# الفصل الثامن

## مدخل الى
## البرمجة بلغة بيسك



obeikandl.com

obeikandl.com

## 1- تقديم :

لقد قلنا في الفصل السابق بأن **البرمجة** هي عملية وضع **الخوارزمي** بلغة معينه . ويدعى الخوارزمي المكتوب باحدى لغات البرمجة **بالبرنامج** . ويفترض ان يكون كل **سطر** من **سطور** هذا البرنامج مفهوم من قبل الحاسبة. وكذلك يجب ان تكون الحاسبة قادرة على تنفيذ كل **سطر** في **البرنامج** . وهناك كما قلنا سابقا عدة لغات للبرمجة ، الا اننا سوف نقوم بدراسة لغة بيسك ضمن هذا الفصل.

تتكون لغات البرمجة بصورة عامه من مجموعة معينه ومحددة من **التعابير** و**الاوامر** و**الوظائف** . و**الاوامر** عبارة عن مجموعة من الرموز والحروف التي تشكل مفردات مفهومة من قبل الانسان حيث تكون قريبة جدا لمفردات اللغة الاعتيادية . وعندما نقوم بتغذية الحاسبة بأحد هذه **الأوامر** فإن الحاسبة تقوم على الفور بتنفيذ عملية معينة . و **الاوامر** تكون مبرمجة اصلاً من قبل الشركة المصنعة للحاسبة . ومن هذه الاوامر مثلاً **تعبير (نفّذ RUN)** ، فهذا الايعاز (أو الأمر) يجعل الحاسبة تنفذ البرنامج الذي قمنا بتغذيته لها ، أي الموجود في الذاكرة في حينها . أما **التعابير** و**الوظائف** فسوف نجدها تفصيلياً في الفقرات التالية .

## 2- السطر :

يتألف البرنامج بلغة بيسك من مجموعة من السطور ، يبدأ كل **سطر** منها برقم يدعى **رقم السطر** . وهو عبارة عن **عدد صحيح موجب** (أكبر من الصفر) يبدأ به كل سطر من البرنامج ، ويمكن ان يكون على شكل عدد مكون من رقم واحد أو أكثر ولغاية خمسة ارقام . ولقد جرت العادة على ترقيم سطر البرنامج بلغة بيسك بمضاعفات العشرة ، أي 10,20,30, ...الى آخره . ولا يفوتنا هنا أن نذكر ان هذا الترقيم له فائدة مهمة ، وهي انه عندما يريد المبرمج اضافة (أو حَشْر) سطر ، أو أكثر (ولغاية تسع سطور) ، بين سطرين في البرنامج الاصلي فأنه

سيتمكن من ذلك إذا كانت السطور مرقمة بطريقة مضاعفات العشرة. فلو اردنا مثلاً اضافة سطر بين السطرين المرقمين (10) ، (20) على التوالي ، فأننا يمكن أن نضيف في أي مكان من البرنامج سطور مثل 19,...,14,13,12,11 . حيث تقوم الحاسبة بوضعها بين السطرين (10) و (20) حتى ولو قمنا بإدخالها بعد السطر الاخير في البرنامج (سطر رقم 9980) مثلاً. وذلك لأن الحاسبة تقوم بخزن السطر حسب تسلسل **رقم السطر** . ويبدأ تنفيذ البرنامج بالتسلسل ابتداءً من السطر ذو الرقم الاقل ولغاية السطر ذو الرقم الاعلى في البرنامج .

فلو قمنا بإدخال برنامج مؤلف من أربعة سطور ارقامها هي 4,8,15,2 حسب تسلسل ادخالها الى الحاسبة . فأن الحاسبة سوف تقوم بخزنها وتنفيذها بحيث يكون رقم السطر تصاعديا أي 15,8,4,2 . لذا يجب علينا توخي الدقة والحذر عند كتابة سطور البرنامج وارقامها كي لا يحصل خطأ في تسلسل سطور البرنامج بشكل يجعل النتائج مختلفة عن مضمون الخوارزمي الذي نقصده .

يمكننا إجمالا ان نضع **صيغة عامه للسطر بلغة بيسك** وهي:

**رقم السطر  نوع التعبير  القيم**

وقد شرحنا اعلاه **رقم السطر** ، وكذلك شرحنا **القيم** تفصيليا في الفقرات (9,8) من الفصل السابع . أما نوع التعبير فيمكن ان يكون **إدخال/اخراج** ، **حساب** ، **منطق** ، **اختيار** (أو انتخاب) (للسيطرة على مسار تنفيذ البرنامج ، انظر الفقرة (2-2) - الفصل الأول) ، **تكرار**، **خزن / استرجاع** ، **تحريك البيانات** . وسوف نتطرق الى أهم هذه الانواع من التعابير في الفقرات اللاحقة ، والتي تفيدنا في كتابة برامج بسيطة . كما اننا سنشرح المزيد من التعبير والاوامر والوظائف . في الفصل التاسع لغرض الاستزادة واتمام الفائدة .

## 3- تعبير (الملاحظه REM) :

إذا كان التعبير (في سطر برنامج البيسك) هو كلمة **REM** فهذا يعني ان هذا السطر في البرنامج وهمي بالنسبة للحاسبة . أي ان كل ما يعقب تعبير **(الملاحظة REM)** لن تفهمه الحاسبة ولا تنفذ ما فيه . وبعبارة أخرى فأن تعبير **(الملاحظة REM)** يفيد المبرمج أو القاريء وذلك بأن يضع ملاحظاته بعد كلمة **REM** ، كي يتذكر ما هو المقصود ب**السطر** ( أوالسطور) اللاحق في البرنامج ، أو ما هي معاني بعض الرموز المستخدمة في البرنامج . كما نرى ذلك في الامثلة التالية .

| البرنامج | الترجمة بالعربية |
|---|---|
| 10 REM A = LENGTH, B = WIDTH | 10 ملاحظة A = الطول ، B = العرض |
| 106 REM now substitute the values in Equation | 106 ملاحظة عوض الان القيم في المعادلة |
| 92 REM DON'T FORGET X'S VALUE | 92 ملاحظة لاتنسى قيمة X |

ومن الامثلة اعلاه نلاحظ ان ما يعقب كلمة (**REM**) لا يعدو كونه ملاحظات يستفاد منها المبرمج (أو القارىء) بشكل أساسي ، ولا تتمكن الحاسبة من تنفيذه .

## 4- تعابير الادخال/الاخراج :

وتستخدم لإدخال البيانات والمعلومات الى الحاسبة ، أو التي نحصل بواسطتها على النتائج (أو ما شابه) بشكل مرئي (على شاشة تلفزيونية (مونيتور)) أو مطبوعة على الورق أو مرسومة أو مسموعة .. الى آخره من وسائل الادخال والاخراج في الحاسبة . وهي على أنواع عديدة منها :

**1-4- التعبير المباشر للادخال :**

حيث يمكننا ادخال البيانات بشكل مباشر الى الحاسبة ، بأن نكتب الرمز الحرفي للثابت أو المتغير ويليه علامة المساواة (=) ومن ثم القيمة التي تكافئه . كما موضح في الامثلة الثلاثة التالية :

```
521  X=10.5
22  Y%=80
1111 Z$="زيد بن عمرو"
```

**2-4- تعبير (دع LET) :**

نستخدم هذا التعبير لإدخال قيم المتغيرات ، فلو اردنا مثلاً ان نجعل القيمة العددية للمتغير (z) تساوي (10) فيمكننا كتابة ذلك بالشكل التالي:

```
135 LET  Z = 10.
```

حيث تقوم الحاسبة بتعويض قيمة (z) ، اينما وجدت في البرنامج بعد هذا السطر ، بما يقابلها كقيمة عددية . ويمكن أن نغيّر قيمة (z) بعد ذلك بقيمة اخرى مثل :

```
141 LET  Z = 7.0
```

وبذلك أصبحت قيمة (z) الجديدة تساوي (7) ، وتقوم الحاسبة بتعويض (z) بقيمتها العددية الجديدة اينما وجدت بعد ذلك في البرنامج. وأُلغيت قيمتها القديمة في ذاكرة الحاسبة . ونلاحظ مما تقدم ان هذا التعبير يشبه الى حد كبير تعبير الادخال المباشر ،الذي شرحناه في الفقرة السابقة .

