البسكال – خطوة بخطوة

محمد أنس بن عثمان

الإهـــــــــداء

أقدم خالص شكري وتقديري إلى عائلتي العزيزة وإلى أصدقائي الأعزاء وإلى كل من سيقدم اهتماما بهذا الكتاب الذي يحتوي دروس متواضعة وبسيطة وأيضا إلى كل من كان يتابع سلسلة دروسي الصيفية في تعلّم لغة البرمجة خصوصا منهم تيسير بوبكر, محمد الأخضر الشرفي, رحاب محجوبي, فاطمة محجوبي, حيدر محجوبي, رحاب حفّوظي, محمد أمين خليل وندين العبيدي وإن شاء الله في الإصدار القادم سأضيف العديد من التفاصيل والمسائل والدروس.

محمد أنس بن عثمان

المقدمة

لأني درّست مجموعة من الشباب والشّابات, أردت أن أكمل الجميل بترك تذكار لهم وبدأت الفكرة تتطوّر بمرور الوقت فقدت بدأت في دروة صغيرة لتعليم لغة البرمجة البسكال عن طريق الأنترنت وبعد أن أنهيت بعض الدروس خطرت ببالي فكرة هذا الكتيّب الذي أعتبره مرجعا رائعا لكل تلميذ يدرس البرمجة وهذا هو التذكار الذي وددت أن أتركه إلى أصدقائي الأعزاء وفي هذا الكتاب تناولت لغة البرمجة البسكال ورجائي أن يصبح لكلّ من يقرأ هذه الصفحات شغف بالبرمجة وكتابي هذا مجّانيّ وهو مقدّم للجميع وبشكل خاص لمن أعار سلسلة دروسي اهتماما.

إذن فالبسكال هي لغة برمجة وتعريفي للغة البرمجة هو أنّها لغة تواصل بين الحاسوب ومستعمله يقوم بها بصنع برمجيات تقوم بتنفيذ جملة من التعليمات, هذه التعليمات أثناء البرمجة تجمّع وتحوّل إلى برنامج يقوم بتنفيذها عند فتحه وتشغيله.

يستحسن تنزيل برنامج التربو بسكال الذي سنستعمله في البرمجة, للحصول على الرابط يمكنك الإتصال بي عن طريق بريدي الشخصي medanasbo@hotmail.fr أو عن طريق هذا الرابط http://www.mediafire.com/?zpi3xekgg478j3t

مرحبا بالعالم

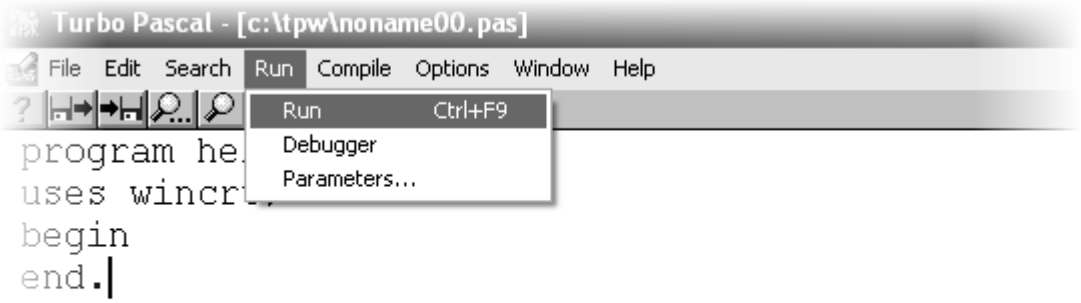
السلام عليكم في هذا الدرس سنتعرف بعض الأساسيات في لغة البرمجة البسكال وسنستخدم مترجم "Turbo Pascal"
إذن فل نبدأ بفتح التربو بسكال ولنكتب هذه الأسطر :

```
program hello;
uses wincrt;
begin
end.
```

في هذه الأسطر لدينا 4 كلمات مفتاح وهي program, uses, begin و end.
كما أنّ لدينا السطران الأول والثاني ينتهيان بنقطة وفاصلة ;
سأقوم بالشرح سطرا سطرا:

- **السطر الأول نقوم من خلاله بإخبار البسكال عن اسم البرنامج الذي نود صنعه وقد قمنا بتسميته hello وذلك عن طريق استخدام الكلمة المفتاح program وبما أن هذه تعليمة فيجب أن نقوم بإنهائها بالنقطة والفاصلة ; فكل تعليمة في لغة البرمجة البسكال تنتهي بالنقطة والفاصلة, لا يمكن أن نسمي اسم البرنامج "program" فهي كلمة مفتاح أي أنها محجوزة من طرف البسكال وهذا يعني أنّها خاصة فقط بالبسكال ولا يمكن استعمالها في تسمية البرنامج أو المتغيرات (سنتعرف عليها في الدرس القادم).**
- **السطر الثاني نقوم من خلاله باستدعاء مكتبة wincrt وهي مكتبة فيها مجموعة من التعليمات الجاهزة التي تجعل علمية البرمجة أسهل كما أنّه لا يمكن أن نقوم بصنع برنامج بدون استدعائها إذا نستعمل التربو بسكال, نستعمل uses وهي كلمة مفتاح للقيام بعملية الاستدعاء ودائما التعليمة يجب أن تنتهي بنقطة وفاصلة.**
- **السطر الثالث هو begin ومن خلاله نخبر البرنامج أنّ مجموعة التعليمات التي سينفذها ستبدأ و end. التي تنتهي بنقطة وهو شرط أكيد نخبر بها البرنامج أنّه قد انتهى من تنفيذ التعليمات.**

في برنامج التربو بسكال نقوم بالذهاب إلى قائمة "RUN" ونختار الأمر "RUN" مثل ما هو موضح بالصورة أدناه



سيقوم بتشغيل برنامج ذو نافذة بيضاء وفارغة, طبعا هي فارغة لأننا لم نقم بوضع تعليمات ليقوم بتنفيذها, هذه التعليمات يجب أن تكون مكتوبة بين **begin** و **end.**

نقوم بإغلاق النافذة التي فتحة التي صنعها البسكال ومن ثمة نعود إلى التبرو بسكال لنضع له أوّل تعليمة وهي تعليمة الكتابة أو الطباعة التي نقوم من خلالها بكتابة نص في تلك النافذة إذا نقوم بكتابة هذه التعليمة كما قلت بين الـ **begin** والـ **end.**

```
write('Hello World');
```

بعد أن قمنا بكتابتها نقوم مرة ثانية بتشغيل البرنامج عن طريق الأمر **"RUN"** والنتيجة ستكون نافذة مكتوب فيها **Hello World** وهو النص المكتوب بين ضفرين وهذان الضفران موجودان بين القوسين, القوسين نضع داخلهما إعدادات التعليمة **"write"** الإعدادات التي وضعناها هي واحدة فقط وهي سلسلة من الأحرف وهي **"Hello World"** مكتوبة بين ضفرين ' وذلك من أجل إخبار أن ما قمنا بكتابته هو سلسلة من الأحرف.

لو نقوم بالقيام بتجربة أخرى لنفهم أكثر كيفية التعالم مع تعليمة **write** لنقم بكتابة هذه التعليمات بدلا من الأخرى.

```
write('Hello');
write(' World');
```

ستقوم هذه التعليمة بنفس الشيء إذا فتعليمة **write** دائما تكتب الإعداد الموجودة داخلها في نفس السطر بالبرنامج وللذهاب من أجل الكتابة في السطر التالي بالإمكان استخدام تعليمة **writeln**, مثال على ذلك:

```
write('Hello');
writeln(' World');
```

وبالنسبة إلى البرنامج الذي قمنا بصنعه فيمكننا إيجاده في المسار التالي:

```
C:\TPW\
```

ويكون اسمه موافقا لاسم الملف الذي فيه مجموعة التعليمات (يسمّى أيضا بملف السورس كود) على سبيل المثال البرنامج الذي قمنا بصنعه الآن قمت بكتابة السورس كود الخاص به في ملف باسم noname00.pas فالبرنامج الذي قمت بصنعه اسمه noname00.exe, مثال على ذلك إلى اليسار نجد ملف السورس كود وإلى اليمين نجد البرنامج:

ملاحظة:

اسم البرنامج يجب أن يكون متكونا من أحرف من A إلى B أو أعداد أو مطه _

انتهى الدرس

المتغيرات

المتغيرات هي كائنات تتميز بثلاثة خصائص وهي كالآتي :

1. **الاسم**
2. **النوع**
3. **القيمة**

كما أنها سميت المتغيرة لأنه من الممكن أن نغير قيمتها.

