Definition of terms Computer:

A computer is an electronic device that manipulates information, or "data." It can store, retrieve, and process data.

الكمبيوتر: هو جهاز الكتروني يعالج المعلومات او "البيانات". لديه قدره على تخزين واسترجاع ومعالجة البيانات.

Computer Algorithm: a precise specification of a sequence of instructions to be carried out by the computer to solve a given problem.

خوارزميه الكمبيوتر: هي مواصفات دقيقه لسلسله من الارشادات التي يتعين تنفيذها بواسطة الكومبيوتر ومن اجل حل مشكله معينه.

Computer Programming: developing and implementing various instructions to enable a computer to do a certain task.

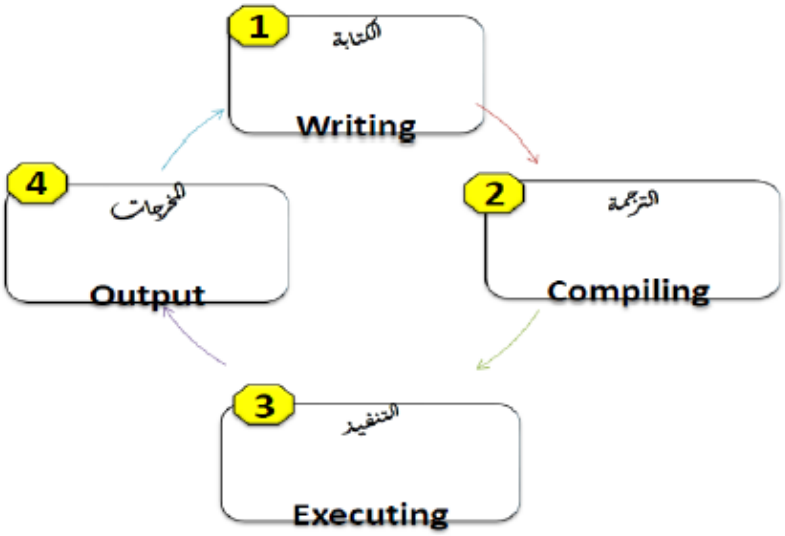
برمجة الكمبيوتر: عملية تطوير وتنفيذ مجموعات مختلفة من التعليمات لتمكين الكمبيوتر من القيام بمهمة معينه.

Programmer: an individual that composes instructions for computer systems to refer to when performing a given action.

المبرمج: هو الفرد الذي يقوم بتأليف التعليمات لأنظمة الكمبيوتر للرجوع إليها عند تنفيذ إجراء معين.

بناء البرنامج

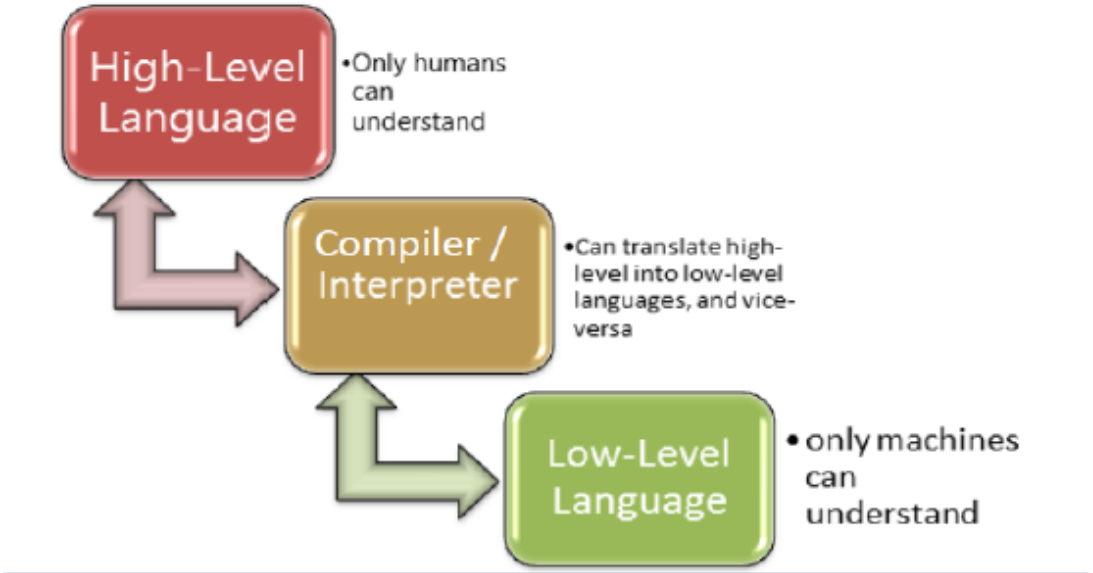
Build the Program

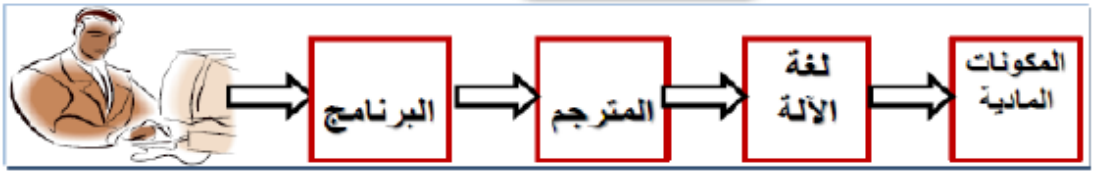


,

مراحل تنفيذ البرنامج

Program Implementation Stages





المفسّر (Interpreter) والمترجم (Compiler)