ولكي نتخيل ذلك بشكل مبسط ، سوف نفترض ان ذاكرة الحاسبة أشبه ما تكون بمجموعة كبيرة من الصناديق الصغيرة (مثل صناديق البريد الموجودة في دائرة البريد) وهناك لكل صندوق منها **عنوان** يوضع

على بابها ، وهو هنا يمثل **اسم المتغير** (أو رمزه) . ولو فتحنا باب هذا الصندوق فسنجد داخله **محتويات** تمثل القيمة الفعلية (سواءاً كانت عددية أو حرفية) لذلك المتغير . وعندما نقول للحاسبة ، مثلاً ، ان قيمة (10=z) كما في السطر رقم (135) اعلاه ، فذلك معناه اننا حجزنا موضع في الذاكرة (يشبه الصندوق) اسميناه (z) ، ووضعنا داخل هذا الموضع قيمة (z) والتي تساوي (10) .

واذا اردنا في سطر لاحق ان نزيد قيمة المتغير (z) . بمقدار معين (ولنفترضه واحد مثلاً) فأننا لا نفعل شيئاً سوى ان نضع قيمة المتغير الجديدة تساوي قيمة المتغير القديمة مضافاً لها واحد مثل السطر التالي:

```
136 LET  Z= Z+1
```

وهذا يعني بكل بساطة إن الحاسبة سوف تقوم بالتفتيش عن موضع **الذاكرة المعنون** (z) وتجلب محتوياته (أو قيمة المتغير (z) العددية) وتضيف لها واحد وتضع الناتج من جديد في نفس الموضع والمعنون (z) وبذلك سوف تكون قيمة (z) الجديدة هي (10+1) أي (11). كما جاء في السطرين (135) ، (136) ، هذا على فرض انهما موجودان في نفس البرنامج . على ان نتذكر ان القيمة (10) للمتغير ( z) سوف تُمحى بشكل نهائي من ذاكرة الحاسبة . وهذه الحالة مهمة جدا ، حيث سنقوم مراراً بالاستفادة منها في تنفيذ البرنامج وسنسميها **(العدادات)** .و نستطيع بواسطتها السيطرة على تنفيذ البرنامج ، وكذلك في إكساب الخوارزميات (أي البرامج) مرونة أعلى في حل المسائل . وهذا ما سنراه تفصيليا في الامثله والفقرات اللاحقة .

### 3-4- تعبير (إدخال INPUT) :

لقد درسنا في الفقرتين السابقتين نوعين من تعابير الادخال المباشر .

ويمكننا ان نقوم بادخال البيانات الى الحاسبة بتعبير آخر هو (**إدخال INPUT**) ، وتكون **الصيغة العامه** لهذا التعبير هي:

n **INPUT** *a,b,c,...,z*

حيث يمثل ( *n* ) رقم السطر كما اسلفنا . وتمثل الرموز *z,...,c,b,a* المتغيرات التي سنقوم بإدخال قيمها وهي قد تكون رموز لمتغيرات عددية (سواءاً حقيقية أو عشرية) أو لمتغيرات حرفية (من نوع السلسلة) .

**4-4- تعبير الإخراج (اطبع PRINT) :**

وهذا التعبير معناه اننا نطلب من الحاسبة ان تقوم بطبع (أي إخراج) النتائج على الشاشة التلفزيونية (مونيتور). فلو اردنا مثلاً من الحاسبة ان تطبع قيمة (Z) على الشاشة ، فما علينا إلا ان نكتب السطر التالي :

170 **PRINT** Z

وحالما تنفذ الحاسبة هذا السطر من البرنامج ، فأنها سوف **تطبع**

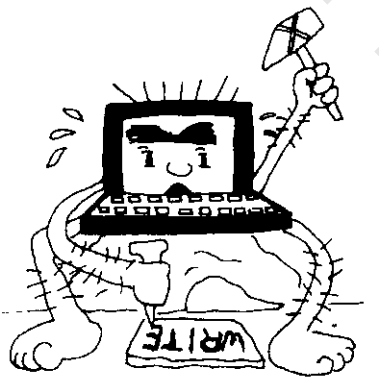
(**على الشاشة**) آخر قيمة للمتغير (Z) . ولو اردنا طباعة قيم أكثر من متغير في نفس التعبير ، مثل R,Z,Y,X فيمكننا ان نفعل ذلك كما في السطر التالي :

14 **PRINT** X,Y,Z,R

وعند تنفيذ هذا السطر تقوم الحاسبة بطبع قيم المتغيرات المطلوبة وتطبع كل منها ضمن حيز معين عبارة عن عدد محدد من المواضع المحجوزة سلفا لقيمة كل متغير ويتم حجز هذه المواضع بواسطة **الفارزة**. ولو قمنا باستبدال الفوارز الموجودة في السطر (14) **بفوارز منقوطة** ، عند ذاك سوف تقوم الحاسبة بطبع النتائج (قيم المتغيرات) الواحدة تلو الاخرى مباشرة ، أي بدون حجز عدد محدد من المواضع . وهذا يوفر مجالاً أوسع ضمن كل سطر لملئه قدر الامكان بالنتائج المطبوعة . ويمكن كتابة الصيغة لهذا التعبير بالشكل التالي :

***n*** **PRINT** ***z,...,c,b,a***

حيث يمثل ( ***n***) رقم السطر ضمن البرنامج ، أما الرموز ***(z,...,c,b,a*** ) فتمثل المتغيرات المطلوب طباعة قيمها على الشاشة .



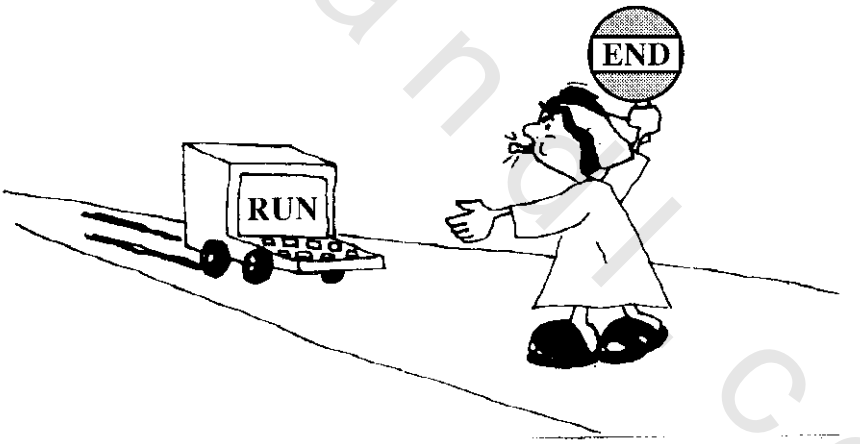

**WRITE**

**5- تعبير (نهاية END) :**

هذا التعبير يستعمل كسطر أخير لإنهاء البرنامج الرئيسي الذي ستقوم الحاسبة بتنفيذه علماً بأنه ليس بالضرورة ان يكون هذا التعبير كسطر أخير لكل البرنامج . فربما يأتي بعد السطر الذي يحوي هذا التعبير سطور أو أجزاء أخرى من البرنامج ، ولكن مثل هذه الاجزاء لا تعتبر ضمن البرنامج الرئيسي . مثل سطور تمثل البرامج الثانوية الملحقة بالبرنامج الرئيسي أو قيم البيانات الداخلة الى الحاسبة أو ما شابه مما سوف نشرحه لاحقا .

## 6- الأمر (نَفِّذْ RUN) :

هذا الأمر تتم طباعته بشكل مباشر الى الحاسبة ، أي بدون الحاجة الى ترقيمه برقم السطر ، كما لاحظنا في حديثنا عن التعابير الواردة في الفقرات السابقة اعلاه . ولكون (**نَفِّذْ RUN**) هي أمر ، فإن الحاسبة سوف تقوم بتنفيذ البرنامج الموجود في ذاكرتها فور طبعه وادخاله .




## 7- البرنامج :

كما تعلمنا من دراستنا للخوارزمي (الفصل الأول) فأن **البرنامج** هو عبارة عن سلسلة من الخطوات المكتوبة بلغة مفهومة من قبل الحاسبة، والتي تؤدي في حالة تنفيذها الى حل المسألة المطلوبة . كما

انه يفترض ان الحاسبة (التي يتم تنفيذ البرنامج فيها) تكون قادرة على تنفيذ كل خطوة (أو **سطر**) من خطوات **البرنامج** وضمن حدود امكانتها المحددة من قبل الشركة المصنعة .

من مجموعة التعابير التي تعلمناها في الفقرات السابقة اعلاه ، سوف نقوم بمحاولة لعمل مجموعة من البرامج البسيطة . وسوف نقوم اولاً بكتابة **الخوارزمي** اللازم لحل المسألة على شكل خطوات باللغة العربية ثم نقوم بترجمتها الى **مخطط إنسيابي** وبعد ذلك نكتب **البرنامج بلغة بيسك** .بعد ذلك سوف نلغي تدريجياً كتابة الخوارزمي وسنكتفي بالمخطط الانسيابي والبرنامج فقط .

