مقدمة:

- **البرنامج Program:** هو مجموعة من التعليمات المرتبة منطقياً، الهدف منها تنفيذ عمل معين على بيانات محددة من أجل الحصول على معلومات مفيدة.
- **لغة البرمجة Programming Language:** تحدد طريقة صياغة تعليمات البرنامج من خلال مجموعة من القواعد خاصة بكل لغة.

لكتابة أي برنامج نتبع الخطوات التالية:

١. تحديد وتعريف المشكلة.
٢. كتابة الخوارزمية.
٣. تحويل الخوارزمية إلى تعليمات مناسبة.
٤. تنفيذ البرنامج لمعرفة صحة النتائج.

أولاً. تحديد وتعريف المشكلة: وفي هذه الخطوة نحدد:

- الهدف من البرنامج (حساب أرباح شركة، حساب معدل طلاب...)
- نوع وحجم المدخلات.
- نوع وحجم المخرجات.

ثانياً. كتابة الخوارزمية: لكتابة أي برنامج نحتاج إلى تحديد خوارزمية الحل، ويتم ذلك باستخدام عدة طرق منها:

- الطريقة النصية Pseudeo-code (شبه الترميز)
- الطريقة البيانية Flow Chart .

الطريقة النصية Pseudeo-code : تستخدم هذه الطريقة مجموعة تعليمات منها:

- تعليمة القراءة : اقرأ (اسم متحول)
- تعليمة الكتابة : اكتب (عبارة أو قيمة)
- تعليمة الاسناد: =
- التعليمة الشرطية : إذا (الشرط) نفذ (مجموعة تعليمات ١) وإلا (مجموعة تعليمات ٢)
- التعليمة التكرارية : مادام (الشرط محقق) كرر (مجموعة تعليمات)

مثال ١: اكتب الخوارزمية اللازمة لجمع عددين وطباعة النتيجة على الشاشة، مستخدما الطريقة النصية.

```
اقرأ  a , b
C = a + b
اكتب  "Result is : ", c
```

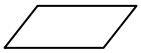
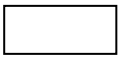
مثال ٢: اكتب الخوارزمية اللازمة لقراءة عدد موجب من المستخدم ، مستخدماً الطريقة النصية.

```
اقرأ  x
اذا ( x > 0 )  اكتب  "correct value"
والا  اكتب  "Error"
```

مثال ٣: اكتب الخوارزمية اللازمة لحساب مجموع الأعداد من ١ إلى L حيث L قيمة مدخلة من قبل المستخدم.

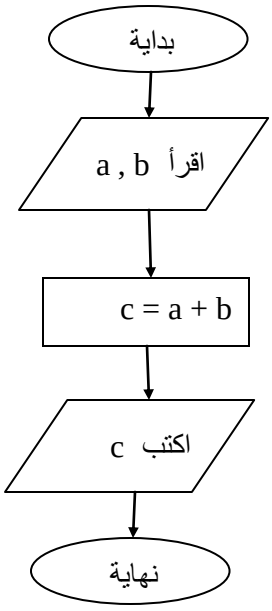
```
s= 0     (المجموع)
i= 1
اقرأ  L
مادام  ( i < = L)  كرر
s = s + i
i =  i+ 1
اكتب  " Sum is : " , s
```

الطريقة البيانية: مخططات التدفق Flow chart : وفي هذه الطريقة يتم التعبير عن الخوارزمية وخطواتها بمجموعة من الرموز والأسهم، من أهم هذه الرموز:

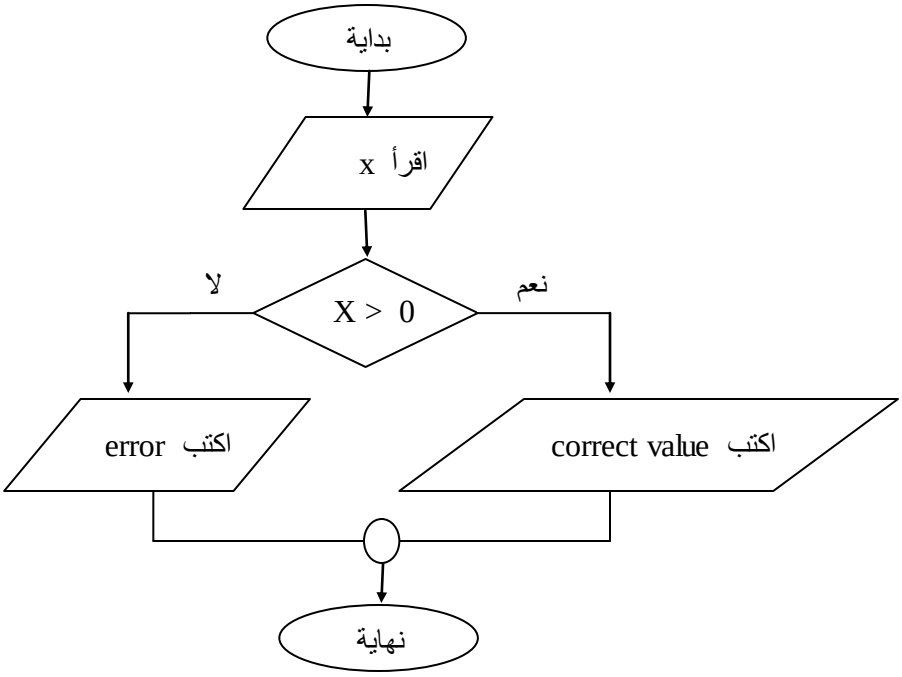
	المعنى	الرمز
start – end	بداية و نهاية	
Input – output	تعليمات القراءة و الكتابة	
Processing	الاسناد والمعالجة	
Decision	قرار	

- نستخدم الأسهم للربط بين الأشكال السابقة .
- اتجاه الأسهم يدل على تسلسل تنفيذ التعليمات

مثال ١: اكتب الخوارزمية اللازمة لجمع عددين وطباعة النتيجة على الشاشة، مستخدما الطريقة البيانية.



مثال ٢: اكتب الخوارزمية اللازمة لقراءة عدد موجب من المستخدم ، مستخدماً الطريقة البيانية.



بعد الانتهاء من كتابة الخوارزمية نختار إحدى لغات البرمجة المناسبة لصياغة أوامر البرنامج، حيث لكل لغة برمجة قواعد معينة يجب اتباعها، ولن يعمل البرنامج في حال وجود أخطاء في قواعد اللغة .

أنواع لغات البرمجة:

يوجد حالياً العديد من لغات البرمجة والتي يمكن تقسيمها إلى:

- **لغات منخفضة المستوى (Low Level Languages):** وهي اللغة التي يستطيع الحاسب أن يفهمها مباشرة وهي معروفة من قبل البنية الصلبة للحاسب. أوامر هذه اللغة مكتوبة بالنظام الثنائي أي عبارة عن سلاسل من الأصفار والواحدات (٠ و ١)، من هذه اللغات :
  - لغة الآلة Machine Languge
  - لغة التجميع Assembly Language

- **لغات عالية المستوى (High Level Languages):** سميت كذلك لأنه أصبح بإمكان المبرمج كتابة البرامج دون دراية عميقة بالتفاصيل الدقيقة التي يستخدمها الحاسب للقيام بهذه العمليات، وتعبيرات هذه اللغات تشبه إلى حد كبير التعبيرات التي يستخدمها الإنسان في التواصل مع الآخرين... تتميز بسهولة اكتشاف الأخطاء وتصحيحها ، من هذه اللغات:
  - لغة الفيجوال بيسك Visual Basic
  - لغة الباسكال Pascal
  - لغة الجافا Java
  - لغة C , C++, C#

ملاحظة: تحتاج البرامج المكتوبة باللغات عالية المستوى إلى عملية ترجمة (Compilation) باستخدام المترجم (Compiler).