- برنامجي المفسّر والمترجم الخاصين بلغات برمجة الحاسب يقومان بترجمة اللغات العالية إلى لغة الآلة ليتمكن جهاز الحاسب من تنفيذ التعليمات والأوامر بسرعة عالية جداً.
- **برنامج المفسّر Interpreter:** يقوم بترجمة أوامر وتعليمات البرنامج المكتوب بإحدى لغات البرمجة سطراً سطراً إلى لغة الآلة أثناء قرائتها ليتم تنفيذها سطراً سطراً أيضاً. ويمتلك المفسّر القدرة على التفاعل مع البرنامج أثناء تنفيذه وإجراء أي تغيير في البرنامج ثم متابعة التنفيذ.
- **برنامج المترجم Compiler:** يقوم بترجمة أوامر وتعليمات البرنامج المكتوب بإحدى لغات البرمجة ككتلة واحدة إلى لغة الآلة أثناء قرائتها، ليتم تنفيذها دفعة واحدة، ويعتبر أسرع من برنامج المفسّر Interpreter بعدة مرات في تنفيذ البرنامج، ولكن عند وجود أي خطأ في البرنامج لابد من إصلاح الخطأ وإعادة تنفيذ كل البرنامج من جديد.

Features of a Good Computer Algorithm:

A good algorithm must possess the following features:

1. It must have a start and an end.
2. It must be effective.
3. It must be unambiguous.
4. It must terminate.
5. It must produce at least a result.
6. It may accept input or generate its input internally.
7. It must not have an unending look.

ميزات خوارزمية كومبيوتر جيدة:

يجب أن تمتلك الخوارزمية الجيدة الميزات التالية:

يجب أن يكون لها بداية ونهاية.

يجب أن تكون فعالةً.

يجب أن تكون واضحة لا لبس فيها.

يجب أن تنتهي.

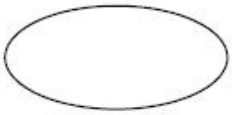
يجب أن تنتج نتيجة على الأقل.

يمكنها قبول المدخلات أو توليد مدخلاتها داخليًا.

يجب ألا يكون لها نظرة لا نهاية لها.

A flowchart: is a type of diagram that represents an algorithm, workflow or process. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem.

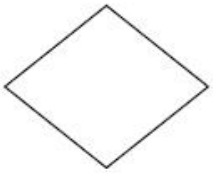
المخطط الانسيابي هو نوع من المخططات التي تمثل خوارزمية أو سير عمل أو عملية. يوضح المخطط الانسيابي الخطوات على شكل مربعات من أنواع مختلفة، وترتيبها من خلال ربط المربعات بالسهام. يوضح هذا التمثيل التخطيطي نموذج حل لمشكلة معينة.



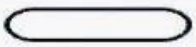
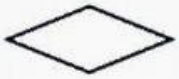
Start or End of the program

Input or Output Operations

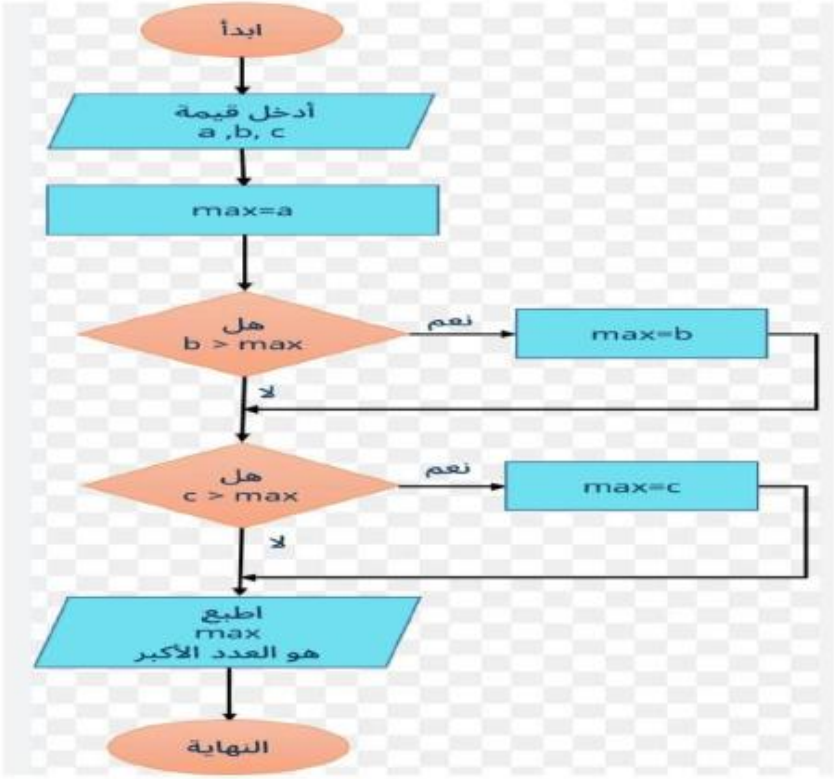
Computational steps or processing function of a program