## 8- أسئلة محلولة :

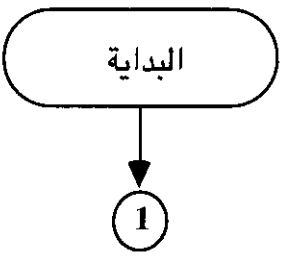
**سؤال** : أكتب برنامجا لإيجاد نتيجة حاصل ضرب العدد (X%) في نفسه وأرمز الى الناتج بالرمز (Y%) . علماً بأن قيمة (X%) تساوي (13) .

**الجواب :**

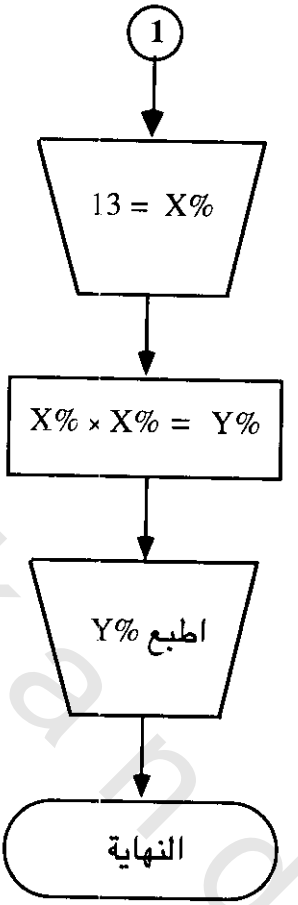
**الخوارزمي :**

1) X% = 13 .
2) Y% = X% x X% .
3) أطبع قيمة Y% .

**المخطط الانسيابي :**




obeikandl.com




**البرنامج :**

```
10  X%=13
20  Y%=X%*X%
30  PRINT Y%
40  END
```

**سؤال** : جد مساحة دائرة (A!) نصف قطرها (R%) يساوي (15) . ثم جد قيمة المحيط (C!) علماً بأن النسبة الثابتة (PI!) تساوي 22/7 .



obeikandl.com

**الجواب :**

**الخوارزمي :**

1) نصف قطر الدائرة (R%) = 15 .

2) النسبة الثابته 22∧7 .

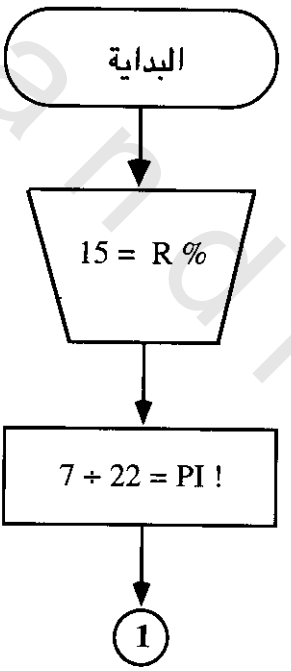
3) مساحة الدائرة (A!) = النسبة الثابته (PI!) x (نصف القطر (R%))$^2$ .

4 ) محيط الدائرة (C!) = 2 x النسبة الثابته (PI!) x نصف القطر (R%) .

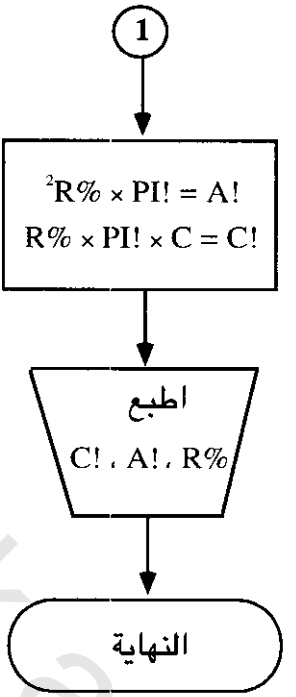
5) اطبع قيمة C!, A!, R% .

6) النهاية .

**المخطط الانسيابي :**




obeikandl.com




**البرنامج :**

```
10  R%=15
20  PI!=22./7.
30  A!=PI!*R%^ 2
40  C!=2*PI!*R%
50  PRINT R%, A!, C!
60  END
```

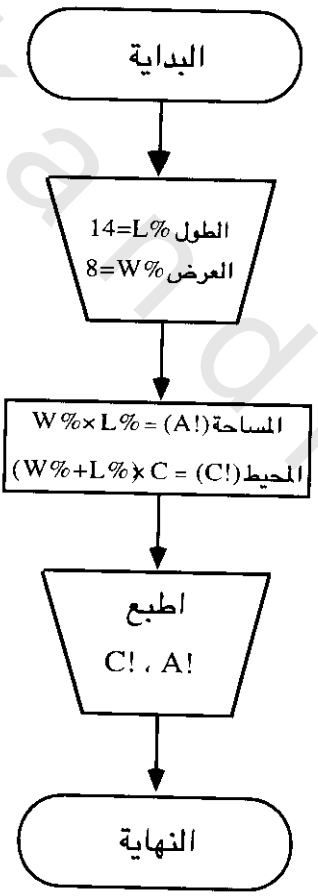
**سؤال** : اكتب الخوارزمي وارسم المخطط الانسيابي ثم اكتب البرنامج بلغة بيسك لإيجاد مساحة المستطيل (A!) ومحيطه (C!) ، اذا علمت بأن طول المستطيل (L%) يساوي (14) سم ، وعرضه (W%) يساوي (8) سم .

obeikandl.com

**الخوارزمي :**

1) طول المستطيل (L%) = 14 .
2) عرض المستطيل (W%) = 8 .
3) مساحة المستطيل (A!) = الطول (L%) x العرض (W%) .
4) محيط المستطيل (C!) = 2 x (الطول (L%) + العرض (W%) .
5) أطبع قيمة المساحة (A!) ، المحيط (C!) .
6) النهاية .

**المخطط الانسيابي :**






obeikandl.com

**البرنامج :**

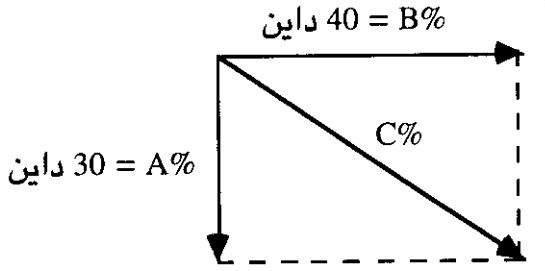
```
5   L%=14
10  W%=8
15  A!=L%*W%
20  C!=2*(L%+W%)
25  PRINT A!,C!
30  END
```

لقد لاحظنا من الأسئلة السابقة ان الخوارزمي لا يعدو ان يكون ترجمة حرفية للمخطط الانسيابي وللبرنامج . لذا ، ومن السؤال التالي فصاعداً سنكتفي برسم المخطط انسيابي، وبعد ذلك نكتب البرنامج بلغة بيسك ، لكونهما صورتين من صور الخوارزمي .

**سؤال** : أثرت في جسم القوتان (A%) ومقدارها (30) داين جنوباً ، و(B%) ومقدارها (40) داين شرقا . ما مقدار القوة المعادلة (المحصلة) لهاتين القوتين . إرسم المخطط الانسيابي وأكتب البرنامج اللازم لحل المسألة .

**الجواب :**

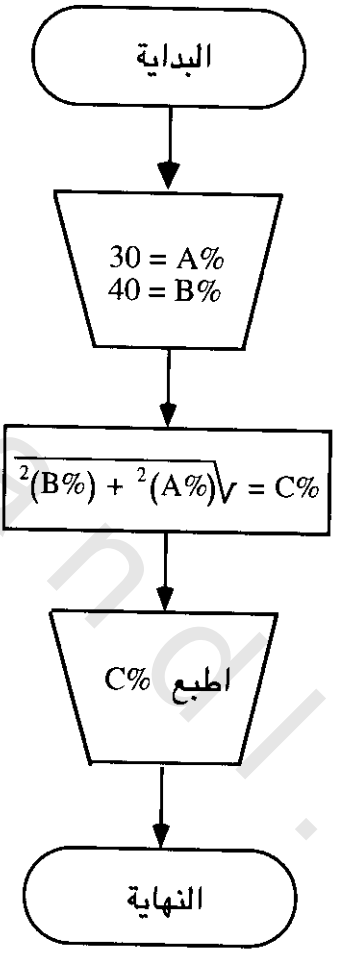
نرسم القوتين (A%) و (B%) فنلاحظ ان المحصلة (C%) تساوي وتر في مثلث قائم الزاوية وكما يلي:



وبتطبيق **نظرية فيثاغورس** نجد ان :

$$C\% = \sqrt{(A\%)^2 + (B\%)^2} = ((A\%)^2+(B\%)^2)^{0.5}$$

ويمكن رسم **المخطط الانسيابي** لهذه المسألة كالتالي :




obeikandl.com

**البرنامج :**

```
22  A%=30
24  B%=40
26  C!=(A%^2+B%^2)^0.5
28  PRINT C!
30  END
```

**سؤال :** جسم صلد على شكل متوازي المستطيلات قاعدته مربعه الشكل ، طول ضلع المربع (L1) يساوي (0.5) متر ، وارتفاع الجسم هو (L2) ويساوي (0.2) متر ، ووزن الجسم (FORCE) يساوي (60) نيوتن . ما مقدار الضغط الذي يسلطه على منضده يوضع فوقها اذا كان :

أ) مستنداً على قاعدته ؟

ب) مستنداً على أحد أوجهه الجانبية ؟

**الحل :**

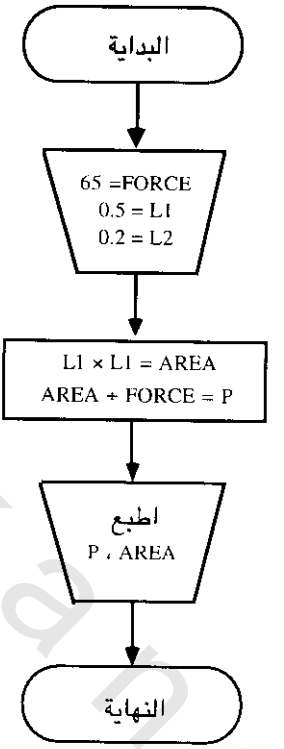
إن مقدار الضغط يساوي القوة المسلطة على السطح بصورة عمودية (الوزن) مقسومة على مساحة السطح أي أن :

$$\text{الضغط} = \frac{\text{القوة}}{\text{المساحة}}$$

فنرمز للضغط بالرمز (P) ، والمساحة بالرمز (AREA) ، فتصبح المعادلة السابقة كالتالي :

$$\frac{\text{FORCE}}{\text{AREA}} = \text{P}$$

**المخطط الانسيابي** للفرع (أ) من السؤال هو :




أما **البرنامج** فيمكن كتابته كما يلي :

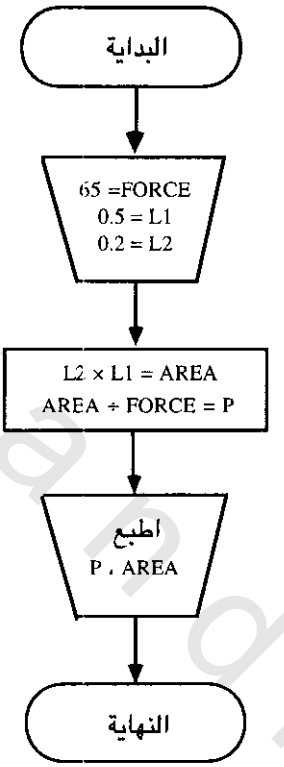
```
10  FORCE=65
20  L1=0.5
30  L2=0.2
40  AREA=L1*L1
50  P=FORCE/AREA
60  PRINT P,AREA
70  END
```



obeikandl.com

أما بالنسبة للفرع (ب) فنجد فيما يلي **المخطط الانسيابي والبرنامج** :

**المخطط الإنسيابي** :




أما البرنامج لهذا الفرع من السؤال فهو يشبه برنامج الفرع (أ) اعلاه عدا السطر رقم (40) حيث يتم تغييره ليصبح كالتالي :

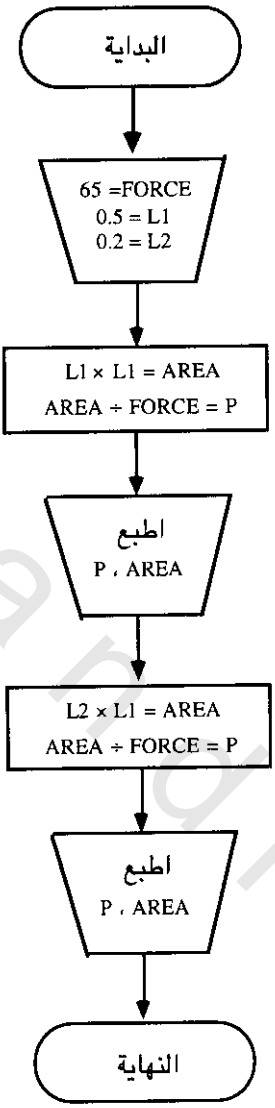
**40 AREA = L1*L2**

ويمكننا دمج الإجابة عن الفرعين معاً وذلك كما يلي :



obeikandl.com

## المخطط الانسيابي :




obeikandl.com

البرنامج :

```
10  FORCE=65
20  L1=0.5
30  L2=0.2
40  AREA=L1*L1
50  P=FORCE/AREA
60  PRINT P,AREA
70  AREA=L1*L2
80  P=FORCE/AREA
90  PRINT P,AREA
100 END
```

## 9- تعابير السيطرة :

من الأمثلة في الفقرة السابقة نلاحظ ان كافة البرامج المكتوبة هي نموذج مثالي **لخوارزمي التتابع** . حيث ان الحاسبة تقوم بتنفيذ البرنامج خطوة بعد خطوة . ولكن لا يمكننا أن نحل جميع المسائل بهذا النوع من **الخوارزمي** فقط . بل نحتاج بالتأكيد الى خوارزميات أخرى، ومنها مثلاً **خوارزمي الإنتخاب** (انظر الفقرة (2-2) الفصل الأول). ولابد لنا والحالة هذه أن نستعمل تعابير من نوع آخر غير ما أسلفناه ، وسوف ندعوها (**تعابير السيطرة**) وهي التي تفيدنا في تغيير مسار تنفيذ البرنامج والانتقال الى خطوات (أو سطور) أخرى لا تقع على التتابع . وقد يكون هذا الانتقال غير مشروط والذي يمكن ان نعبر عنه ب **التعبير (أذهب الى GOTO)** ، أو يكون الانتقال مشروطاً كما نجده في **التعبير (إذا ...آنذاك ... وإلا ...IF... THEN... ELSE)** أو قد يكون التعبير يفيدنا في السيطرة على تكرار تنفيذ جزء من البرنامج

عدد معين من المرات ، ومثال على ذلك **تعبير (الأجل FOR)** والذي يتبعه بعد عدد من السطور تعبير آخر هو **(التالي NEXT)** حيث يشكل هذين التعبيرين ما يسمى **بحلقة التكرار** ، وسوف نشرح كل هذا في الفقرات التالية :

### 9-1- تعبير (أذهب الى GOTO) :

تكون **الصيغة العامه** لهذا التعبير كما يلي :

```
n1 GOTO n2
```

حيث تمثل ( *n1*) رقم السطر الذي يحوي هذا التعبير . أما ( *n2*) فيمثل رقم السطر المطلوب الإنتقال اليه . كما في المثال التالي:

```
16 GOTO 121
```

أي معناه إذهب (أو إنتقل) إلى السطر المرقم (121) . ونستفيد من هذا التعبير في قطع التتابع الاعتيادي للبرنامج والإنتقال فوراً الى سطر آخر ، بغض النظر عن موقعه في البرنامج . وسوف نلمس الفائدة من ذلك في الامثلة التي ستتضمنها الفقرات التالية.

obeikandi.com

**2-9- تعبير (إذا... آنذاك... وإلا... IF... THEN...ELSE) :**

هذا التعبير هو من التعابير المشروطه ، أي أن الحاسبة تقوم بتنفيذ شيئاً ما اذا ما توفرت شروط (أو حالات) معينه تفيد هذا التنفيذ . فعلى سبيل المثال **إذا** كانت (X) تساوي (Y) **آنذاك** (Z) تساوي (5) **وإلا** أذهب لتنفيذ السطر رقم (90) . ويمكن ان نكتب ذلك في البرنامج وكما يلي :

```
18  IF X=Y THEN Z=5 ELSE GOTO 90
```

وهذا يعني إن الحاسبة سوف تقوم بفحص قيمتي المتغيرين (X) و (Y) ، فإذا وجدتهما متساويين عندئذ تضع قيمة (Z) تساوي (5) ، وتنتقل لتنفيذ السطر التالي للسطر المرقم (18) في البرنامج . أما اذا لم تكن قيمتي (X) و (Y) متساويتين فعندئذ لا تغير الحاسبة قيمة (Z) الاصلية بل سوف تنتقل لتنفيذ السطر رقم (90) عوضاً عن ذلك .

