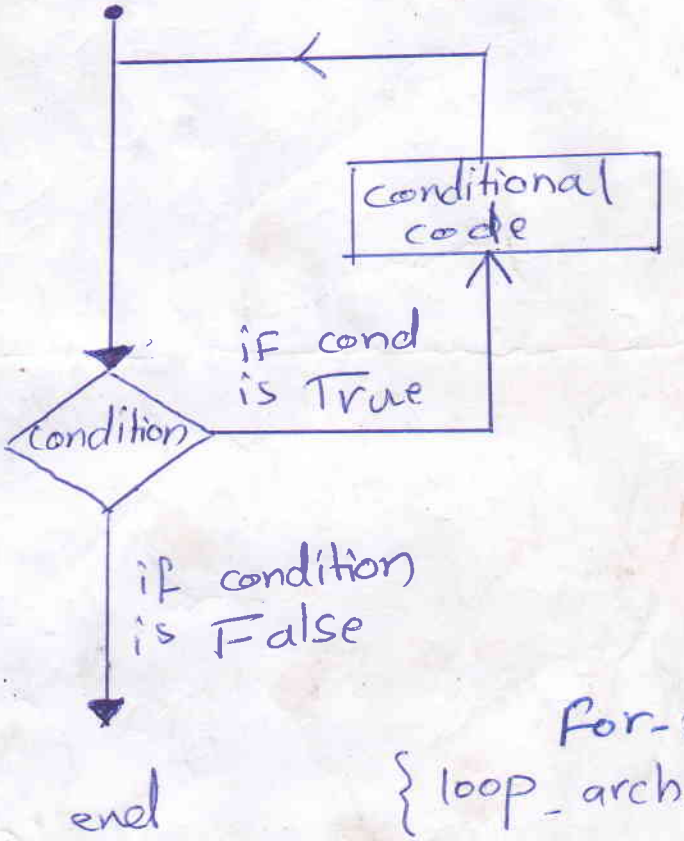
lecture 4

"loops"

- we use the loop when we need to execute ablock of code several times.

in general statements are executed sequentially the First statement in a function is executed First, Followed by Second statement and so on



for-Do
{ loop_architecture }

* types of loops.
1- For-do
2- while-do
3- Repeat-until
4- nested-loop.

do loop :- the statement inside For loop executed a number of time depending on the control condition.

Syntax:

① For var_name := initial value To Final value Do
statement;

② For var_name := initial value DownTo Final value Do
statement;

ex:① Program ss;
var
a: integer;
begin
For a:= 10 to 20 do
begin
writeln(value of a:',a);
end;
end.

ex:②

Program aa;
var
a: integer;
begin
For a:= 20 downto 10 do
begin
writeln('value of a:',a);
end;
end.

ملاحظة: الجملة التي بعد الـ do مباشرة هي التي تكون تابعة للتكرار.
بمعنى هي التي تحتوي للتكرار عدد المرات. حسب المدى المعطى لجملة الـ for .

the type of variable in the loop must be the same as the type of initial and final values.

For example, Considering that the variable I of the integer type. decide whether the following Examples are True or False?

→ For i := 1 to 's' do
→ For i := 'a' to 'z' do
→ For i := 10 to 1 do
→ For i = 5 downto 10 do
{the final value is not the same data type of i.} القيمة النهائية ليست من نفس نوع i.
{initial and final value is different data type of the i} القيمة الابتدائية والنهائية ليست من نفس نوع i

Flow of Control in a For-Do loop.

- the initial step is executed First, and only once. This step is ~~execut~~ allow you to declare and initialize any loop Control variables.
- Next the Condition is evaluated, if it is True, the body of the loop is executed. if it is False, the body of the loop does not execute and Flow of Control jumps to the next statement just after the loop body.
- After the the body of the For-Do loop executes, the value of the variable is either increased or decreased.
- the Condition now evaluated again. if it is T, the loop executes and the process repeats itself (body of loop, then increment step, and then again Condition) after the Condition become False, the For-Do loop terminates.

While - do loop :- this statement in Pascal allows repetitive computations till some test condition is satisfied.

syntax:

```
while (condition) do S;
```

(S: statement)

where condition is Boolean or relational expression, whose value would be true or False and S a simple statement or group of statements within Begin ... end block.

ex:

```
while number > 0 do
  begin
    sum := sum + number;
    number := number - 2;
  end;
```

when the condition become False, Program control passes to the line immediately following the loop.

(2) while - Do loop

At first the Condition is checked is it true or not? if the Condition is True, the loop will take the first value and execute the block of code till the end of the block.
after that the Condition will checked again and if it is True the block of code will executed again and so on, {if the condition evaluated to false the flow of control jumps ~~to the~~ just after the loop body, and executes the stmts after the loop body.

- program for printing the number from 1 to 5.