Decision making and branching

البداية أو النهاية (Terminal)	
الإدخال أو الإخراج (Input/Output)	
معالجة أو عملية (Process)	
اتخاذ قرار (Decision)	

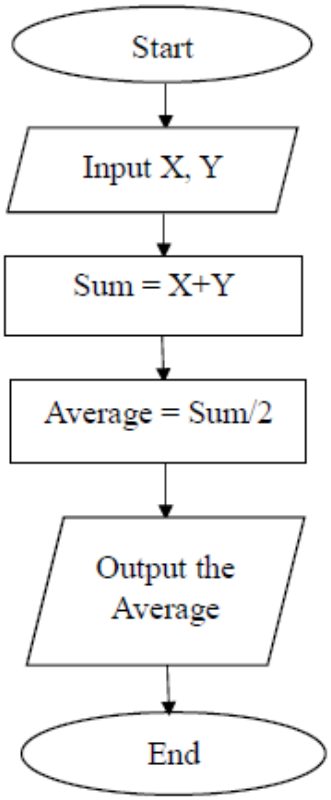
Examples



Example 1: Write an algorithm and draw the flowchart to find the average of two numbers.

To find the average of two numbers, you can follow this simple algorithm:

1. Start
2. Input the first number (x)
3. Input the second number (y)
4. Calculate the sum: sum = x + y
5. Calculate the average: average = sum / 2
6. Output the average
7. End



Example 2: Write an algorithm and draw a flowchart to calculate 2^4.

Algorithm: Step1: Input Base (2), Power (4)

Step2: Product=Base

Step3: Product=Product*Base

Step4: Product=Product*Base

Step5: Product=Product*Base

Step6: Print Product

Example 3: Write an algorithm and flowchart to convert temperature from Celsius to Fahrenheit.

C: temperature in Celsius

F: temperature Fahrenheit

Algorithm

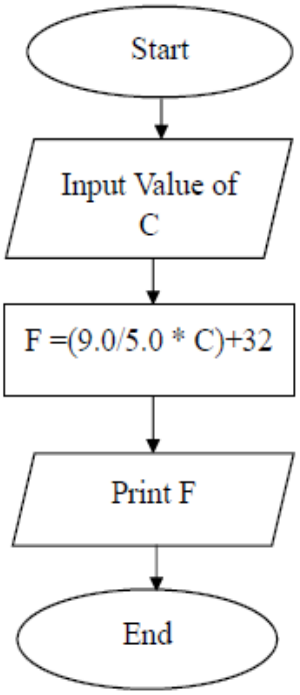
Step1: Start

Step2: Input temperature in Celsius C

Step3: F = (9.0/5.0 * C) + 32

Step4: Display Temperature in Fahrenheit F

Stop5: End



Example 4: Write an algorithm and flowchart to determine if the integer number entered from the keyboard is even or odd.

Algorithm

Step1: Start

Step2: Input the number (n)

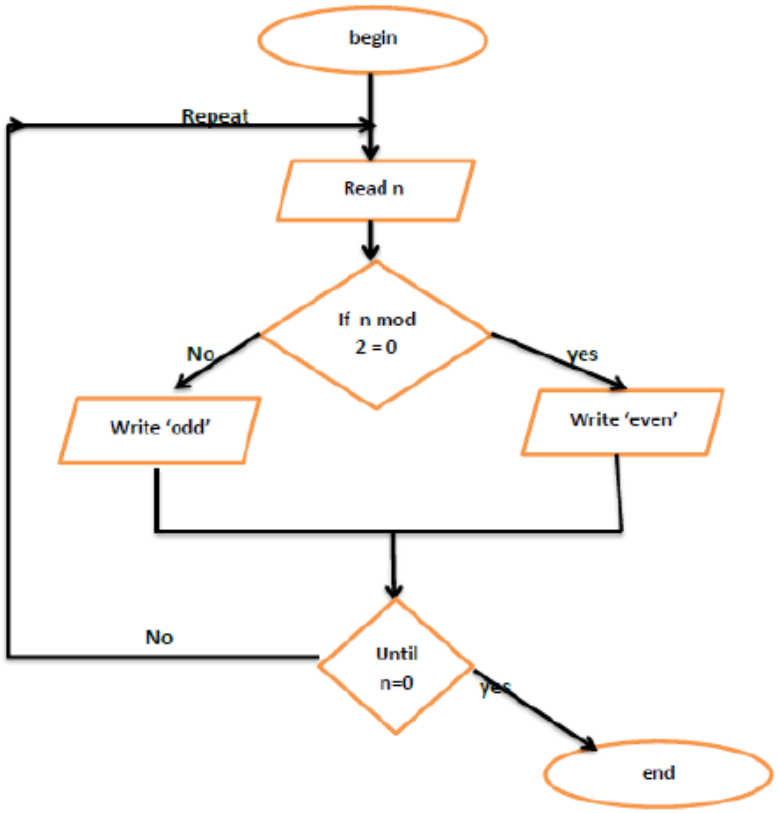
Step3: If the number (n) mod 2 = 0

Step4: yes-----Write (even)

Step5: No-----Write (Odd)

Step6: Until the number (n) = 0

Step7: End



HW1: Write an algorithm and flowchart to print the smallest number between two integers.

