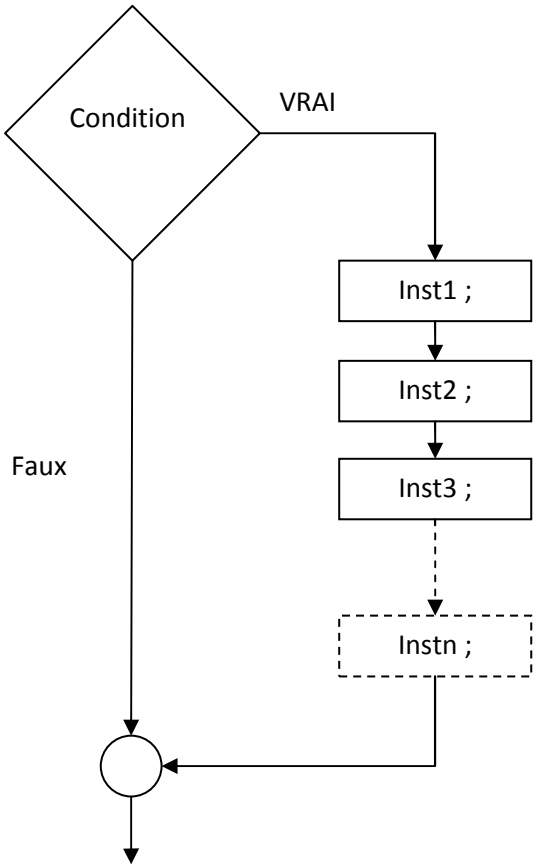
حيث:

***Si …Alors…FinSi*:** كلمات محجوزة (رمز التعليمة).

condition : عبارة منطقية نتيجتها إما صحيح أو خطأ (Vrai / Faux).

Inst1: تعليمة من التعليمات التي تم أو سيتم التطرق لها.

في المخطط البرمجـــي:



تنفيـــــــذ التعليمـــة: يقوم المنفذ (المعالج المركزي في حالة البرنامج):

أولا بحساب وتحديد قيمة العبارة المنطقية *Condition*. (ستكون طبعا إما صحيح أو خطأ Vrai / Faux)

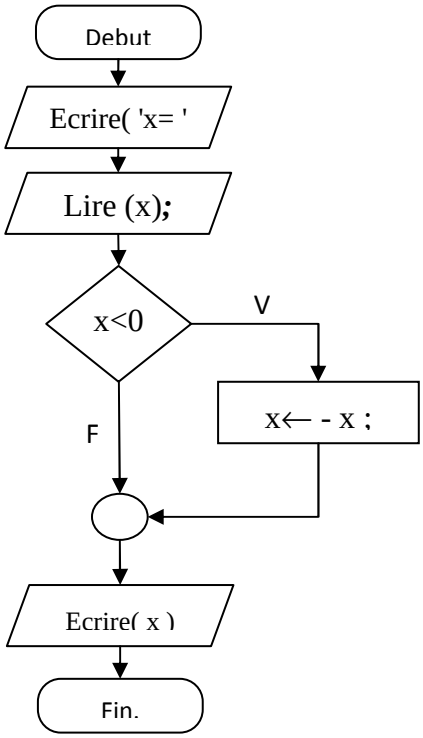
ثانيا:

- اذا كان الشرط (Condition) صحيح سيقوم المنفذ بتنفيذ التعليمات inst1 ، inst2 ، inst3 ... الى غاية الكلمة **FinSi ;**
- اذا كان الشرط (Condition) خاطئا لا يقوم المنفذ بتنفيذ التعليمات inst1 ، inst2 ، inst3 ... بل يذهب مباشرة لتنفيذ ما بعد الكلمة **FinSi ;**.