ولو اردنا (على سبيل المثال ايضاً) من الحاسبة أن تدقق قيمة متغيرٍ ما ، ولنسميه (X) ، **فإذا** كانت تساوي (10) **آنذاك** نطلب من الحاسبة ان تطبع عبارة (X) تساوي (10) **وإلا** (أي إذا لم يتحقق هذا الشرط) نطلب من الحاسبة ان تطبع عبارة أخرى مختلفة وهي (شيئاً ما خطأ) .

ويمكن ان نعبر عن ذلك بلغة بيسك كما يلي :

```
301 IF X=10 THEN PRINT "X EQUALE 10"
    ELSE PRINT "SOMETHING WRONG"
```

أو نكتبها باللغة العربية ، اذا كانت الحاسبة ثنائية اللغة (عربي - انكليزي) ، فنستبدل عبارة "SOMETHING WRONG" بعبارة " شيئاً ما خطأ " . ولا يفوتنا هنا ان نذكر ملاحظة هامه جداً وهي اننا حين نطلب من الحاسبة طبع أي نص ، دون ان تقوم بمعالجته (أي تطبعه كما هو دون تغيير) فإننا نضع النص داخل **علامتي إقتباس** ( " ).

**3-9- أسئلة محلولة :**

كي نتعود على كيفية وضع تعابير السيطرة (التي درسناها في الفقرتين اعلاه) سوف نقوم بحل بعض الأسئلة وكما يلي :

**سؤال** : جد مجموع الاعداد الفردية بين 0 و (1000) . أرسم المخطط الانسيابي وأكتب البرنامج لعمل ذلك .

**الحل** :

نفرض ان العدد الفردي هو متغير نرمز له بـ (X) والذي سيأخذ القيم التالية :

$$X = 1, 3, 5, 7, ...., 999$$

أما المجموع ، والذي سنرمز له بالرمز (SUM) ، فهو :

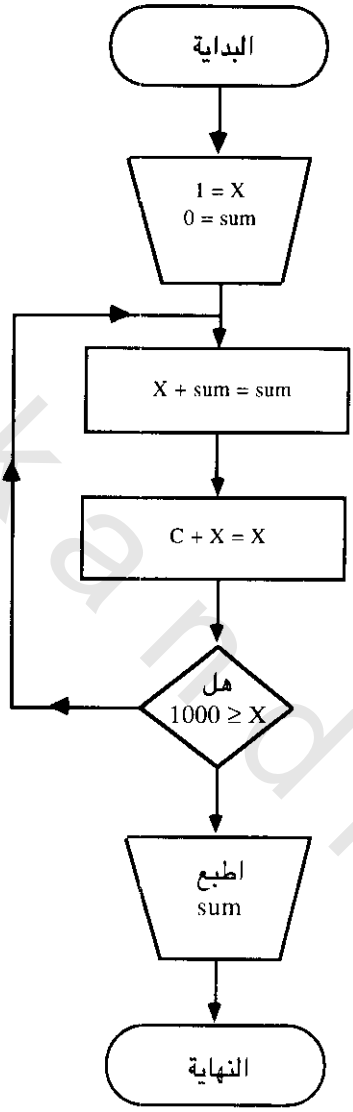
$$SUM = 1+3+5+7+.... + 999$$

وعند البداية ستكون قيمة (X) تساوي (1) ، وقيمة (SUM) تساوي (0) ، (راجع الفقرة (4-2) الفصل الأول).

فنقوم بجمع الاثنان (X) و (SUM) ونضع النتيجة كقيمة جديدة للمتغير (SUM) . أي أن (SUM) الآن تساوي (1) . وبعد ذلك نزيد قيمة (X) بمقدار (2) (أي تصبح (X) تساوي (3)) وذلك لأن (X) لا تأخذ سوى الأرقام الفردية . ثم نجمع (X) الجديدة بالمتغير (SUM) (أي 3+1) ونجعل ذلك كقيمة جديدة لـ(SUM)، وهكذا نستمر بالعملية . ونلاحظ هنا بأننا لا نستطيع السيطرة على الزيادة الحاصلة في قيمة (X) بحيث لا تتجاوز تلك القيمة العدد (1000) ، لذا يفترض ان نقوم دائما بفحص قيمة (X) قبل جمعها ، فإذا كانت أقل نستمر بعملية الجمع . أما إذا كانت قيمة (X) أكبر من (1000) فإننا نتوقف عن الجمع ، ونقوم بطبع آخر قيمة للمتغير (SUM) والذي يمثل المجموع النهائي .

مما تقدم نرى أن خوارزمي الحل هو خليط من **التتابع والانتخاب**

ويمكن ان نضع **المخطط الانسيابي والبرنامج** كما يلي :




obeikandl.com

```
13  X=1
14  SUM=0
15  SUM=SUM+X
16  X=X+2
17  IF X<=1000 THEN GOTO 15  ELSE
    PRINT SUM
18  END
```

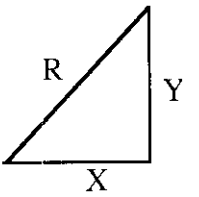
ومن البرنامج ، نلاحظ إنه قد استفدنا من **التعبير (إذا...آنذاك...وإلا...IF...THEN...ELSE)** في اتخاذ قرار بشأن قيمة المتغير (X) بحيث ان الحاسبة سوف تستمر بحساب قيمة المتغير (SUM) ما دامت قيمة (X) تساوي أو أقل من (1000) . أما إذا وصلت قيمة (X) الى أكبر من ذلك ، فأن البرنامج سوف يخرج من **حلقة التكرار** (الخطوات من (15) الى (17)) وسيطبع قيمة (SUM) ويذهب لتنفيذ السطر التالي (وهو (18)) الذي ينهي البرنامج .

ونلاحظ من الشرح اعلاه انه توجد هناك عدة سطور في البرنامج قد تكرر تنفيذها عده مرات ، وحتى تحققت حالة معينه لدينا (وهي X أكبر 1000) عند ذلك يتوقف التكرار . وتسمى مثل هذه الحالة **بحلقة التكرار** والتي سنشرحها لاحقاً ضمن فقرة منفصلة .

**سؤال** :اذا كان المتغير (X) هو ضلع في المثلث القائم الزاوية والذي يكون فيه الوتر (R) يساوي (3) والضلع الثالث (Y) يأخذ القيم (صفر ,4,3,2,1) . جد قيمة (X) لكل حالة من الحالات الخمسة السابقة للضلع (Y) .

**الحل :**

نرسم المثلث القائم الزاوية الذي وتره هو (R) وضلعيه المتعامدين هما (X) و(Y) وكما يلي :



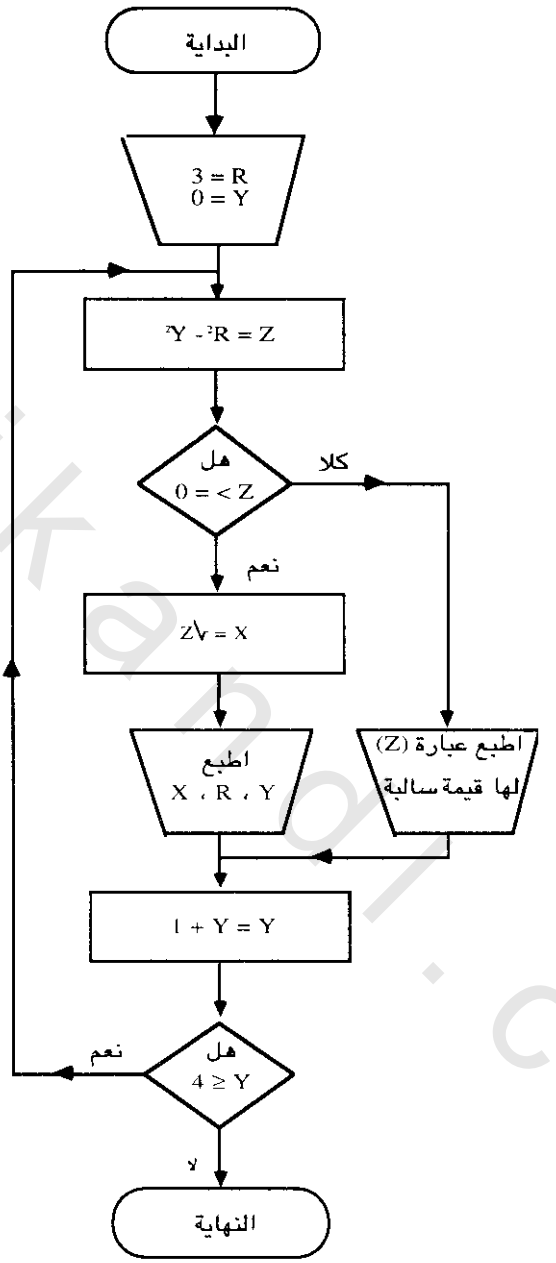

ولإيجاد قيمة (X) يمكننا ان نطبق **نظرية فيثاغورس** فنحصل على :

$$X = \sqrt{R^2 - Y^2}$$

وعند التعويض عن المتغير (Y) في المعادلة اعلاه بالقيم الخمسة الواردة ، نستطيع الحصول على قيم (X) التي تقابلها ، على أن نتذكر دوماً بأن القيمة العددية لـ ($R^2-Y^2$) وهي الكمية التي سنسميها (Z) والموجودة تحت الجذر التربيعي يجب ان تفحص بعد كل تعويض عن القيم، وذلك للتأكد بأنها كميه موجبه وليست سالبة. ونلاحظ ايضاً ان قيم (Y) تبدأ بالقيمة (0) وتنتهي بالقيمة (4) وبزيادة قدرها (1) في كل مرة .