HW2: Write an algorithm and flowchart to print the Student Mark. If it is greater than 60, it prints Filling; if it is less, it prints Successful.

Q/ What are the main stages to follow when solving a general problem?

ماهي المراحل الرئيسية التي يجب اتباعها عند محاولة حل مشكلة عامة؟

Step 1: Define the problem.

Step 2: analyze the problem.

Step 3: Propose and evaluate possible solutions.

Step 4: Select and justify the optimal solutions.

Step 5: Implementation and review.

Structured programming modular programming

Structured programming (sometimes known as modular programming) is a subset of procedural programming that enforces a logical structure of the written program to make it more efficient and easier to understand and modify.

البرمجة المنظمة البرمجة المعيارية

البرمجة المنظمة (المعروفة أحيانًا بالبرمجة المعيارية) هي مجموعة فرعية من البرمجة الإجرائية التي تفرض بنية منطقية على البرنامج المكتوب لجعله أكثر كفاءة وأسهل في الفهم والتعديل.

Modular programming is a software design technique that separates a program's functionality into independent, interchangeable modules. This approach allows for easier maintenance, debugging, and collaboration, as each module can be developed and tested independently.

البرمجة المعيارية هي تقنية تصميم برمجيات تتضمن فصل وظائف البرنامج إلى وحدات مستقلة قابلة للتبديل. يسمح هذا النهج بصيانة أسهل واستكشاف الأخطاء وإصلاحها والتعاون، حيث يمكن تطوير كل وحدة واختبارها بشكل مستقل.

Top-Down Program Design

Top-down program design is an approach to program design that starts with the general concept and repeatedly breaks it down into parts. In other words, it begins with the abstract and continually subdivides it until it reaches the specific.

تصميم البرمجة من أعلى إلى أسفل هو نهج لتصميم البرامج يبدأ بالمفهوم العام ويقسمه مرارًا وتكرارًا إلى أجزاء. بعبارة أخرى، يبدأ بالمفهوم المجرد ويقسمه باستمرار حتى يصل إلى المفهوم المحدد.

Consider creating the prime factorization of a number like 1540. The steps involved might look like:

- 1540
- 2×770
- $2 \times 2 \times 385$
- $2 \times 2 \times 5 \times 77$
- $2 \times 2 \times 5 \times 7 \times 11$

Top-down programming design works the same way. We start with the overall objective and wind up with a series of steps needed to accomplish it.

يعمل تصميم البرنامج من الأعلى إلى الأسفل بنفس الطريقة. نبدأ بالهدف العام وننتهي بسلسلة من الخطوات اللازمة لإنجازه.

Bottom-Up Program Design

The bottom-up program design works oppositely. It starts with the parts and repeatedly combines them to achieve the general concept. In other words, it begins with the specific and continually combines it until it reaches the abstract. For example, consider the factorization from the previous section. For the bottom-up design, the steps involved might look like:

يعمل تصميم البرنامج من الأسفل إلى الأعلى على العكس من ذلك. فهو يبدأ بالأجزاء ويجمع بينها بشكل متكرر لتحقيق المفهوم العام. بعبارة أخرى، يبدأ بالجزء المحدد ويجمعه باستمرار حتى يصل إلى المجرد. على سبيل المثال، ضع في اعتبارك التحليل إلى عوامل من القسم السابق. بالنسبة للتصميم من الأسفل إلى الأعلى، قد تبدو الخطوات المتضمنة على النحو التالي:

- $2 \times 2 \times 5 \times 7 \times 11$
- $2 \times 2 \times 5 \times 77$
- $2 \times 2 \times 385$
- 2×770
- 1540

The top-down approach is mainly used by Structured programming languages like C, Pascal, Fortran, COBOL, etc.

بناء البرنامج

```
Program   ABC ;

Begin

End.
```

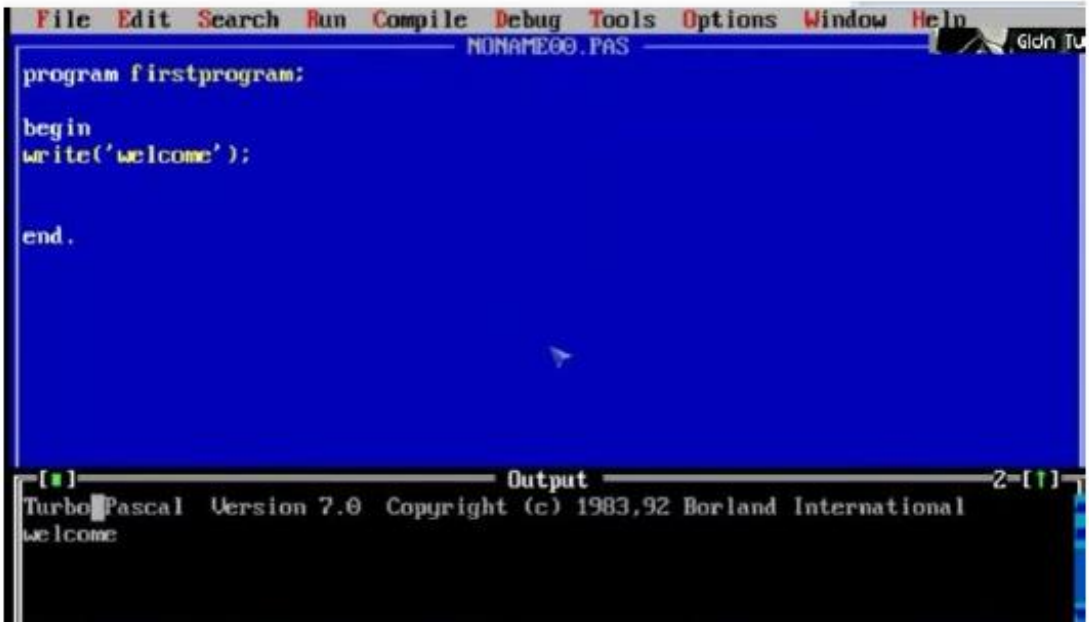
** بعد كلمة program يكون اسم البرنامج (يجب ان يبدأ اسم البرنامج بحرف و لا يكون هناك فاصل بين الاسم)