أمثلـــــــة:

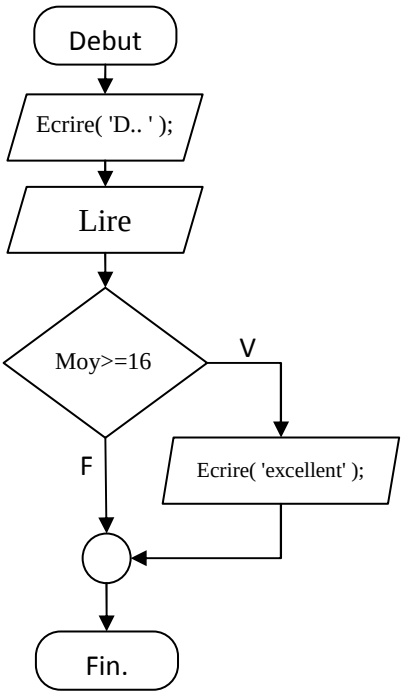
1. الخوارزمية التالية تقوم بحساب القيمة المطلقة لعدد ما وتعرض النتيجة على الشاشة:

```
Algorithme Valeur_Absolue;
Variables
        x :réel ;
Debut
        Ecrire( 'x= ' );
        Lire (x);
        Si (x<0) Alors
                x← - x ;
        FinSi ;
        Ecrire(x) ;
Fin .
```



2. الخوارزمية التالية تعرض ***Excellent*** اذا كان المعدل أكبر من 16:

```
Algorithme MoyExc;
Variables
        Moy :réel ;
Debut
        Ecrire( 'Donnez la Moyenne= ' );
        Lire (Moy);
        Si (Moy>=16) Alors
                Ecrire( 'excellent' );
        FinSi ;
Fin .
```



ب. تعليمة الشرط المتعدد

الكتابـــــة العامـــــة:

```
Si( condition ) Alors
        Inst1 ;
        Inst2 ;
        Inst3 ;
        ……
SiNon
        InstA ;
        InstB ;
        InstC ;
        ……
FinSi ;
```

حيث:

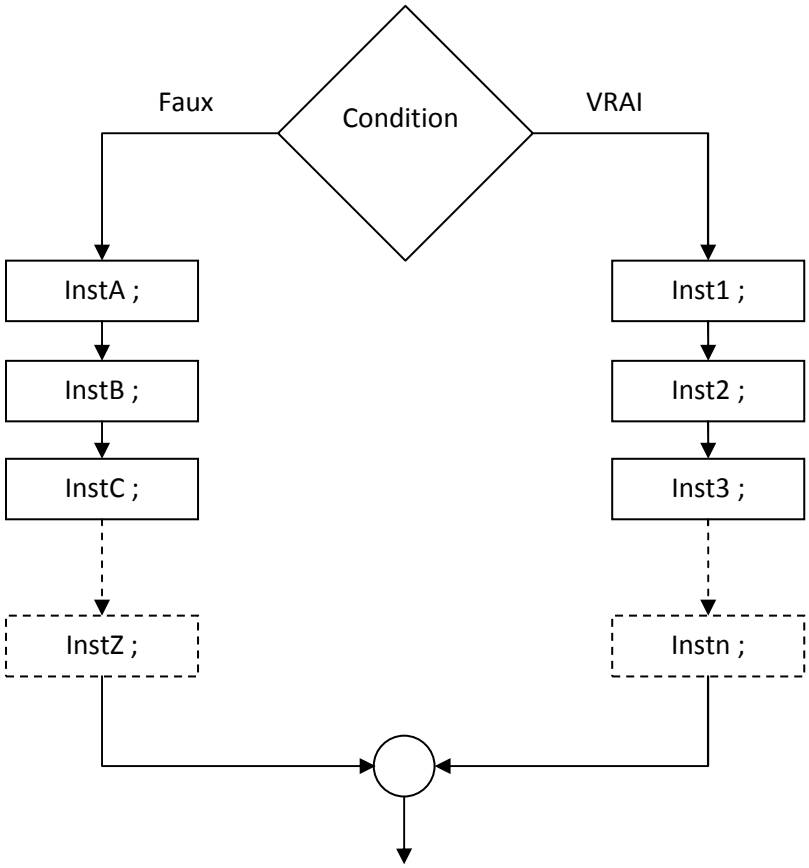
***Si …Alors…SiNon…FinSi*:** كلمات محجوزة (رمز التعليمة).

condition : عبارة منطقية نتيجتها إما صحيح أو خطأ (Vrai / Faux).

