Ministry of Higher Education and Scientific Research
Djilali BOUNAAMA University - Khemis Miliana(UDBKM)
Faculty of Science and Technology
Department of Mathematics and Computer Science

Chapter 4

Iteration/Repetetive Structures : Loops

MI-L1-UEF121 : Algorithms and Data Structures I

Noureddine AZZOUZA

n.azzouza@univ-dbkm.dz

Course Topics

1. Introduction
2. « for » loop
3. « while » loop
4. « repeat ... until » loop
5. Nested loops

Introduction

Problem

- ✓ Write an algorithm that displays the multiplication table of an integer between 1 and 10

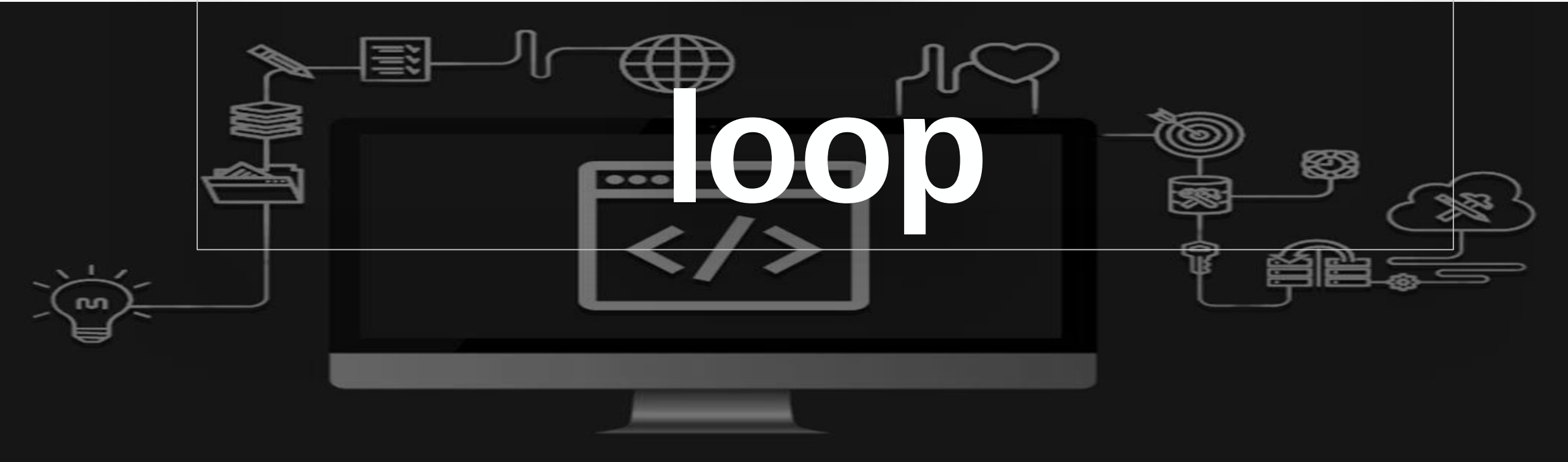
Definition

- ✓ A **repetitive structure** is a structure that repeats the same processing as many times as desired depending on an **execution condition**.
- ✓ A **Repetitive Structure** is also called **Iterative Structure** or a **Loop**.
- ✓ A loop consists of four essential elements :
  - ➢ A **statement block**, which will be executed a certain number of times;
  - ➢ A **condition/test**, which concerns at least one so-called ***loop variable***.
  - ➢ An **initialization** of the ***loop variable***.
  - ➢ An **update/modification** to the ***loop variable*** to stop the loop.

Definition

- ✓ There are 3 forms of **repetitive structures** (loops)
  1. The **FOR** loop
  2. The **WHILE** loop
  3. The **REPEAT** loop
- ✓ These structures have the same power; but by convention the choice of a loop depends essentially on the nature of the problem to be solved
- ✓ **Infinite loop**: is a loop whose condition never changes, causing an infinite number of repetitions.
- ✓ **Iteration**: it is a complete loop cycle including the loop block, the condition test, and also the modification.

for loop

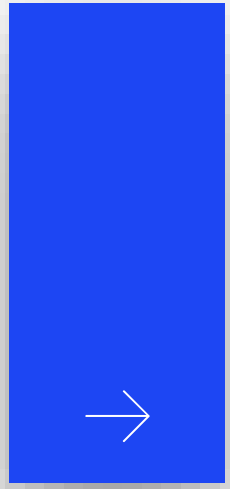


the FOR loop

- ✓ the FOR loop is a repetitive structure that iterates the same processing for a ***range of values*** between a lower bound and an upper bound. the FOR loop is used when the number of repetitions is ***known*** in advance

Syntax

```
for   Loop Variable  :=  initial Value  to  Final value  do

                Bloc

endfor
...
```



Running for loop

1st Stage : **Initialization** of the Loop Variable with the Initial Value.