** بعد كلمة begin يبدأ بناء البرنامج من ايعازات و اوامر

** بعد الانتهاء من البرنامج نكتب ايعاز ال . end و توضع ال (.) بعد ال end الاخيره بمعنى ان البرنامج انتهى

مثال :-

```
Program firstprogram ;

Begin

Write ( ' welcome ') ;

End .
```



ايعازات write و writeln

Write: يقوم بطباعة الجملة داخل نفس السطر

Writeln: يقوم بطباعة الجملة والإنتقال بعدها إلى السطر الذي يليه

ايعازات read و readln

Read: يقوم بقراءة البيانات داخل نفس السطر

Readln : يقوم بقراءة البيانات من المستخدم وبعدها ينتقل المؤشر للسطر التالي

ملاحظات

- عند كتابة اي برنامج يجب تعريف المتغيرات و الثوابت الموجوده داخل ابرنامج
- المتغيرات (variables) عباره عن اسماء تخزّن في ذاكرة الحاسوب . هذا الاسم يستخدم لتخزين المعلومات في الذاكره .يمكننا استخدام انواع مختلفه من المعلومات في المتغيرات , مثلاً الارقام و السلال و غيره
- الثوابت (constants)هي القيمة الثابته التي لا يمكن تغييرها مثل (مساحة الدائره , النسبه الثابته ... الخ)

يجب تعريف المتغيرات و الثوابت قبل البدء ببناء البرنامج الخاص بنا اي قبل begin

المتغيرات تكتب var
الثوابت تكتب const

انواع المتغيرات

Name	Type	Exampel
String	**سلسل احرف**	'zahraa','reem'
integer	**اعداد صحيحة**	3,4,6
Real	**اعداد حقيقية**	3.14,503.2
Boolean	**قيمة منطقية**	True,False
Character	**احرف**	'A','E'

مثال :- اكتب برنامج لجمع عددين

```
Program summation ;
Var
Fn , sn , sum : integer ;
Begin
Writeln (‘enter the first number ’);
Readln (fn);
Writeln ( ‘ enter the second number ’) ;
Readln (sn) ;
Sum:= fn + sn ;
Writeln ( ‘ sum=’ , sum) ;
Readln
End.
```

ملاحظه : في نهاية البرنامج قمنا بكتابه readln حتى تثبت شاشه التنفيذ و لا تختفي مباشرة بعد التنفيذ

```
Program example;
Var
X: integer;
S: string;
Begin
Write ('Enter your Name: ');
Readln(s);
Write ('Enter your Age: ');
Readln(x);
Writeln ('Your Name is: ‘, s,' your age is: ‘, x);
End.
```

HW/What are the outcomes of this program?

Q/ Write a program in Pascal to calculate the area of the circle.

```
program CircleArea;
const
Pi = 3.14159;
var
radius, area: real;
begin
      clrscr;
      writeln ('Program to calculate the area of a circle');
      write ('Enter the radius of the circle: ')
      readln(radius);
      area: = Pi * radius * radius;
     writeln ('The area of the circle is ', area);
      readln; {Pause the program to view the result before exiting}
end.
```

HW/ Write an algorithm and flowchart for the above program.

Explanation:

Pi is a constant used to approximate the value of Pi (π).

The user is prompted to input the radius of the circle.

The area is calculated using the formula Pi * radius^2.