ملاحظة:

اسم المتغيرة له نفس شرط تسمية البرنامج.

على سبيل المثال لدينا متغيرة من نوع "متغيرة حاملة لسلسلة من الأحرف" أي "string" اسمها "TEXT" وقيمتها "Hello World" فمن الممكن أن نغير قيمتها فتصبح "I am a string" أما بالنسبة للتعامل المتغيرات مع تعليمة الكتابة "write" أو "writeln" فالمطلوب فقط وضع اسم المتغيرة داخل الإعدادات, مثلا

```
writeln(TEXT);
```

وهنا لم نضع اسم المتغيرة بين ضفرين فلو فعلنا ذلك لكانت التعليمة ستكتب "A" بدلا من قيمتها فكل شيء مكتوب بين ضفرين في هذه التعليمة يكتب مثلما هو.

بالنسبة لعملية تعيين المتغيرات فهي تتم باستعمال الكلمة المفتاح "var" تليها لائحة بمجموعة المتغيرات التي نود تعيينهم نضع فيها اسم المتغيرة ومن ثمة نقطتان تليهما نوع المتغيرة وننهي تعليمة التعيين بنقطة وفاصلة, أمثلة :

```
program variables;
uses wincrt;
var
   number : integer;
   number2 : real;
   text : string;
   character : char;
   binary : bool;
begin
end.
```

نلاحظ أنّ تعيين المتغيرات يجب أن يكون قبل بداية البرنامج أي قبل begin.

هنا لدينا أنواع عديدة من المتغيرات وفي هذا الجدول سنوضح الفرق بينهم:

النوع	قيمتها يمكن أن تكون:
Integer	مجموعة الأعداد الموجبة والسالبة فقط بدون فاصلة
Real	أي عدد مهما كان بالفاصلة, موجبا أو سالبا
String	مجموعة من الحروف والرموز والأعداد
Char	حرف أو رمز أو رقم واحد فقط
Boolean	صحيح أو خطأ

هناك طريقتان لإسناد قيم إلى المتغيرات

1. نقوم نحن بإسنادها إلى المتغيرة عن طريق كتابة القيمة في السورس كود, على سبيل المثال نخبره بأن المتغيرة "number" من نوع "integer" قيمتها تساوي "9".
2. عن طريق تعليمة "readln" أو "read" الفرق بينهما مثل الفرق بين تعليمتي الكتابة, وهما تعليمتان يعطيان فرصة إلى مستخدم البرنامج (خلال استعماله) بكتابة القيمة التي يريد ومن ثمة إسنادها إلى المتغيرة عن طريق الضغط على زر الإرسال ENTER.

الطريقة الأولى:

نقوم بكتابة اسم المتغيرة يليه نقطتان وعلامة مساواة تليها القيمة وننهي التعليمة بالنقطة والفاصلة, أمثلة على ذلك :

```
number := 1404;
number2 := -2.07;
text := 'Hello Mr.Reader';
character := 'Y';
bool := false;
```

نلاحظ أن المتغيرات التي تحمل قيمتها أحرفا يجب أن تكتب بين ضفرين.

الطريقة الثانية:

نقوم بكتابة اسم المتغيرة داخل إعدادات التعليمة "readln", أمثلة :

```
readln(number);
readln(text);
```

تنتظر هذه التعليمة مستعمل البرنامج حتى يكتب القيمة التي يريد ثم يغط على "ENTER".