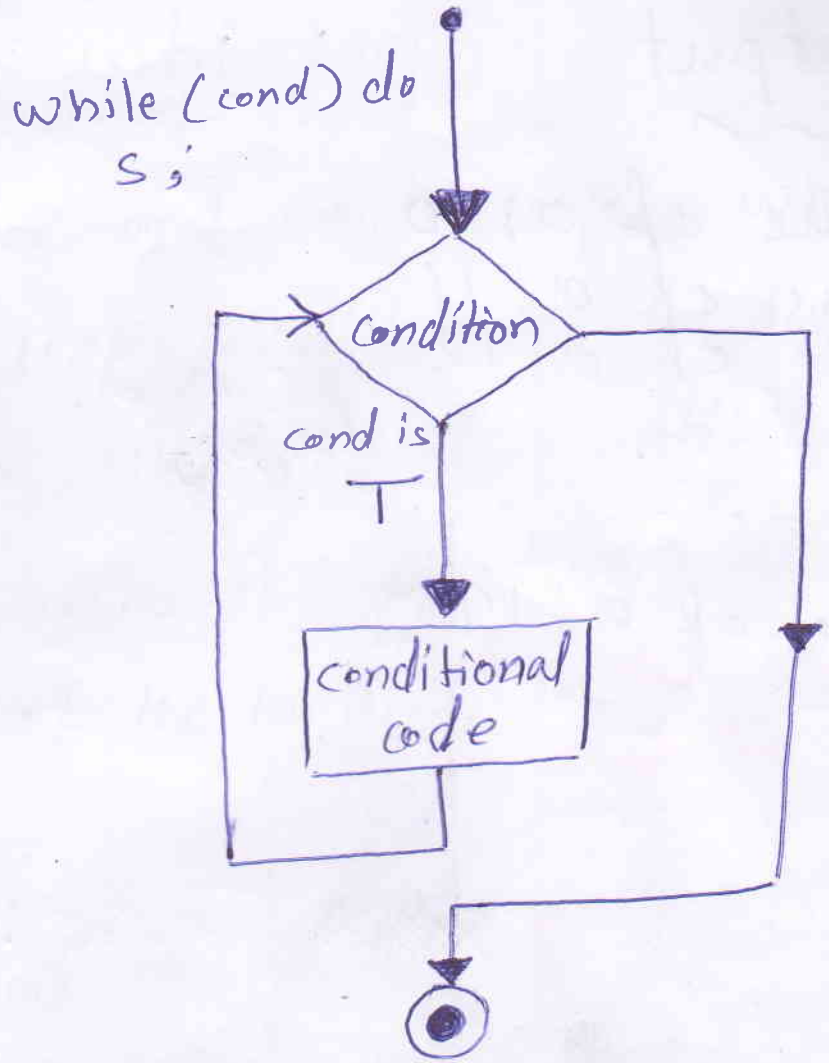
```
program print;
var
  i: integer;
begin
    i:=1;
    while i<6 Do
  begin
      writeln (i);
      i:= i+1;
  end;
End.
```

*Explain this program.

after declaring the variable I as an integer value, we have set the initial value of the variable. it must be set, so that the Compile will know the begining value. otherwise if will set the default value is 0.

er that we start with the loop
we set the condition as i < 6
means, if the value of i is le
than 6 the block of code is execu
and if the i is equal or greater
than 6 the block of code does no
execute and the control of progra
is terminates.

→ if the condition is true, the value
of i will be printed and then
execute the the statement i:=i+1.
means the previous value of i i
1 +1, it will be 2 . in the (i
this process is repeated till the
condition become false.




```
Program while;
var
   a: integer;
begin
   a:= 10;
   while a<20 do
   begin
      writeln ('value of a: ', a);
      a:= a+1;
   end;
end.
```

output →

The output

```
value of a: 10
value of a: 11
value of a: 12
- - - -
- - - - -
value of a: 19
```

Repeat-until Loop

unlike for and while loops, which test the loop condition at the top of the loop, the repeat--until loop in Pascal checks its condition at the bottom of the loop. {in the repeat no need to begin-end