condition statement in pascal

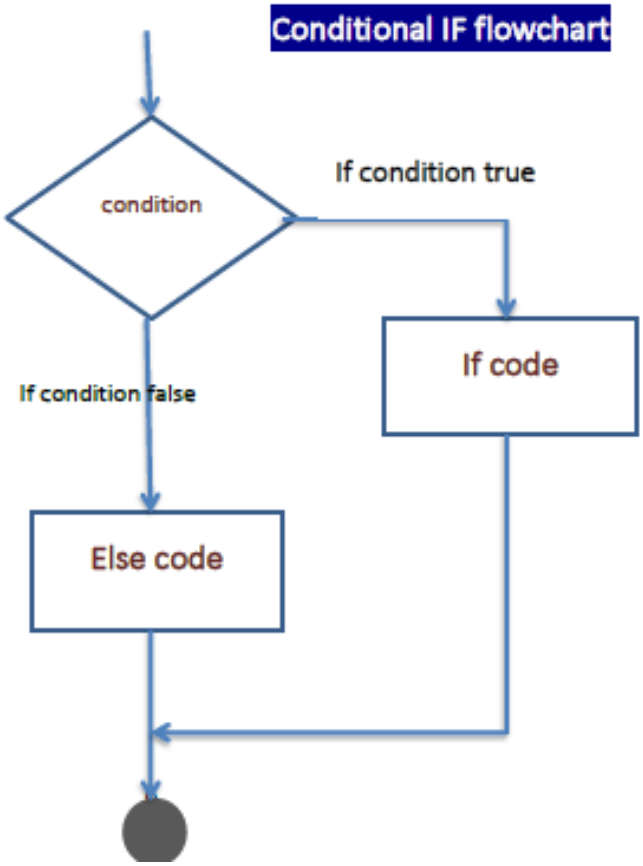
العبارات الشرطية في لغة باسكال

الجملة الشرطيه : في البرمجة عبارة الجمله الشرطيه تعني اتخاذ القرار , في حال تحقق الشرط يكمل البرنامج و في حال لم يتحقق يذهب الى جملة عدم التحقق

القرارات التي يمكن اتخاذها مع الجملة الشرطيه if :-

>	**اكبر من**
<	**اصغر من**
>=	**اكبر من او يساوي**
<=	**اصغر من او يساوي**
=	**يساوي**
<>	**لا يساوي**

ممكن رسم ال if الشرطيه بالمخطط التالي



في لغه باسكال توجد جملتين للشرط:

- **If**
- **Case**

If-then-else

An **if-then** statement can be followed by an optional **else** statement, which executes when the Boolean expression is **false**.

Syntax

The syntax for the if-then-else statement is −

```
if condition then S1 else S2;
```

For example,

```
if color = red then
  writeln('You have chosen a red car')

else
  writeln('Please choose a color for your car');
```

If the boolean expression **condition** evaluates to true, then the if-then block of code will be executed, otherwise, the else block of code will be executed.

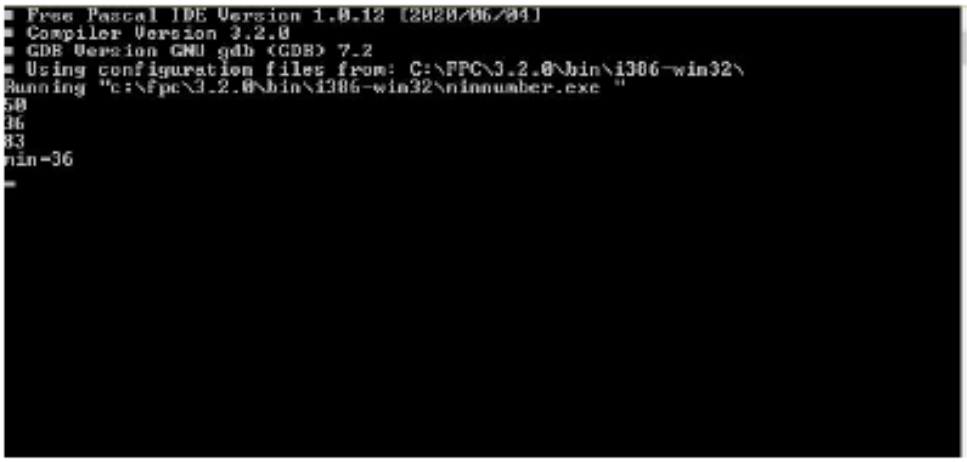
إذا تم تقييم شرط التعبير المنطقي على أنه صحيح، فسيتم تنفيذ كتلة التعليمات البرمجية if-then ، وإلا فسيتم تنفيذ كتلة التعليمات البرمجية else.

```
Program big;
Var
X: integer;
Begin
Read(x);
```

```
If x>10 then writeln (' bigger than 10 ')
Else
Writeln (' less than 10 ');
End.
```

مثال :- اكتب برنامج يقرأ ثلاث اعداد و يجد الاصغر بينهم باستخدام IF

```
Program minnumber ;
Var
a,b,c,min : integer ;
begin
readln (a,b,c) ;
if (a<b) and (a<c) then
min:=a
else
if (b<a) and (b<c) then
min:=b
else
min:=c ;
writeln ('min=' , min) ;
readln ;
end.
```



شاشة التنفيذ

ملاحظات:-

- لا نستخدم الفارزه المنقوطه داخل ال IF الشرطيه الا بعد الانتهاء منها (أي بعد اخر else نستخدمها)
- للربط داخل جملة if نستخدم (and) او (or)

and = ان تكون الحالتين صحيحه

Or= يجب ان تكون حاله واحده او الحالتين صحيحه

** التعابير الرياضيه Arithmetic Expression

نوع العمليه	عملها	نوع العدد
- * +	الطرح – الضرب ـ الجمع	تقبل الاعداد الحقيقيه و الصحيحه
/	القسمة	اعداد حقيقية فقط
div	ناتج القسمة (ما قبل الفاصله)	اعداد صحيحه
mod	الباقي من القسمه (ما بعدالفاصله)	اعداد صحيحه

مثال :ـ اكتب برنامج يختبر الارقام في حال كانت فرديه او زوجية

```
Program  digit;
Var
X: integer;
Begin
Write ('x=');
Readln (x);
If x mod 2=0 then
Writeln ('even')
Else
Writeln ('odd');
Readln
End.
```