وفيما يلي **المخطط الإنسيابي والبرنامج** لهذا المثال :

**المخطط الانسيابي :**




obeikandl.com

**البرنامج :**

```
10  R=3
20  Y=0
30  Z=R^ 2 - Y^2
40  IF Z>=0 THEN GOTO 70 ELSE GOTO 50
50  PRINT " Z HAVE NEGATIVE VALUE"
60  GOTO 90
70  X=Z^0.5
80  PRINT Y,R,X
90  Y=Y+1.0
100 IF Y<=4.0 THEN GOTO 30 ELSE GOTO
    110
110 END
```

**ملاحظات مهمة :**

1) يمكن ان نحذف كلمة GOTO عندما ترد في **تعبير (إذا ... آنذاك ...وإلا ...**IF...THEN...ELSE...) فنستطيع كتابة السطرين (40) و (100) كما يلي :

```
40  IF Z>=0 THEN 70 ELSE 50
100 IF Y<=4.0 THEN 30 ELSE 110
```

2) إذا كان السطر المراد الانتقال اليه بعد كلمة ELSE ، هو فعلاً السطر التالي فيمكن حذف كلمة ELSE وما بعدها . ويمكننا إعادة كتابة



obeikandi.com

التعبيرين (40) و (100) بالصيغة التالية :

```
40  IF Z >=0. THEN 70
100 IF Y<=4.0 THEN 30
```

في السطر (40) ستفهم الحاسبة بأنه اذا كانت قيمة (Z) أكبر أو يساوي الصفر فيجب أن تنتقل الى السطر المرقم (70) ، أما اذا كانت قيمة (Z) أصغر من صفر (أي لا يتحقق الشطر فأن الحاسبة سوف تنتقل لتنفيذ السطر التالي في البرنامج .

3) لو لم نضع السطر رقم (60) في البرنامج أعلاه فأن الحاسبة سوف تنفذ السطرين (50) و (70) واحد تلو الاخر . وهذا معناه بأنه رغم كون قيمة (Z) هي أصغر من صفر ( والدليل اننا نفذنا السطر (50)) فأن البرنامج سوف ينتقل لتنفيذ السطر (70) لعدم وجود ما يمنعه من ذلك. لذا فاننا لو لم نضع السطر (60) في البرنامج لأصبح لدينا خطأ في المعنى المقصود عند وضعنا للخوارزمية .

4) نلاحظ هنا انه توجد ايضا حلقة تكرار بين السطرين (30) و (100) يرتبط عدد مرات تكرارها بقيمة المتغير (Y) .

### 4-9- تعبير (لأجل FOR) :

عند حل السؤالين في الفقرة السابقة لاحظنا وجود عدة سطور في البرنامج قمنا بتكرار تنفيذها عدة مرات ، وقد اسميناها **حلقة التكرار**. وللخروج من **حلقة التكرار** يجب أن يتحقق عدد معين من مرات التكرار ، أو يجب أن يتحقق شرط معين يستوجب الخروج من هذه الحلقة . ويمكننا ان نفعل ذلك بإستخدام تعابير السيطرة ، ومنها **تعبير (لأجِل FOR)** مع **تعبير (التالي NEXT)** . ويجب ان يوضع **تعبير**

**(لأجل FOR)** كسطر أول في حلقة التكرار وتكون صيغته العامه كما يلي :

***n* FOR *v* = *e1* TO *e2* STEP *e3***

حيث تمثل ( *n*) رقم السطر في البرنامج ، و ( *v*) هو رمز لمتغير يستعمل **كعداد** نحسب من خلاله عدد مرات التكرار . أما ( *e1*) فهي تمثل القيمة الاولية (أو الابتدائية) للمتغير(v) ، تمثل ( *e2*) القيمة النهائية للمتغير(v)، أما الرم(e3) فهو يمثل مقدار الزيادة الحاصلة في قيمة ( *e1*) ، في كل مرة يعاد فيها تكرار الحلقة ، حتى تصل الى قيمة مساوية أو أقل من ( *e2*) . كما نرى ذلك المثال التالي :

**100 FOR X=1 TO 1000 STEP 2**

وهذا معناه ان الحاسبة ستقوم بالبدء بتكرار عدد من السطور التي تلي السطر رقم (100) في البرنامج . على أن تضع قيمة المتغير (X) مساوية للقيمة الابتدائية (أي *e1* في الصيغة العامه) وهي (1) ، عند تنفيذ حلقة التكرار لأول مرة . وبعد ان نصل الى السطر الاخير في **حلقة التكرار** يعاد تنفيذ الحلقة من جديد ولكن قيمة المتغير (X) ستزيد عن القيمة السابقة **بخطوة** (STEP) مقدارها *e3* (وهي (2) في هذا المثال) . ويتم تدقيق (X) (بعد هذه الزيادة) مع القيمة النهائية لـ(X) ، (أي *e2* في الصيغة العامة ) والتي تساوي هنا (1000) ، فإذا كانت أكبر من (1000) يتم الخروج من **حلقة التكرار** وينتقل البرنامج لتنفيذ السطور التي تقع بعد الحلقة المباشرة . أما إذا كانت قيمة (X) بعد الزيادة أقل أو تساوي (1000) فإن البرنامج يستمر بتنفيذ **حلقة التكرار**.

ويعود من جديد الى السطر الاول في الحلقة ويزيد (X) بمقدار (2) ويتم ايضاً تدقيق قيمة (X) مع القيمة النهائية وهكذا . ويمكن ان نلخص ذلك بأن المتغير (X) يأخذالقيم (999,...,9,7,5,3,1) .

علما بأن قيمة المتغير( *e3*) في الصيغة العامه اذا كانت (1) ، فأنه يمكننا الغاء (الخطوة **STEP**) وما بعدها في الصيغة العامه مثل :

```
20 FOR ZAL=12 TO 24
```

وهذا معناه ان قيمة المتغير (ZAL) تبدأ بقيمة إبتدائية مقدارها (12) وتنتهي بقيمة نهائية مقدارها (24) على ان تكون الزيادة في كل مرة بقدر (1) . أي ان المتغير ZAL يأخذ القيم (24,...,14,13,12) على التوالي مع كل اعادة تنفيذ **لحلقة التكرار** .