برنامج بسيط:

سنقوم الآن بصنع برنامج يطلب من مستعمله أن يكتب اسمه ومن ثمة يرحب به فيقول له مرحبا بك Mr. ثم يكتب اسمه, إذا فالمراحل هي كالآتي :

1. نطلب من مستعمل البرنامج أن يكتب اسمع عن طريق تعليمة الكتابة
2. يقوم المستعمل بكتابة اسمه في متغيرة من نوع "string" عن طريق تعليمة القرائة "readln"
3. نقوم بكتابة "Hellor Mr." تليها قيمة المتغيرة عن طريق تعليمة الكتابة "write"

إذا نقوم أولا بتسمية البرنامج, على سبيل المثال "lesson2" ثم نستدعي مكتبة "wincrt" ومن ثمة نعين المتغيرات التي سنعمل بها وهي متغيرة واحدة التي سنضع فيها الاسم ثم نقوم بوضع التعليمات التي قمنا بترتيبها في العنصر السابق, إذن فالبرنامج سيكون هكذا :

```
program lesson2;
uses wincrt;
var
   name : string;
begin
   writeln('Please write your name : ');
   readln(name);
   write('Hello Mr.');
   write(name);
end.
```

لم نستعمل "writeln" في النهاية وذلك لنكتب الجملة في سطر واحد ويمكن أيضا أن نستعمل طريقة أخرى وهي أن نكتب كل شيء داخل تعليمة واحدة فقط ونفصل بين الإعدادات بفاصلة "," فيكون الشكل هكذا :

```
writeln('Hello Mr.', name);
```

العمليات الحسابية:

من الممكن أن نستعمل العمليات الرياضية مع المتغيرات العددية فمثلا نقوم بجمع قيمتين, هذه لائحة لمجموعة العمليات التي يمكننا أن نستعملها مع المتغيرات :

نوع المتغيرة	العملية	الرمز
Integer	الجمع	+
	الطرح	-
	الضرب	*
	خارج القسمة	div
	باقي القسمة	mod
Real	الجمع	+
	الطرح	-
	الضرب	*
	القسمة	/

أمثلة عن كيفية استعمالها, لنأخذ ثلاثة متغيرات A, B و C من نوع integer

```
var
   A, B, C : integer;
```

ملاحظة:

من الممكن تعيين أكثر من متغيرة بنفس النوع في نفس التعليمة وذلك بفصل أسمائهم بفاصلة " , ".

الآن سأسند للمتغيرة C قيمة المتغيرة A مجموعة مع قيمة المتغيرة B, إذن أفعل هكذا:

```
C := A + b;
```

ملاحظة:

البسكال لا يبالي إن كانت الحرف كبيرة أم صغيرة فهو لا يفرق بين " A" و " a".

هنا قمنا بكتابة اسم المتغيرة C تليها نقطتان وعلامة تساوي تليها القيمة تليها نقطة وفاصلة, القيمة هنا هي قيمتا A و B مجمّعتان.

بالنسبة للفرق بين أنواع القسمة الخاصة بالنوع integer :

```
A := 16 div 5;
```

هنا A ستكون قيمتها 3 وهي خارج القسمة والباقي هو 1

```
B := 16 mod 5;
```

هنا B ستكون قيمتها 1 لأن باقي قسمة 16 على 5 يساوي 1 والخارج يساوي 3.

16 = 1 + 5 * 3

```
C := 16 / 5;
```

هنا C قيمتها تساوي 3.2 إذن فـ C هي متغيرة من نوع أعداد حقيقية.

انتهى الدرس

الشروط والاحتمالات

الشرط أو الاحتمال هو وضعية إمّا أن تكون صحيحة أو أن تكون خاطئة, على سبيل المثال قمنا بصنع برنامج يطلب من المستعمل أن يكتب عددا ومن ثمة يرى إن كان هذا العدد موجب أم سالب, فإن كان العدد أكبر أو يساوي من صفر فهو موجب وإن كان العكس فهو سالب, في البداية سنبدأ بشرط واحد نرى به إن كان موجبا أم لا فإن كان موجبا يعلمنا بذلك وإن لم يكن كذلك يتوقف البرنامج عن العمل.

الشرط نستعمله عن طريق الكلمة المفتاح if بعدها نفتح قوسين ونضع بينهما الشرط وبعد ذلك نكتب then و begin و end;

هنا نلاحظ أنّنا استعملنا النقطة والفاصلة في كلمة end

```
if (الشرط) then
begin
end;
```

بين الـ begin و end; نقوم بوضع مجموعة التعليمات التي سينفذها البرنامج في حالة ما كان الشرط صحيحا, لنصنع برنامجا صغيرا.