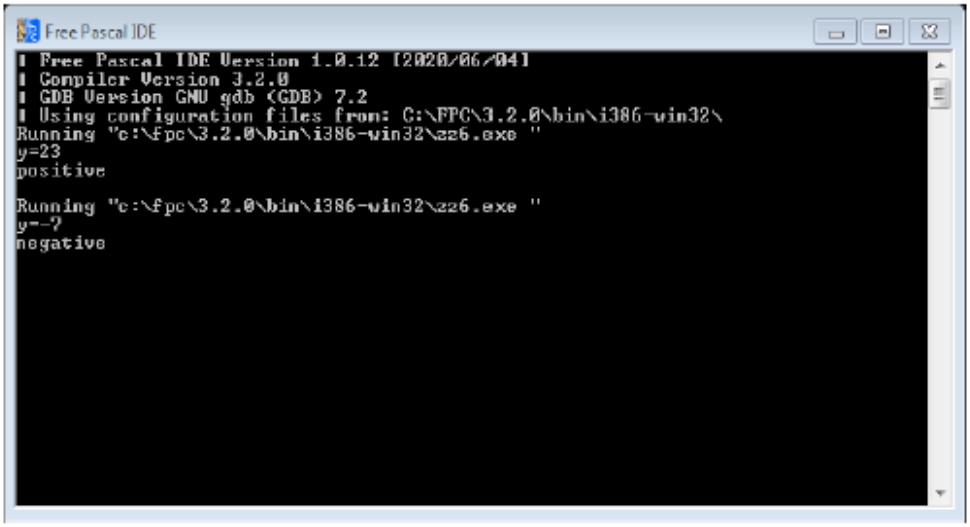
```
Free Pascal IDE Version 1.0.12 [2020/06/04]
Compiler Version 3.2.0
GDB Version GNU gdb (GDB) 7.2
Using configuration files from: C:\FPC\3.2.0\bin\i386-win32\
Running "c:\fpc\3.2.0\bin\i386-win32\digit.exe "
x=6
even

Running "c:\fpc\3.2.0\bin\i386-win32\digit.exe "
x=9
odd
```

شاشة التنفيذ

مثال\\ اكتب برنامج يختبر العدد اذا كان موجب يطبع (positive) واذا كان سالب يطبع (negative).

```
Program number;
Var
y:integer;
Begin
Write(' y= ');
Readln(y);
If(y>0) then
Writeln('positive')
Else
Writeln('negative');
Readln
End.
```

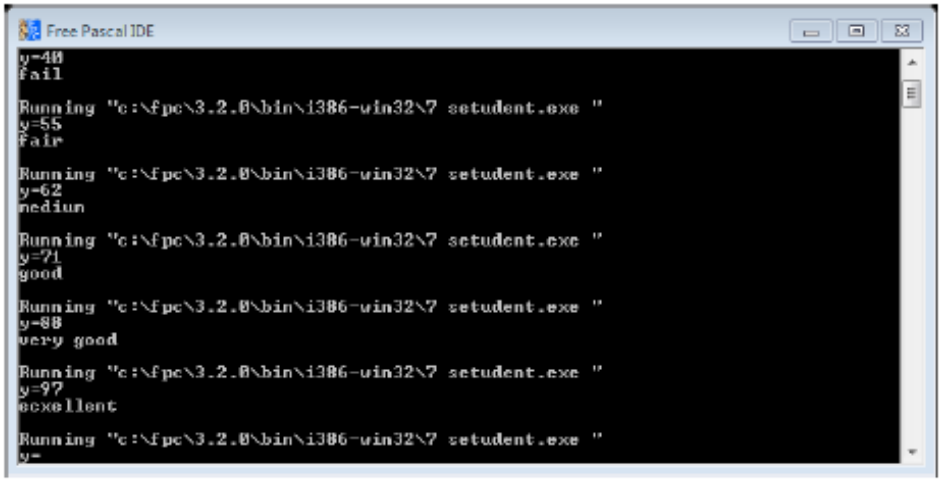
شاشة التنفيذ



س / اكتب برنامج يحسب تقدير درجة الطالب

```
program student;
var
y: integer ;
begin
write('y=');
readln(y);
if y < 50 then
writeln ('fail')
else
if (y>=50) and (y<60) then
writeln ('fair')
else
if (y>=60)and (y<70) then
writeln ('medium')
else
if (y>=70) and (y<80) then
writeln ('good')
else
if (y>=80) and (y<90) then
writeln ('very good')
else
if (y>=90) and (y<=100) then
writeln ('ecxellent');
readln;
end.
```

شاشة التنفيذ



مثال : اكتب برنامج لطباعة مساحة و محيط المستطيل

```
program rectangle ;
var
x ,y , area , per :integer;
begin
write('x=');
readln(x);
write('y=');
readln(y);
area:=x*y;
writeln('area=',area);
per:=(x+y)*2;
writeln('per=',per);
readln;
end.
```

شاشة التنفيذ

استخدام case of

في الكثيرمن الاحيان تحتاج الى تكرار العديد من الشروط في برنامج واحد
و ذلك يعني العديد من جمل if و لكن في هذه الحاله سيكون البرنامج طويل و معقد
و من هنا جائت عبارة case كبديل لجملة if