Inst1: تعليمة من التعليمات التي تم أو سيتم التطرق لها.

InstA: تعليمة من التعليمات التي تم أو سيتم التطرق لها.

في المخطط البرمجـــي:



تنفيـــــــذ التعليمـــة: يقوم المنفذ (المعالج المركزي في حالة البرنامج):

أولا بحساب وتحديد قيمة العبارة المنطقية *Condition*. (ستكون طبعا إما صحيح أو خطأ Vrai / Faux)

ثانيا:

- اذا كان الشرط (Condition) صحيح سيقوم المنفذ بتنفيذ التعليمات inst1 ، inst2 ، inst3 ... الى غاية الكلمة **FinSi ;**
- اذا كان الشرط (Condition) خاطئا سيذهب المنفذ مباشرة الى الكلمة ***SiNon*** ليقوم بتنفيذ التعليمات instA ، instB ، instC ... إلى غاية الكلمة**; FinSi**.

أمثلـــــــة:

1. الخوارزمية التالية تحسب **نسبة الخصم** حسب رقم أعمال زبون ما: نسبة الخصم تكون 5% اذا كان رقم الأعمال أقل من 1000000 دج و 10% إذا كان رقم الأعمال أكبر من 1000000 دج.

Algorithme Résultats_Etudiant;

Variables

Moy :réel ;

Debut

Lire (Moy)***;***

Si (Moy>=10) ***Alors***

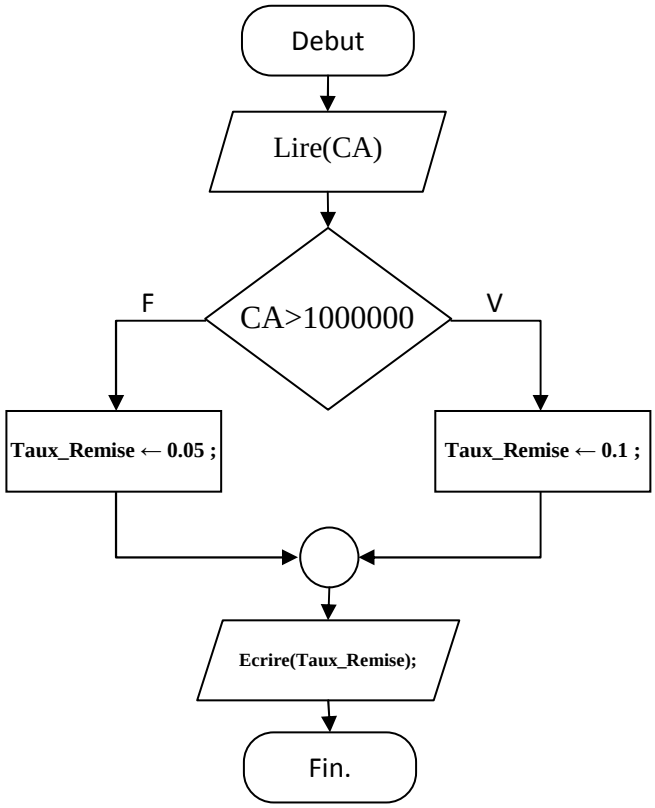
Ecrire('Admis') ;

SiNon

Ecrire(Ajournée') ;

FinSi ;

Fin .



2. الخوارزمية التالية تحدد ما إذا كان الطالب ناجح أو راسب حسب معدله.