**5-9- تعبير (التالي NEXT) :**

وهو يمثل آخر سطر في **حلقة التكرار** ، حيث يجب ان تنتهي حلقة التكرار التي تبدأ **بالتعبير (لاجل FOR)** بسطر اخير في الحلقة يحوي **التعبير (التالي NEXT)** وصيغته العامه هي :

```
n NEXT v
```

حيث تمثل *n* رقم السطر ويمثل المتغير *v* رمز المتغير المستعمل في تعبير (**لاجل FOR**) والذي سبق شرحه فلو اردنا مثلاً التعبير عن **حلقة التكرار** للمثال الاول في الفقرة السابقة فيمكننا ان نكتبه بالصورة التالية :

```
...  .....
...  .....
90   .....
100 FOR  X=1  TO  1000  STEP 2
...  .....
...  .....
...  .....
310 NEXT  X
...  .....
...  .....
```

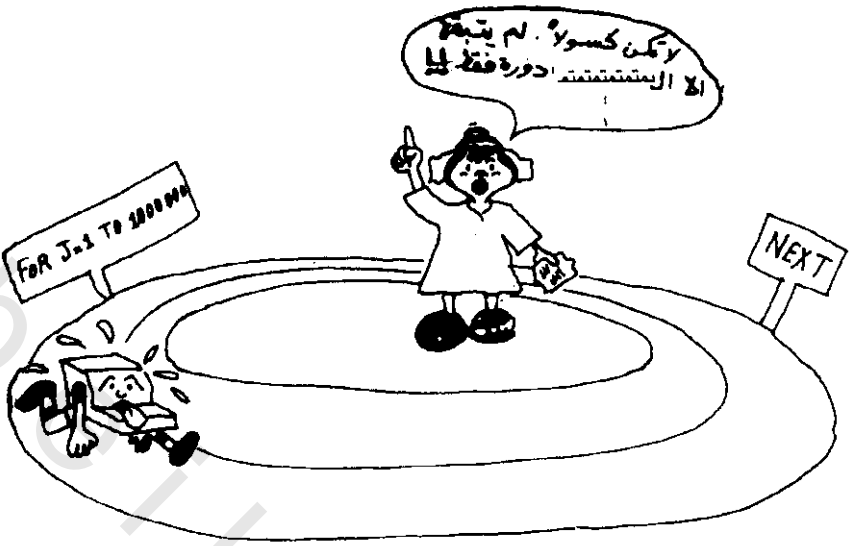
أما المثال الثاني في الفقرة السابقة فيمكننا كتابته بالصورة التالية:

```
...  .....
...  .....
...  .....
20   FOR  ZAL=12  TO  24
...  .....
...  .....
185 NEXT  ZAL
...  .....
...  .....
```

حيث تمثل الخطوط المتقطعة في الامثلة اعلاه سطور في البرنامج .
وتبدو **حلقة التكرار** كجزء منه .



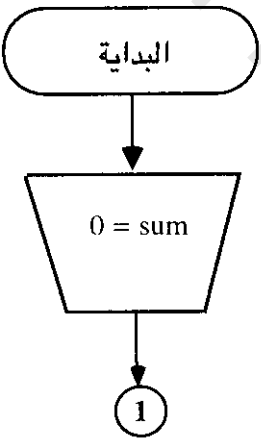
obeikandl.com

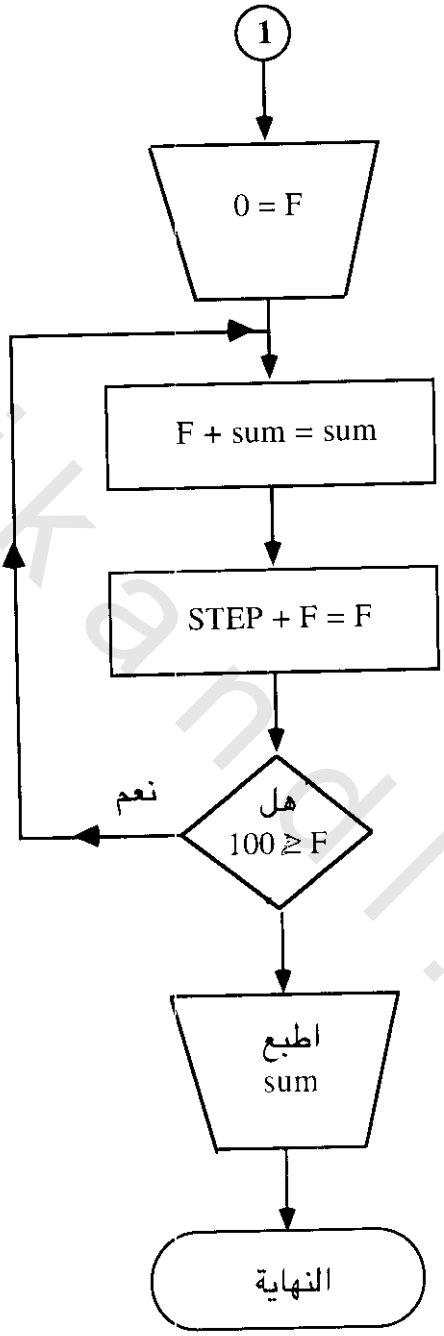



## 6-9- اسئلة محلولة :

**سؤال** : جد مجموع الاعداد الزوجية بين (0) و (100) . ارسم المخطط الانسيابي ، واكتب برنامجاً لعمل ذلك على أن تستخدم تعبيري (**لأجل FOR**) و (**التالي NEXT**) .

**الحل** :






obeikandl.com

```
113  SUM=0.
114  FOR F=0 TO 100 STEP 2
115  SUM=SUM+F
116  NEXT F
117  PRINT "THE SUMATION OF ODD
     NUMBERS BETWEEN (0) AND (100) =" ; F
118  END
```

واذا كانت الحاسبة التي نشغل عليها البرنامج ثنائية اللغة (عربي - انكليزي) فيمكن ان نكتب السطر (117) بصيغة أخرى هي :

```
117  PRINT "= (100) و (0) مجموع الاعداد الفردية"
     ; F
```

ولو قارنا هذا الحل مع حل السؤال في الفقرة (3-9) من هذا الفصل والخاص بايجاد مجموع الاعداد الفردية . لأستطعنا تمييز الفرق بين الطريقتين ومدى سهولة حل السؤال بالطريقة الاخيرة .

**سؤال** : اذا علمت ان قيمة (Y) تساوي (2) جد قيمة مجموع خمسة حدود من المتوالية التالية :

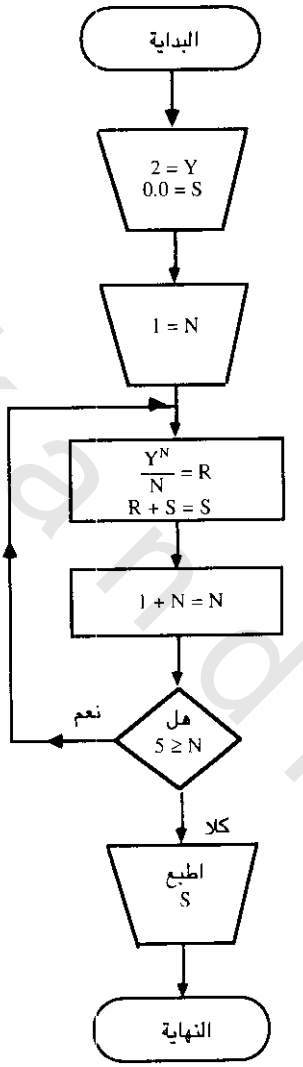
$$S= Y+Y^2/2 +Y^3/3 +Y^4/4+- - - - +Y^n/n$$

ارسم المخطط الانسيابي وأكتب برنامجاً للحل .

**الجواب** :

نلاحظ وجود معامل يرافق المتغير (Y) في كل حد من حدود المتوالية وسوف نرمز له بالرمز (n) ، وهو يمثل رقم الحد في المتوالية . ولو رمزنا للمجموع بالرمز (S) والذي يمثل مجموع الخمسة حدود الاولى

من المتوالية (أي أن n =1,2,3,4,5) . فيمكننا وضع المخطط الانسيابي بالصورة التالية :




obeikandl.com

```
10   Y=2.
20   S=0.0
30   FOR  N=1  TO  5
40   R=(Y^ N)/N
50   S=S+R
60   NEXT N
70   PRINT S
80   END
```

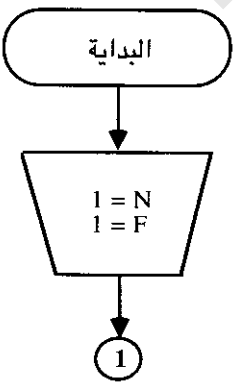
**سؤال** : ما هو مفكوك العدد (n) . ارسم **المخطط الانسيابي** واكتب **برنامجا** لإيجاد مفكوك العدد (10) .