الشكل العام لجملة ال case هو

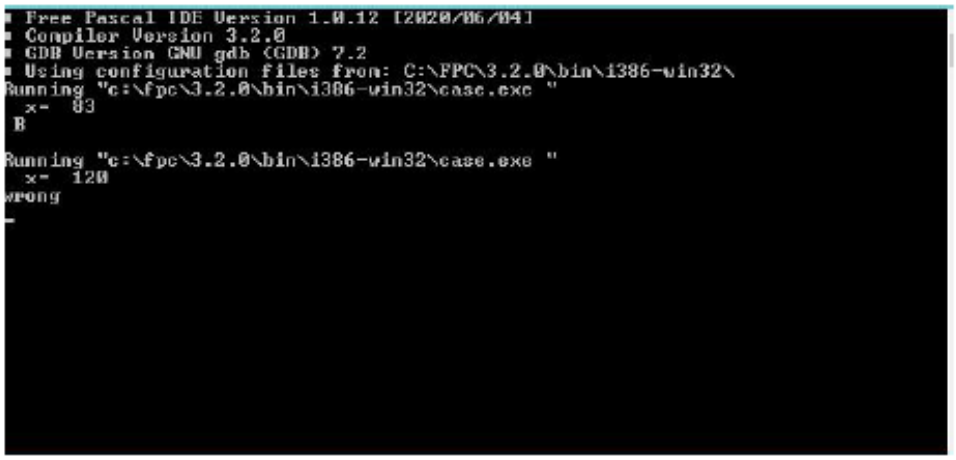
```
Case variable of
.
.
End;
```

- نلاحظ ان جملة case لا تحتوي على begin و لكن تحتوي على end
- يتم تحديد المدى بكتابة العنصر الاول و الاخير بينهما نقطتان فقط (..)
- لكتابة عناصر مختلفه يكون الفصل بين العناصر بالفارزه (,)

مثال :- حل برنامج تقدير درجات الطلاب بواسطة case of

```
Program text;
Var
X:integer;
Begin
Write (‘x=’);
Read(x);
Case x of
90..100 : writeln('A');
80..89 : writeln('B');
70..79 : writeln('C');
60..69 : writeln('D');
50..59 : writeln('E');
0..49 : writeln('F');
Else
Writeln (‘wrong’);
End;
readln
End.
```

شاشة التنفيذ

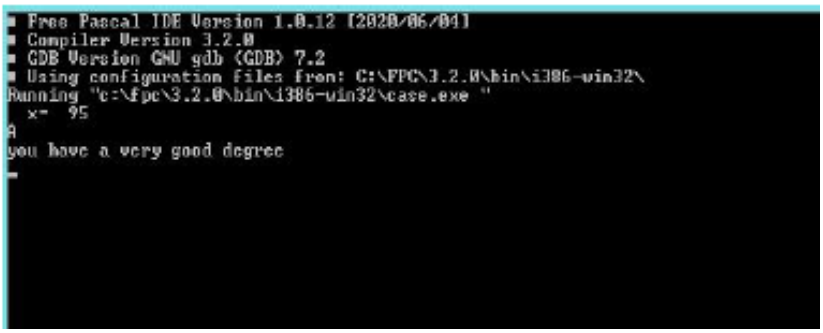


ملاحظة:- في حال يوجد اكثر من امر في الشرط الاول اي الدرجه ما بين 90 و 100 و يكون هنالك جمليتين للطباعه في شاشه العرض , بهذه الحاله نحتاج الى عبارة begin نكتب بداخلها الاوامر المراد ادخالها ثم انهائها بعبارة ;end

مثال

```
Program text;
Var
X:integer;
Begin
Write ('x=');
Read(x);
Case x of
90..100:
   Begin
      Writeln('A');
      Writeln(' you have a very good degree ');
    End;
80..89 : writeln('B');
70..79 : writeln('C');
60..69 : writeln('D');
50..59 : writeln('E');
0..49 :
   Begin
     Writeln('F');
     Writeln('you have to Re do the exam');
   End;
Else
Writeln('wrong');
End;
Readln
End.
```

شاشة التنفيذ



مثال :ـ اكتب برنامج يطبع ايام الاسبوع كمخرجات للارقام من 1 الى 7 و بعكسه يطبع خطأ باستخدام case .. of

```
program week;
var
n: integer;
begin
write('day number= ');
readln (n);
case n of
1: writeln ('saturday');
2:writeln ('sunday');
3:writeln ('monday');
4:writeln('tusday');
5:writeln ('wednsday');
6: writeln ('thursday');
7: writeln ('friday');
else
writeln ('error');
end;
readln
end.
```

شاشة التنفيذ

Q/ Write a program to do a welcome message to 5 names , and if you enter a different name write (you are not on the list) using Case ..Of

```
Program name;
Var
X:string;
Begin
Writeln('type your name');
Read(x);
Case x of
'zahraa'        :writeln('welcome MS.ZAHRAA');
'ali'           :writeln('welcome MR.ALI');
' mohammed' :writeln('welcom MR.MOHAMMED');
'osama'         :writeln('welcome MR.OSAMA');
'houda'         :writeln('welcome MS.HOUDA');

Else
Writeln(' you are not on the list');
End;
Readln
End.
```

شاشة التنفيذ