Syntax:-

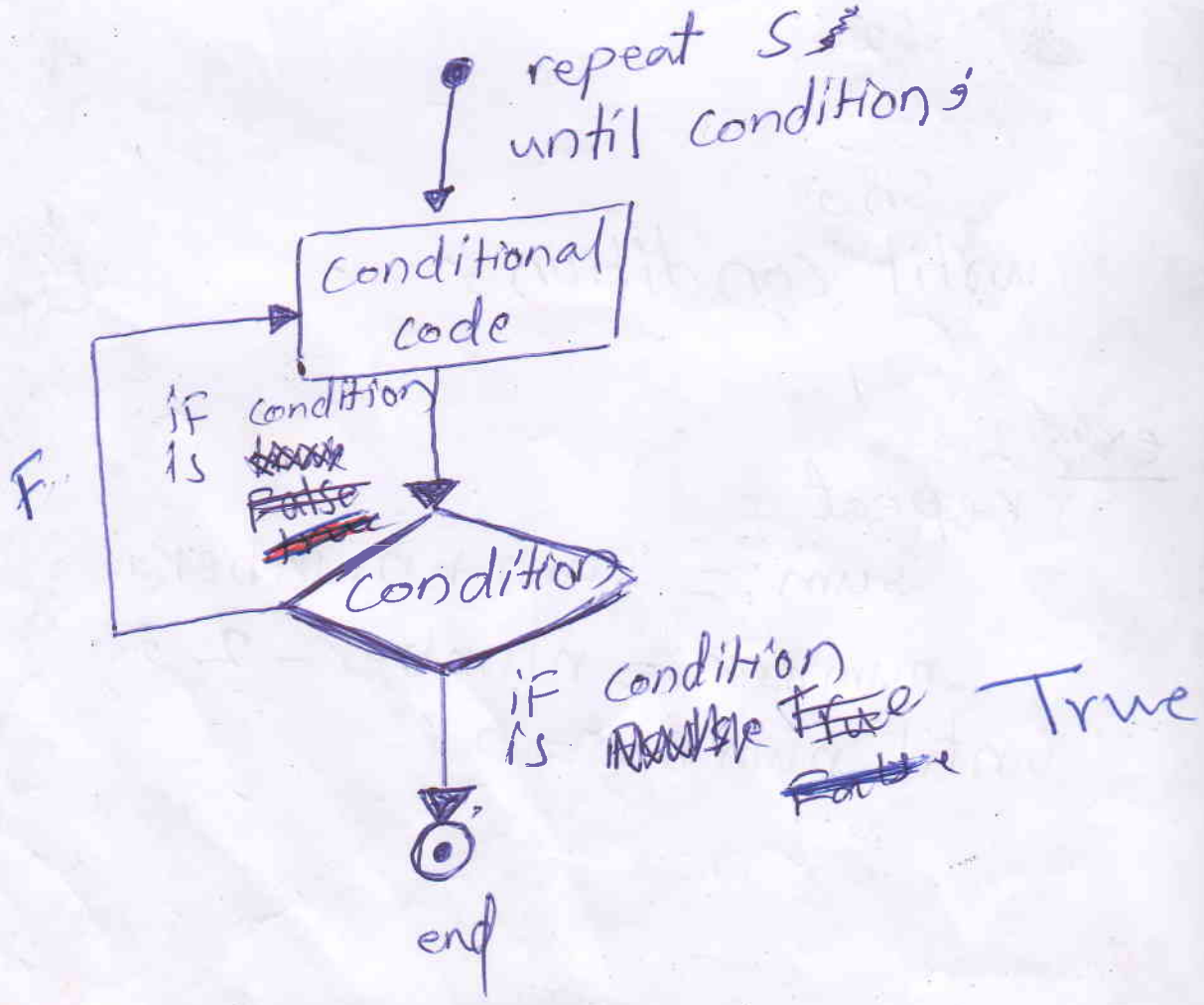
```
repeat
    S1;
    S2;
    ...
    ...
    Sn;
until condition;
```

ex:

```
repeat
    sum := sum + number;
    number := number - 2;
until number = 0;
```

notice that the conditional expressio
appears at the end of the loop, so t
statement in the loop execute once before
the condition is tested.

if the condition is ~~false~~ false, the flow of
control jumps back up to repeat and
the statement in the loop execute agai
this process repeats until the given
condition becomes ~~true~~. true.



```
program repeat;
var
  a: integer;
begin
  a:=10;
  repeat
    writeln('value of a:', a);
    a:= a+1
  until a=20;
end.
```

4) Nested loops :- pascal allows using one loop inside another loop.

ex: Find prime numbers from 2 to 50.

```
program prime;
var
  i,j: integer;
begin
  for i:= 2 to 50 do
  begin
  for j:= 2 to i do
  if (i mod j) = 0 then
  if (j=i) then
  writeln (i, 'is prime');
end;
end.
```

ex:1 طباعة الشكل التالي.

```
*
* *
* * *
* * * *
```

```
Program m;
Var
  x, i : integer;
begin
  For i:=1 to 4 do
    begin
      For x:= 1 to i do
        write ('*');
      writeln;
    end;
end.
```

- يجب ان يكون المتغير مطابق لما موجود في جزء الـ var ← هنا حسب نوع المتغيرات

يطبق جملة بعد for أربع مرات
write ('*');
writeln;

output

```
*
**
***
****
```

trace

i=1
* x=1 to 1

i=2
** x=1 to 2

i=3
*** x=1 to 3

i=4
**** x=1 to 4