بالنسبة لشرطنا المتعلق بهذا المثال سنستعمل المتغيرة لنطبق عليها الشرط فالبداية نقوم بتعيين المتغيرة ولنسمي البرنامج lesson3 فالكود سيكون هكذا:

```
program lesson3 ;
uses wincrt ;
var
   number : real ;
begin
   writeln('Please write a number') ;
   readln(number) ;
end.
```

هنا نقوم بوضع الشرط وهو إن كان العدد أكبر أو يساوي من الصفر فيجب عليه أن يعلمنا بذلك, الشرط سيكون هكذا:

```
if (number >= 0) then
begin
   writeln('The number that you wrote is positive!') ;
end ;
```

إذن الشرط هو علامة أكبر بجنبها تساوي تليها القيمة التي نريدها في الشرط وهي الصفر, في هذا الجدول لائحة من الرموز التي يمكن أن نستعملها في الشروط ومعها شرحها

الرمز	الشرح
>	أكبر
<	أصغر
>=	أكبر أو يساوي
<=	أصغر أو يساوي
=	يساوي
<>	لا يساوي

سنقوم الآن بتطوير البرنامج بوضع احتمال آخري يخبرنا إن كان الرقم سالبا, وذلك بكتابة نفس التعليمة السابقة ولكن مع شرط معكوس وطبعا يجب أن نكتبها بعد الشرط القديم وقبل .end والشرط هو أن يكون العدد أقلّ من الصفر:

```
if (number < 0) then
begin
   writeln('The number that you wrote is negative!') ;
end ;
```

يمكننا أن نقوم بهذا الشرط بطريقة أخرى, إن أخطأ الشرط الأول يقوم بفعل تعليمات أخرى وذلك باستعمال الكلمة المفتاح else تليها begin و end ; بينهما مجموعة من التعليمات التي سينفذها, إذن فالتعليمات ستكون هكذا:

```
if (number >= 0) then
begin
   writeln('The number that you wrote is positive!')
end
else
begin
   writeln('The number that you wrote is negative!') ;
end ;
```

ملاحظة:

إذا ما توفرت مجموعة شروط ملتصقة تنتهي بـ else يجب أن لا نستعمل فيها النقطة والفاصلة إلا بعد الـ else

بالإمكان أن نضع عديد الشروط ملتصقة وذلك باستعمال else if يليها الشرط و then, مثال على ذلك:

```
if (number > 0) then
begin
   writeln('The number that you wrote is positive!')
end
else if (number = 0) then
begin
   writeln('The number that you wrote is equal zero')
end
 else
begin
   writeln('The number that you wrote is negative!') ;
end ;
```

إذا كانت لدينا تعليمة واحدة في شرط ما فيمكننا الاستغناء عن الـ begin و end, مثال على ذلك:

```
if (number >= 0) then
   writeln('The number that you wrote is positive!')
else
   writeln('The number that you wrote is negative!') ;
```

في حالة ما كثرة الشروط بالإمكان استعمال الكلمة المفتاح case وذلك بكتابتها تليها مباشرة اسم المتغيرة التي سنستعمل شرط المساوات معها يليها الكلمة المفتاح of تليها مجموعة الشروط و end ; تنتهي طبعا بالنقطة والفاصلة, في الشروط نضع قيمة المساوات تليها نقطتان تليها begin و end ; بينهما مجموعة التعليمات التي تنفذ في حالة ما صحّ الشرط, مثال على ذلك:

```
case number of
   1 : begin
          writeln('The number is equal one') ;
       end;
   2 : begin
          writeln('The number is equal two) ;
       end;
   3 : begin
          writeln('The number is equal three') ;
       end;
   4 : begin
```

```
        writeln('The number is equal four') ;
    end;
end;
```

في حالة ما كان لدينا تعليمة واحدة لكلّ شرط إذا ما تحقق فمن الممكن الاستغناء عن الـ begin و end;

في حالة ما أردنا أن يقوم بتعليمات إذا ما كانت الشروط خاطئة فمن الممكن استعمال الكلمة المفتاح otherwise وذلك بهذا الشكل:

```
case number of
  1 : begin
        writeln('The number is equal one') ;
      end;
  2 : begin
        writeln('The number is equal two) ;
      end;
  3 : begin
        writeln('The number is equal three') ;
      end;
  otherwise
      begin
        writeln('The number is not equal either one or two or three!') ;
      end;
end;
```