- **المترجم (Compiler)**: عبارة عن برنامج يترجم البرنامج (Source Code) المكتوب بأي لغة إلى لغة الآلة (Machine Language)، وتحفظ النتيجة القابلة للتنفيذ المباشر حفظاً مستقلاً بحيث يمكن تنفيذها في أي وقت.

- **المفسر (Interpreter)**: عبارة عن برنامج يختبر البرنامج الحاسوبي تعليمة تلو الأخرى وتنفذ كل العمليات الموجودة في التعليمة الواحدة على حدة. لذا تكون هذه الطريقة بطيئة التنفيذ، وتطبق على لغات البرمجة البسيطة.

خلال عملية الترجمة قد تظهر أخطاء في صياغة البرنامج المصدر (Source code) يجب على المبرمج تصحيحها، وهنا نميز ٣ أنواع من الأخطاء:

- **أخطاء في قواعد اللغة Syntax Errors:** وهي أخطاء املائية في كتابة أوامر اللغة المستخدمة.
- **أخطاء أثناء التشغيل Run-Time Errors:** تظهر عند تنفيذ البرنامج مثل عدم حجز مساحة كافية للمتحولات، أو الدخول في حلقة لانهائية، وتظهر رسالة بنوع الخطأ.
- **أخطاء منطقية Logical Errors**: تظهر عند تنفيذ البرنامج على عينة من البيانات فنحصل على نتائج خاطئة أو غير متوقعة.

لغة البرمجة C++

الشكل العام للبرنامج في C++:

```
#include <iostream.h>
Main ()
{
  تعليمات البرنامج
}
```

السطر الأول من البرنامج يتم فيه تضمين المكتبات المطلوبة للتنفيذ، ليبدأ في السطر التالي تعريف الدالة الرئيسية main وهي الدالة التي يبدأ عندها تنفيذ البرنامج.

ضمن قوسي البداية والنهاية لهذه الدالة يتم كتابة التصريحات والتعليمات البرمجية اللازمة.

التعليمات الأساسية في C++:

- التصريح عن المتحولات Declaration:

يجب التصريح عن أي متحول قبل استخدامه في البرنامج وذلك ليقوم المترجم بحجز مكان لهذا المتحول في الذاكرة ليتم تخزين قيمة المتحول فيه ، الشكل العام للتصريح عن متحول في C++ :

```
type   name=value;
```

مثال:

```
int  num1=0;
```

هنا تم التصريح عن متحول نوعه عدد صحيح اسمه num1 وقيمته الابتدائية تساوي الصفر.

ملاحظات:

- يمكن عدم إسناد قيمة للمتحول عند التعريف. مثل: int num1;
- يمكن تعريف أكثر من متحول من نوع واحد بنفس السطر int num1, num2;

- يشترط في اسم المتحول: أن يبدأ بحرف وليس برقم، وأن لا يكون كلمة محجوزة من كلمات اللغة، وألا يحتوي على فراغات أو رموز خاصة (....,!,$,@ ,?)، ويمكن أن يتضمن أرقام.
- إذا تم التصريح عن أكثر من متحول واحد ضمن نفس السطر الواحد فإنه يجب الفصل بين أسماء المتحولات بفواصل عادية (،).
- يجب الانتباه أن لغة C++ حساسة لحالة الأحرف Case Sensitive، أي أن A,a عبارة عن متحولين مختلفين.
- يمكن وضع التصريحات في أي مكان في جسم البرنامج بشرط التصريح عن المتحول قبل استخدامه.