Algorithme Calcul_Remise;

Variables

CA, Taux_Remise :réel ;

Debut

Lire (CA)***;***

Si (CA>1000000) ***Alors***

Taux_Remise ← 0.1 ;

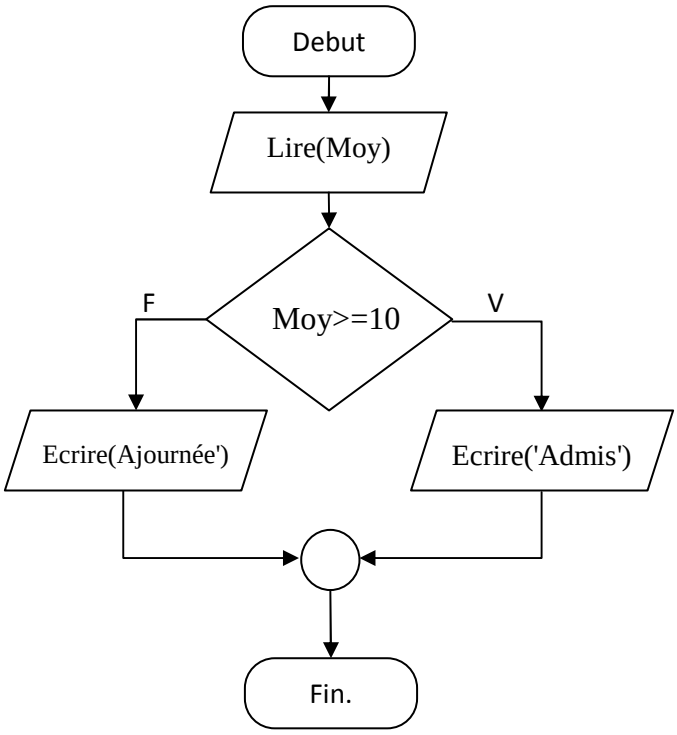
SiNon

Taux_Remise ← 0.05 ;

FinSi ;

Ecrire(Taux_Remise);

Fin .



3. الخوارزمية التالية تقوم بمناقشة حلول معادلة من الدرجة الثانية: $ax^2+bx+c=0$

تحليل بسيط: يقوم المستخدم بإدخال قيم a ، b ، c على أن يقوم المنفذ بحل المعادلة وايجاد قيمتي الحلين x1,x2 ان وجدت.

المدخـلات: a ، b ، c نوعها حقيقي (réel)

المخرجات: x1 ، x2 نوعها حقيقي (réel)

