3.4 عبارة التكرار REPEAT -- UNTIL LOOP

يستخدم هذا الأمر لتكرار عباره أو أكثر لعدد من المرات وفقا لمتطلبات البرنامج والتي يحددها المبرمج , في هذا الأمر فان البرنامج سينفذ على الأقل لمره واحده .. ويكون توقف البرنامج أعتمادا على شرط يوضع بعد (until).

التكرار يبدأ بالأمر (أعد أو كرر) (Repeat) ثم مجموعه من الأيعازات المطلوب تكرارها وتنتهي بالأمر (لغاية) (until) الذي يكون بعده شرط (أي لغاية تحقق هذا الشرط) , المترجم حين يجد العباره (أعد) فأنه يعلم أن المطلوب أعادة تنفيذ العبارات المحصوره بين هذا الأمر والأمر (لغاية).. في كل مره يصل المترجم الى الأمر (لغاية until) يفحص الشرط الذي بعده فأذا كان الشرط غير متحقق (أجابته false) فأن المترجم سيعود الى الأمر (repeat) ويبدأ بالتنفيذ نزولا من جديد , هذه العمليه تستمر لغاية أن يتحقق الشرط وتكون أجابته (true) . الصيغه القواعديه لهذا الأيعاز هي :

```
Repeat
   Instruction 1 ;
   Instruction 2 ;
   Etc…
Until (condition is true ) ;
```

مثال :// هذا برنامج بسيط واجبه أدخال أسماء الطلبه وطباعتها , البرنامج لا يتوقف لغاية أدخال أسم (علي) .

```
Program CH3_Program1;

Var YN: String;
Begin
  Writeln ('enter name of students?');
  Readln (YN);
  If (YN <> ' Ali') then
    Writeln (YN);
  Readln (YN);
  If (YN <> ' Ali ') then
    Writeln (YN);
  Readln (YN);
  If   (YN <>  ' Ali ') then
    Writeln (YN);
  Readln (YN);
  If (YN <> ' Ali') then
    Writeln (YN);
  Readln (YN);
  If (YN <> ' Ali ') then
    Writeln (YN);
  Readln (YN);
  If   (YN <>  ' Ali ') then
    Writeln (YN)
     …
     …
```

هذا البرنامج ممكن أن يستمر بعدد كبير من الخطوات المتشابه وحسب عدد الطلبه المراد طباعة أسمائهم , أن العبارات (اقرأ , أذا , وأكتب) تتكرر بأستمرار في البرنامج أعلاه , لذا فان لغة البرمجه باسكال أوجدت البديل الذي يسهل العمل ويختصر عدد الخطوات ألا وهو عبارات التكرار . واحد من هذه الأوامر هو (repeat) وأذا ما أعدنا كتابة البرنامج أعلاه ولكن مع أستخدام (repeat) , سينتج لنا البرنامج التالي :

```
Program CH3_Program2;

Var YN: String;
Begin
  Writeln ('enter name of students?');
  Repeat          {repeat the code for at least one time}
    Readln ( YN );
    Writeln ( YN ) ;
  Until (YN = 'Ali');
End.
```

ميزة هذا الأمر أن الشرط هو في نهاية التكرار ولذا فأنه سينفذ ولو لمره واحده قبل أن يتم فحص الشرط . أرجو ملاحظة كيف أن البرنامج أصبح أكثر أختصارا وأسهل للمتابعه .

3.5 عبارة التكرار WHILE - DO LOOP

وهو أيضا من أيعازات التكرار وهو يشابه ألى درجه كبيره الأيعاز (Repeat_until) حيث أن واجب الأيعازين هو التكرار لمرات غير محدده ابتداءا وأنما يعتمدان على تحقق شرط معين لأيقاف التكرار , الصيغه القواعديه لهذا التكرار هي :

While *<condition is true>* ***do*** *the following:*

instruction 1;

instruction 2;

instruction 3;

etc...

End; {If while-do loop starts with a begin statement}

ماذا يعني هذا الأمر (عندما يتحقق الشرط أعمل ما يلي) وفي كل مره سينفذ الأيعاز الذي بعده مباشرة ويعود الى (while) ليفحص الشرط هل هو متحقق أم لا فأذا كان متحقق ينفذ وأن كان غير متحقق سيهمل الأيعاز الذي بعد (while) وينفذ ما بعده .