يمكن أن نضع مجموعة من التعليمات في حالة ما تحقق شرطان وذلك بطريقتين:

```
if (number >= 0) then
begin
  if (number <= 100) then
  begin
    writeln('The number that you wrote is between 0 and 100');
  end;
end;
```

أو يمكن ذلك بطريقة أسهل باستعمال الكلمة المفتاح and وذلك بكتابة الكلمة المفتاح if تليها الشرطان كل شرط بين قوسين مفصولان بكلمة and, مثال على ذلك:

```
if (number >= 0) and (number <= 100) then
begin
   writeln('The number that you wrote is between 0 and 100');
end;
```

بالنسبة لهذا الشرط وهو إن كان العدد محصورا بين عددين يمكن تعويضه أيضا بطريقة أخرى أيضا باستعمال الكلمة المفتاح in وذلك بكتابة اسم المتغيرة تليها in يليها المجال التي تنتمي إليه, مثال على ذلك:

```
if (number in [0..100]) then
begin
   writeln('The number that you wrote is between 0 and 100');
end;
```

المجال هو معقوفين كتب بينهما بداية المجال ونهايته مفصولان بنقطتين ".."

يمكن أيضا استعمال المجال مع الأحرف وذلك بكتابة الحرف بين ضفرين, مثال على ذلك:

```
if (character in ['a'..'z']) or (character in ['A'..'Z']) then
begin
   writeln('The character that you wrote is between a and z or A and Z');
end;
```

لأن البسكال يفرّق في قيم المتغيرات بين الحروف الكبيرة والصغيرة وضعنا شرطان يتحقّقان إن صحّ أحدهما وذلك بفصلهما بالكلمة المفتاح or.

إذا ما أردنا أن يتحقق عكس شرط ما يمكن استعمال الكلمة المفتاح not وهي تقوم بعكس نتيجة الشرط العادي وذلك بكتابة الكلمة المفتاح if يليها الشرط مكتوب قبله not, مثال على ذلك:

```
If not (number = 0) then
begin
   writeln('The number that you wrote is not equal zero!');
end;
```

يمكن أن نضع في الاحتمال شرطين او أكثر, مثال على ذلك:

```
if (character in ['a'..'z']) or (character in ['A'..'Z']) or (number in [1..100]) then
begin
   writeln('One or more of the conditions is true!');
end;
```

انتهى الدرس

المصفوفة

المصفوفة هي متغيرة تحمل أكثر من قيمة واحدة بشرط أن تكون جميع القيم من نفس النوع, لكل قيمة رتبة خاصة بها, يمكن أن نعبر عليها برسم مثال بسيط عليها في جدول, هكذا مثلا:

رتبة القيمة	1	2	3	4
القيمة	-56	95	7	2

نقوم بتعيين المصفوفة وذلك بطريقتين:

1. في لائحة التعيين في var نكتب اسم المتغيرة (المصفوفة) تليها نقطتان تليهما كلمة array بعدها طول المصفوفة أي عدد القيم التي يمكن أن تحتويها تليها الكلمة of تليها نوع القيم, مثال على ذلك:

```
Var
   masfufa : array[1..4] of integer ;
```

عدد القيم هو مجال يبدأ من واحد وينتهي بآخر رتبة.

2. يمكن استعمال الكلمة المفتاح type حيث يتم وضعها قبل لائحة التعيين var ونكتب فيها اسما للجدول الذي نود تعيينه في المصفوفة يليه علامة تساوي و array يليها عدد خانات الجدول يليها كلمة of يليها نوع القيم وننهي التعليمة بالنقطة والفاصلة ومن ثمة في لائحة التعيين نكتب اسم المتغيرة يليها نقطتان وبعدها اسم الجدول ومن بعد ذلك نقطة وفاصلة, مثال على ذلك:

```
Type
   table = array[1..4] of integer ;
var
   masfufa : table ;
```

بالنسبة الى التعامل مع المصفوفة فهو شبيه تماما بالتعامل مع المتغيرات التي تحمل قيمة واحدة وذلك بكتابة اسم المتغيرة يليها معقوفين بينهما رتبة القيمة التي نود التعامل معنا, أمثلة على ذلك:

```
readln(masfufa[1]);
writeln(masfufa[2]);
masfufa[3] := 12;
```

انتهى الدرس

الحلقات التكرارية

الحلقات التكرارية نستعملها لتكرار مجموعة من التعليمات لعدد معيّن من التكرارات وهي نوعان نوع يقوم بتكرار التعليمات عدد معيّن من التكرارات والآخر عدد التكرار لا يتوقف إلا بعد أن يتحقق شرط توقف التكرار.