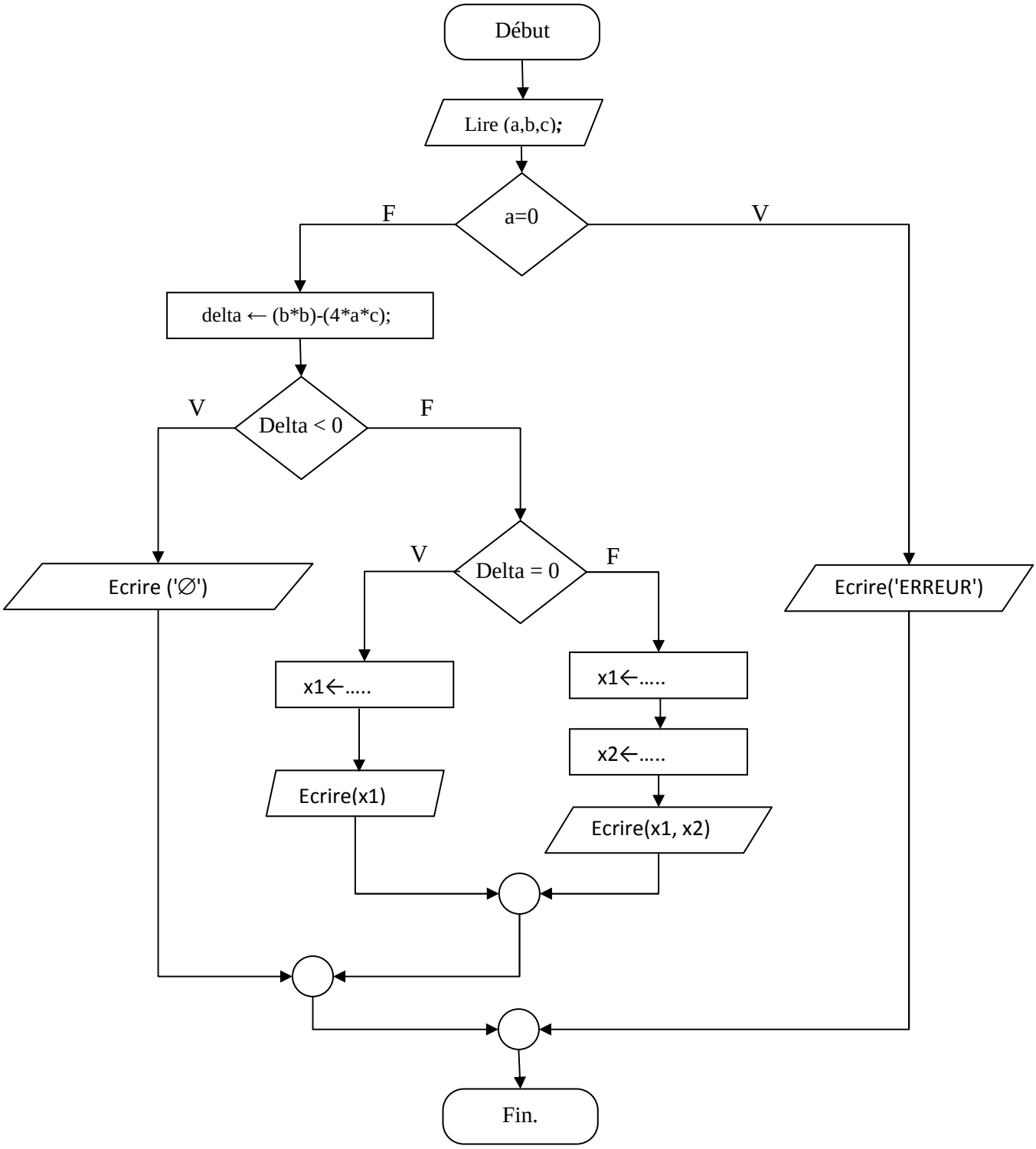
المعالجـة:

- يقوم المنفذ باختبار قيم a و b و c للتأكد من أن المعادلة صحيحة ومن الدرجة الثانية
- يقوم بحساب المميز **delta**=b^2-4ac (اذن نحتاج الى متغير آخر **delta** بالإضافة الى المدخلات والمخرجات)
- يقوم بحساب الحلول حسب اشارة **delta**

من خلال التحليل يمكن تحديد قسم التصريحات الذي لا نحتاج فيه الى ثوابت، بل الى **المتغيرات**: a ، b ، c , x2 ، x1 ، delta

```
Algorithme equation2emeDegree;
Variables
    x1, x2, delta, a, b, c: réel;
Debut
    Lire (a,b,c);
    Si (a=0) Alors
        Ecrire('Equation n'est pas du second degré') ;
    SiNon
        delta ← (b*b)-(4*a*c);
        Si (delta <0) Alors
            Ecrire('l'équation n'admet pas des solutions dans R) ;
        SiNon
            Si (delta =0) Alors
                x1 ← -b/(2*a);
                Ecrire('l'ensemble de solutions est x=', x1) ;
            SiNon
                x1← (−b − √delta )/(2 ∗ a) ;
                x2← (−b + √delta )/(2 ∗ a) ;
                Ecrire('l'ensemble de solutions est x1=', x1, 'x2=', x2) ;
            FinSi ;
        FinSi ;
    FinSi ;
Fin .
```

المخطط البرمجي



المخطط البرمجي لحل معادلة من الدرجة الثانية ذات مجهول واحد