ملاحظه ://:

كما هو الحال في (If and else) فان الأمر (while) ينفذ عباره واحده فقط والتي تأتي بعده مباشره , أما أذا كان هناك أكثر من عباره واحده مطلوب تكرارها ضمن الامر (while) فيجب أن تحدد بين الأمر (begin) والأمر (end)

اذن ما هو الفرق بين (While) و (Repeat) .. لاحظ الجدول (3.1) :

جدول (3.1) : الفارق بين أمري التكرار (repeat , while)

While_Do	Repeat_Until
الشرط في بداية التكرار	الشرط في نهاية التكرار
لا ينفذ أي أيعاز مالم يتم فحص الشرط وتحققه	سيتم تنفيذ الأيعازات المشموله بالتكرار على الاقل مره واحده قبل أن يتم فحص الشرط
تنفذ الأيعازات المشموله بالتكرار عند تحقق الشرط	يتوقف التنفيذ عند تحقق الشرط
يستخدم مع طلبات التكرار غير المحدده بعدد ثابت من التكرارات مسبقا	يستخدم مع طلبات التكرار غير المحدده بعدد ثابت من التكرارات مسبقا
يعتمد أستمرار التنفيذ على تحقق الشرط ويتوقف التنفيذ عند عدم تحقق الشرط	يعتمد أستمرار التنفيذ على عدم تحقق الشرط ويتوقف التنفيذ عند تحقق الشرط

مثال :// مثال بسيط لأدخال مجموعة أرقام وطباعتها بشرط يتم التوقف عند أدخال الرقم (0) .

```
Program CH3_Program3;

Var x : integer;
Begin
  Writeln ('Enter number');
  Readln(x);
  While x <> 0 do
    Begin
      Writeln(x);
      Readln(x);
    End;
End.
```

شرح البرنامج ://

المطلوب من البرنامج أدخال مجموعة أرقام بشرط أن يتوقف عند أدخال الرقم (0) , أذن لما كان أدخال مجموعة أرقام فهذا يعني أننا سنكرر أمر الأدخال أكثر من مره وفي كل مره يجب فحص الرقم لغرض طباعته أذا لم يكن يساوي (0) هذه العمليه ممكن تكرارها 5 مرات 10 مرات 1000 مره أو أكثر حسب طبيعة العمل (تصوروا برنامج يتكون من هذا الكم الهائل من الخطوات المتشابهه !!) لذلك لتجنب هذه العمليه تم أيجاد أيعازات التكرار فيمكن هنا أن نستخدم الأمر (While) لأختصار البرنامج , هذا الأمر يحتاج الى شرط لغرض العمل والتوقف , في هذا المثال البرنامج يتوقف عند ورود الرقم (0) أي أنه يعمل مع الأرقام الأخرى ولما كان الشرط يجب أن يكون (true) لكي يعمل أذن أي رقم لايساوي صفر سوف يجعل البرنامج يعمل لذا جعلنا (x <> 0) , لقد سبق وأن بينا أن المترجم عندما يصل الى أي خطوه فيها متغير سيقوم بعملين الأول يتأكد من تعريف المتغير في حقل المتغيرات والثاني يتأكد من أن المتغير له قيمه وحسب النوع المعلن عنه في حقل المتغيرات . لذا فأنه عندما يصل المترجم الى الأمر (While) يجب أن يجد قيمه للمتغير (x) وهذا هو السبب الذي جعلنا ندخل قيمه للمتغير (x) قبل الأمر (While) وأن لم نقم بذلك فأن البرنامج سيفشل لعدم وجود قيمه للمتغير (x) . كذلك لما كانت هناك أكثر من خطوه مشموله بالتكرار والتي هي الطباعه والقراءه عليه تم تحديدهما بين (begin and end) .

ملاحظه ://

في كل مره يتم قراءة قيمه جديده للمتغير (x) فأن القيمه السابقه ستزول وتحل محلها القيمه الجديده وهذه قاعده عامه يجب أن تلاحظ .

ملاحظه ://

يتم أختيار الشرط بعد الأمر (While) بحيث يساعد حلقة التكرار أن تستمر طالما كان هذا الشرط متحقق , وأن تتوقف الحلقه عن التكرار عندما لا يتحقق هذا الشرط .
في حالة الأمر (Repeat) فأن الشرط ياتي بعد (Until) لذا يجب أن يتم أختياره بحيث عندما يتم فحصه تكون النتيجه (False) أي غير متحقق , لكي يستمر التكرار بالعمل ومتى ما أصبحت نتيجة فحص الشرط (True) فأن التكرار يتوقف .

ملاحظه ://

من السهل كتابة حلقه بشكل عفوي , شرطها لا يصبح متحققا أبدا , هذا سيؤدي الى برنامج مقفل أو مغلق أي يستمر بالتنفيذ الى مالانهايه .