الحلقة الأولى

هي حلقة for تقوم بتكرير مجموعة من التعليمات لعدد معيّن من التعليمات, لاستعمالها شرط أساسي أن نعيّن متغيّرة خاصة بها وعادة تسمّى بـ i (مظهر تقليد عند المبرمجين) , نقوم بكتابة الكلمة المفتاح for تليها المتغيرة i تليها نقطتان وعلامة تساوي والعدد التي ستبدأ به تليها الكلمة to تليها العدد الذي سيتوقف عنده التكرار تليها do تليها الـ begin و ; end, مثال على ذلك:

```
for i := 1 to 5 do
begin
  writeln('Hello, i is equal ', i);
end;
```

قم بكتابة هذا في برنامج بإسم lesson5_part1 وقم بتشغيله وستلاحظ أنه قد كتب هذا:

```
Hello, i is equal 1
Hello, i is equal 2
Hello, i is equal 3
Hello, i is equal 4
Hello, i is equal 5
```

قامت تعليمة الكتابة في كل مرة بكتابة "Hello, i is equal" يليها قيمة المتغيرة i والتي قيمتها تغيّرت خمسة مرّات في كل مرّة يضيف إلى قيمتها رقم واحد:

```
i := i + 1 ;
```

وهذه تعني أن قيمة i تساوي قيمة i السابقة زائدة واحد.

بالإمكان أن يبدأ التكرار من 2 إلى 5 مثلا ولمعرفة عدد مرات التكرار بالإمكان استعمال هذه الطريقة:

العدد الذي يتوقف عنده التكرار (5) نطرح منه عدد بداية التكرار (2) نضيف إليه رقم واحد (1) فتكون العملية هكذا:

5 – 2 + 1 = 3 + 1 = 4

تعرّفنا في الدرس السابق على المصفوفة وبالإمكان صنع برنامج يقوم بإسناد قيم إلى خلايا إلى هذه المصفوفة فبدلا من استعمال تعليمات readln عديدة, بالإمكان استعمال تعليمة readln واحدة بحيث تتغير قيمة i في كل مرّة وهي تمثّل رتبة الخلية في المصفوفة, مثال على ذلك, لدينا مصفوفة عدد خلاياها أربعة بدلا من فعل هذا:

```
readln(masfufa[1]);
readln(masfufa[2]);
readln(masfufa[3]);
readln(masfufa[4]);
```

بالإمكان استعمال هذه الطريقة الأكثر اختصارا:

```
for i := 1 to 4 do
begin
   reaedln(masfufa[i]);
end;
```

هنا قمنا باستعمال تعليمة واحدة ولذلك بإمكاننا أن نستغني عن الـ begin و end;

الحلقة الثانية

وهي حلقة لا تتوقف على تكرير التعليمات إلا عندما يصبح الشرط الذي تحمله خاطئا وهي تعليمة while ولنستعملها نقوم بكتابة الكلمة المفتاح while يليها الشرط ويليه كلمة do تليها begin و end; نضع بينهما التعليمات التي سيكرّرها في حالة ما كان الشرط صحيحا, مثال على ذلك:

```
while (number < 0) do
begin
  writeln('The number must be positive, please write it again !');
  readln(number);
end;
```

الحلقة الثانية

وهي حلقة عكس الحلقة الثانية لا تتوقف عن التكرار إلا عندما يصبح الشرط الذي تحمله صحيحا وهي تعليمة repeat ولنستعملها نقوم بكتابة الكلمة المفتاح repeat تليها مجموعة التعليمات (لا نستعمل begin و end;) تليها كلمة until يليها الشرط بين قوسين يليه نقطة وفاصلة, مثال على ذلك:

```
repeat
   writeln('Please write a number between 0 and 100');
   readln(number);
until (number in [0..100]);
```