أنواع المتحولات:

١. المتحولات من نوع عدد صحيح: int ,long int ,short int, unsigned int ,unsigned short int, unsigned longint
٢. المتحولات من نوع عدد حقيقي أو عشري: float, double ,long double
٣. المتحولات المحرفية: char
٤. المتحولات البوليانية : bool تأخذ إحدى قيمتين (٠ و ١) او (true , false)

ويبين الجدول التالي أنواع المتحولات ومجالاتها:

نوع المتحول	المجال
Char	0 to 255
int	-32768 to 32767
Long int	-2147483648 to 2147483647
Unsigned int	0 to 65535
Float	-3.4E-38 TO 3.45E+38
Double	-1.7E-308 TO 1.7E+307

- تعليمة القراءة أو الادخال Input Statement: Cin >>

مثال١: Cin >> var ;

أي سيتم إدخال قيمة المتحول var من قبل المستخدم.

مثال٢: Cin >> x >> y;

- تعليمة الطباعة أو الاخراج Output Statement: cout <<

مثال١: cout << " welcome " ;

أي ستتم طباعة العبارة welcome على الشاشة.

مثال٢: Cout << x ;

ستتم طباعة قيمة المتحول x على الشاشة.

مثال٣: cout << x << y ;

ستتم طباعة قيمتي المتحولين x , y على الشاشة.

ملاحظة١: هناك رموز معينة تستخدم لتنسيق عملية الإظهار من أهمها:

لترك فراغ بمقدار ضغطة tab بين المخرجات	**\t**
للانتقال إلى سطر جديد	**\n , endl**

ملاحظة٢: لاستخدام تعليمتي الإدخال والإخراج لابد من تضمين المكتبة المختصة بهذا المجال وهي iostream.h وذلك من خلال كتابة التعليمة التالية قبل بداية أي برنامج: #include <iostream.h>

- **الثوابت:** الثابت قيمته لا تتغير في كامل أجزاء البرنامج، ولا يمكن تغييرها ابداً

```
const  type  name=value;
```

العمليات الحسابية في ++C: موضحة في الجدول التالي:

اسم العملية	الرمز بلغة ++c
الجمع	+
الطرح	–
الضرب	*
القسمة	/
باقي القسمة الصحيحة	%

لو أردنا كتابة المعادلة التالية بـ ++C:

$$R = 5 \times 6 + (y \div 4) - 3$$

ستكون بالشكل التالي:

$$R = ((5 * 6) + (y / 4) - 3)$$

أولويات العمليات الحسابية:

أي إذا تعرضنا لعملية حسابية معينة تحتوي على ضرب وقسمة وجمع و..... إلخ ، فإننا سنقوم بتنفيذ العمليات بالترتيب التالي:

- الأقواس: تنفذ أولاً، وإذا وجدت عبارات تحوي أقواس داخل أقواس فإننا ننفذ العمليات على الأقواس الداخلية أولاً. وإذا كان لدينا مجموعة من الأقواس جانب بعضها البعض، عندها نبدأ بالحساب من اليسار إلى اليمين.
- عمليات الرفع إلى أس معين.
- الضرب والقسمة وباقي القسمة: وإذا وجدت على نفس المستوى فإننا نقوم بتنفيذ العملية الأقرب لليسار.

- الجمع والطرح: تتفذ أخيراً.

ملاحظة : إذا تعارضت عمليتان حسابيتان لهما نفس الأولوية يتم تنفيذ العملية الأقرب لليسار.

مثال:

اكتب برنامج لإيجاد $a=\frac{5+x}{z}+\frac{y}{2.7w}$ و $b=\frac{4.5(x+2.3y)^2}{z+w}$

```
#include <iostream.h>

#include <math.h>

main()

{

        float w ,x ,y ,z, a , b;

        cout<<"Enter w, x, y ,z \n";

        cin>> w>> x>> y>> z;

        a = ( ( 5 + x ) / z ) + ( y / 2.7 * w );

        b = ( 4.5 * pow( ( x + 2.3 * y) , 2)) / ( z + w );

        cout<<"a =  "<< a;

        cout<<"\n b = "<< b;

}
```