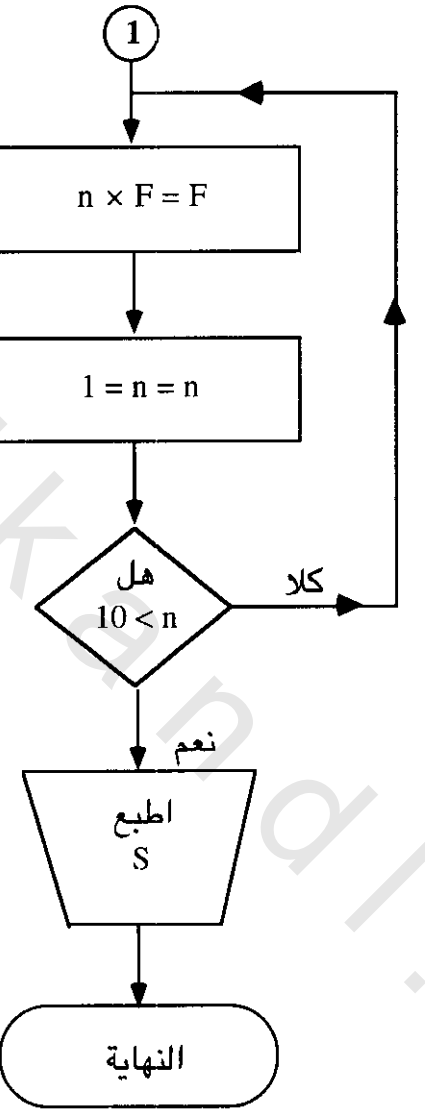
**الجواب** :

مفكوك العدد (n) هو حاصل ضرب كل الاعداد بين (1) و (n) ، وبزيادة مقدارها (1) في كل مرة ، في بعضها أي ان مفكوك

$$(n) = 1 \times 2 \times 3 \times 4 \times \ldots . \times n$$

نرمز لقيمة المفكوك برمز المتغير (F) والذي تكون قيمته الابتدائية تساوي(1). ويمكن ان نرسم **المخطط الانسيابي** كما يلي :






والبرنامج يمكن ان يأخذ الصيغة التالية :



obeikandl.com

```
50  N=1
60  F=1
70  FOR  N=1  TO  10
80  F=F*N
90  NEXT  N
100 PRINT  " FACTORIAL OF (10) = " ; F
110 END
```

إذا كانت الحاسبة التي نشغل عليها البرنامج ، ثنائية اللغة (عربي - انكليزي) فيمكن ان نكتب السطر (100) بصيغة اخرى هي:

```
100 PRINT " = (10)  مفكوك  " ; F
```

## 10- حلقات التكرار :

لقد مرينا في الفقرات السابقة مصطلح (**حلقات التكرار**) ، وعلمنا انها عبارة عن حلقة وهمية تربط مجموعة سطور في الحاسبة يعاد تكرارها عدة مرات اثناء تنفيذ البرنامج ، وكما مرينا في السابق فأن الصيغة العامه لكل من التعبيرين هما :

```
n1  FOR  v  =  e1  TO  e2  STEP  e3
...  ..............
...  ........................

n2  NEXT  v
```

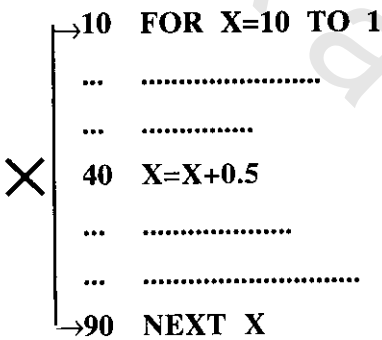


oberkanic.com

حيث تمثل ( *n1*) و ( *n2*) ارقام السطور لبداية ونهاية حلقة التكرار على التوالي. و ( *v*) يمثل **الدليل** أو **العداد** الذي نحسب بواسطته عدد مرات تكرار الحلقة . أما ( *e1*) و ( *e2*) فتمثلان بداية ونهاية قيمة الدليل ( *v*) على التوالي . وتمثل ( *e3*) مقدار الزيادة في قيمة المتغير ( *v*) في كل مرة . وحلقة التكرار هي كافة السطور في البرنامج بدءاً من السطر ( *n1*) وحتى نصل الى السطر ( *n2*) .

وهناك قواعد يجب مراعاتها عند استخدامنا لمثل هذه الحلقات في البرنامج وأهمها ما يلي:

1) **يجب ان لا تتغير قيمة الدليل (أو العداد) داخل حلقة التكرار** ، واذا اضطررنا لتغييره لغرض استخدامها في تسهيل حل المسألة فإننا يجب نغير اسمه مثل :




فهذه الحالة خطأ . وذلك لأن تغيير العداد داخل الحلقة ، تحت نفس الأسم (أي X) ، يسبب الإرباك للحاسبة وينتفي عمل **العداد** (أو **الدليل**) . أما اذا اضطررنا لذلك فيجب ان يكتب السطر المرقم (40) بصيغة أخرى مثل :

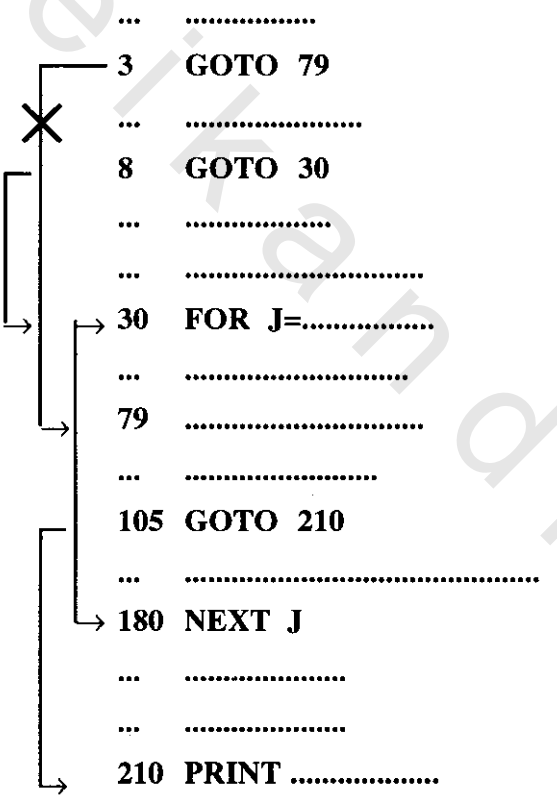
```
40 X1=X+0.5
```

2) لنفس السبب اعلاه **لا يجوز ادخال قيمة جديدة للدليل او العداد ، بأي طريقة كانت ، داخل الحلقة** السابقة مثلاً :

```
75 LET X=20
```

فهذه الحالة خاطئة .

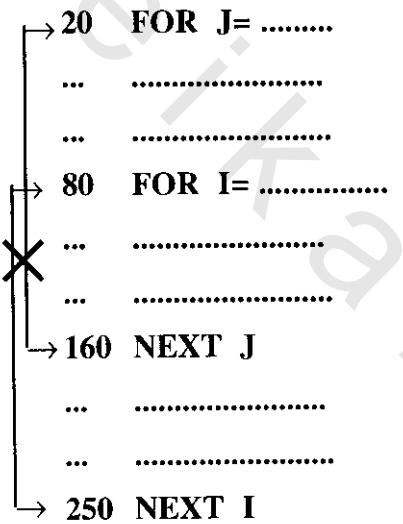
3) **لا يجوز الدخول الى حلقة التكرار إلا من بدايتها . ويجوز الخروج من داخل الحلقة الى خارجها** . مثال:




obeikandl.com

فيجوز مثلاً في السطر (8) ان نقول (GOTO 30) وذلك لأن السطر المرقم (30) يمثل بداية الحلقة ولا يجوز أن نقول (GOTO 79) ، وذلك لأن السطر (79) لايمثل بداية الحلقة . كما نلاحظ في السطر المرقم (105) اننا استطعنا الخروج من حلقة التكرار وهذه الحالة تجوز .

4) **لا يجوز ان تتقاطع حلقتي تكرار مع بعضهما** مثل :




وهذه تمثل حالة خاطئة .

5) **يجوز ان تنتهي أكثر من حلقة تكرار بنفس السطر** . وعند ذاك تكون الصيغة العامه لتعبير (**التالي NEXT**) كما يلي:

**n NEXT v1, v2, v3**

وتمثل *v1, v2, v3* .. رمز العداد لكل حلقة من الحلقات . ونرى ذلك بالتفصيل في المثال التالي :

```
→ 510  FOR  K= .....
  ...  ..........................
→ 580  FOR L= .....
  ...  ..................................
→ 670  FOR M= ..............
  ...  ..................
  ...  ..................
→ 830  NEXT  M,L,K
  ...  ..................
  ...  ..................
```

ويجب ان نذكر هنا ان الحلقة الداخلية (والتي رمز العداد فيها هو (M) ) سوف تنفذ كاملةً ، أي (M) من المرات . بينما لا تتكرر الحلقة التي عدادها هو (L) سوى مرة واحدة فقط . ونفس الشيء ينطبق على الحلقة التي عدادها (K) .

6) **يجوز ان تتداخل اكثر من حلقة تكرار مع بعضها على ان لا تتقاطع** ، وكما يلي :



obeikandl.com

```
100  FOR  T= .....
...  .....
200  FOR  W= .....
...  .....
...  .....
250  NEXT  W= .....
...  .....
300  FOR  A= .....
...  .....
...  .....
...  .....
...  .....
450  NEXT  A
700  NEXT  T
```

obeikandl.com