انتهى الدرس

الأساليب والوظائف

في هذا الدرس سنتعلم كيفية صنع برامج صغيرة داخل برنامج واحد وذلك بجمع مجموعة من التعليمات ومن ثمّة استدعائها ليتم تنفيذها وهناك نوعان:

1. الأساليب: وهي تقوم فقط بتنفيذ مجموعة من التعليمات
2. الوظائف: وهي تقوم بإسناد قيمة إلى متغيرة وهذه القيمة تستخرجها بعد القيام بمجموعة من التعليمات (معادلة رياضية مثلا)

الأساليب

لتعيين أسلوب نستعمل الكلمة المفتاح procedure يليها اسم الأسلوب يليها نقطة وفاصلة تليها begin و end; بينهما مجموعة التعليمات التي ستنفّذ, مثال على ذلك:

```
procedure sayHello;
var
   word : string;
begin
   word := 'Hello';
   writeln(word);
end;
```

المتغيرات قمت بتعيينها قبل كلمة begin.

الأسلوب يجب أن يكون قبل البرنامج الرئيسي الذي سينفذ أي أن يكون مباشرة بعد

```
uses wincrt;
```

ولاستدعاء الأسلوب نكتفي بكتابة اسمه يليه نقطة وفاصلة.

مثال على ذلك:

```
program lesson6;
uses wincrt;
procedure sayHello;
var
   word : string;
begin
   word := 'Hello';
```

```
    writeln(word);
end;

begin
    sayHello;
end.
```

تعليمة writeln أيضا هي أسلوب ولكن الفرق بينها وبين هذا الأسلوب هو في القوسين والإعدادات, مثال على كيفية صنع أسلوب به إعدادات وعملية الإستدعاء:

```
program lesson6_part2;
uses wincrt;
procedure sayHello(name : string);
begin
    writeln('Hello Mr. ', name);
end;
var
    theName : string;
begin
    writeln('Please write your name');
    readln(theName);
    sayHello(theName);
end.
```

هنا قام البرنامج باستدعاء تعليمة sayHello وأسند إليها قيمة المتغيرة theName في الإعدادات.

إذا كانت الإعدادات أكثر من واحد فيجب فصلهم بنقطة وفاصلة, مثال على ذلك:

```
procedure addition(a : integer; b : integer; var c : integer);
begin
    c := a + b;
end;
```

هنا قمت بفصل الإعدادات بالنقطة والفاصلة واستعملت الكلمة المفتاح var وهي تستعمل لتغيير قيمة المتغيرة التي وضعت في الإعدادات أي لو نقوم بفعل هذا:

```
begin
   num1 := 4;
   num2 := 5;
   addition(num1, num2, num3);
   writeln(num3);
end.
```

لو نقوم بفعل هذا فالمتغيرة num3 لن يقوم الأسلوب بأخذ قيمتها مثلما فعل مع باقي المتغيرات بل سيقوم يتغير قيمتها وفي هذا المثال قيمتها ستكون مساوي لـ 9.

الوظائف

الوظائف تجمع ميزات المتغيرات والأساليب فلكل وظيفة نوع وأنواعها هي نفس أنواع المتغيرات والوظيفة تقوم بتنفيذ جملة من التعليمات ونستعملها لإسناد قيمة إلى متغيرة ولكتابة وظيفة نستعمل الكلمة المفتاح function يليها الاسم يليه الإعدادات تليها نقطتان يليهما نوع الوظيفة ويليه نقطة وفاصلة تليهم begin و end;, مثال على ذلك:

```
function absolute_value(number : real) : real;
begin
   if (number < 0) then
      absolute_value := number * (-1)
   else
      absolute_value := number;
end;
```

هذه الوظيفة تعطينا القيمة المطلقة للعدد الذي نكتبه في إعداداتها والقيمة التي ترجعها هي من نوع real ولإسناد القيمة نكتب اسم الوظيفة يليه نقطتان وعلامة تساوي تليها القيمة وهي في هذه المثال القيمة المرسلة في الإعدادات مضروبة في ناقص واحد ان كان العدد المرسل في الإعدادات سالبا وإن كان الشرط خاطئا فالقيمة تبقى مثلما هي.

انتهى الدرس

الفهرس