2nd Stage : Evaluate the **test/condition** and check whether the Loop Variable value is in range or not.

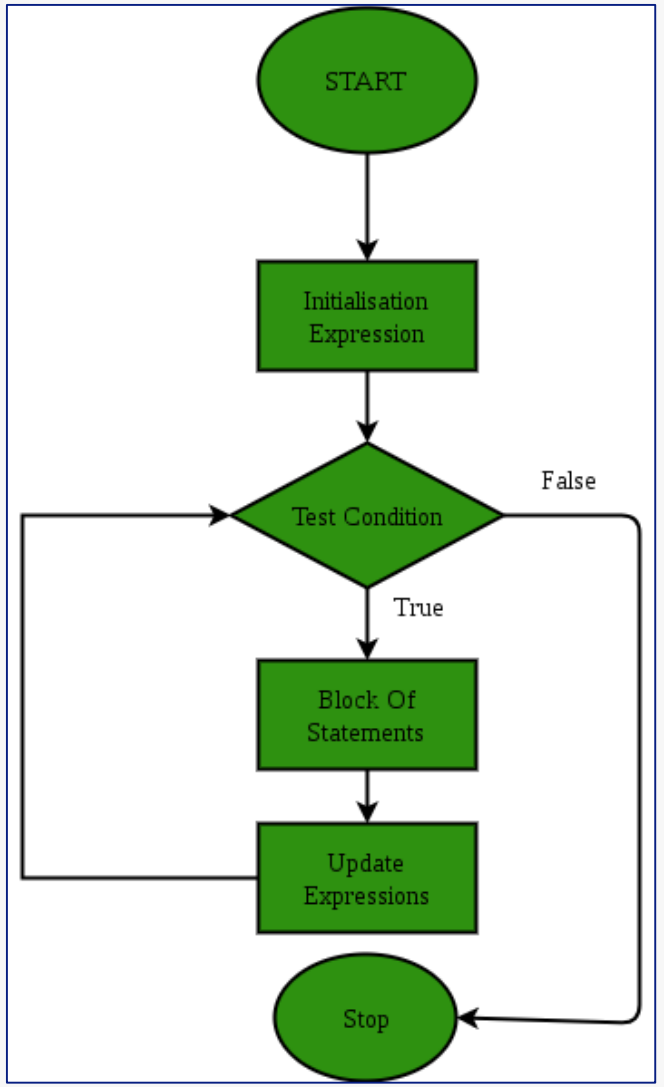
- If it is in the interval then go to the ***3rd Stage***, otherwise go to the ***5th Stage***.

3rd Stage : Execution of the loop **statement block**.

4th Stage : Automatic **update/modification** (Increment) of the value of the Loop Variable according to the value of the Step (default: Step = 1) and return to the ***2nd Stage***.

5th step: Exit from the loop and continue execution of the program starting from the first instruction which comes after ***endfor***.

for loop Flow Diagram



Example1 : Multiplication table of a number

Multiplication table

Write an algorithm which asks the user for an integer N (1 < N < 10), and which then writes the multiplication table of this number

Analyse :

```
Algorithm table_multiple;

Var N, i : integer ;

begin
    read (N);

    for i := 1 to 10 do

        write (N, 'x', i, ' = ', N*i)

    endfor

end.
```

Example2 : Power

Power

Write an algorithm that calculates $\mathbf{a^b}$ such that a and b are two positive integers given by the user.

Analyse :

```
Algorithm  puissance;

Var  a, b : integer ;
     p, i : integer;
begin
    read (a, b);
    p := 1;
    for  i := 1 to b do

        p := p * a;

    endfor

    write ('la puissance =', p)

end.
```

the FOR loop

Declaration

PASCAL

Syntaxe:

FOR compt := val_initial TO val_final DO
Begin ... End

C

Syntaxe:

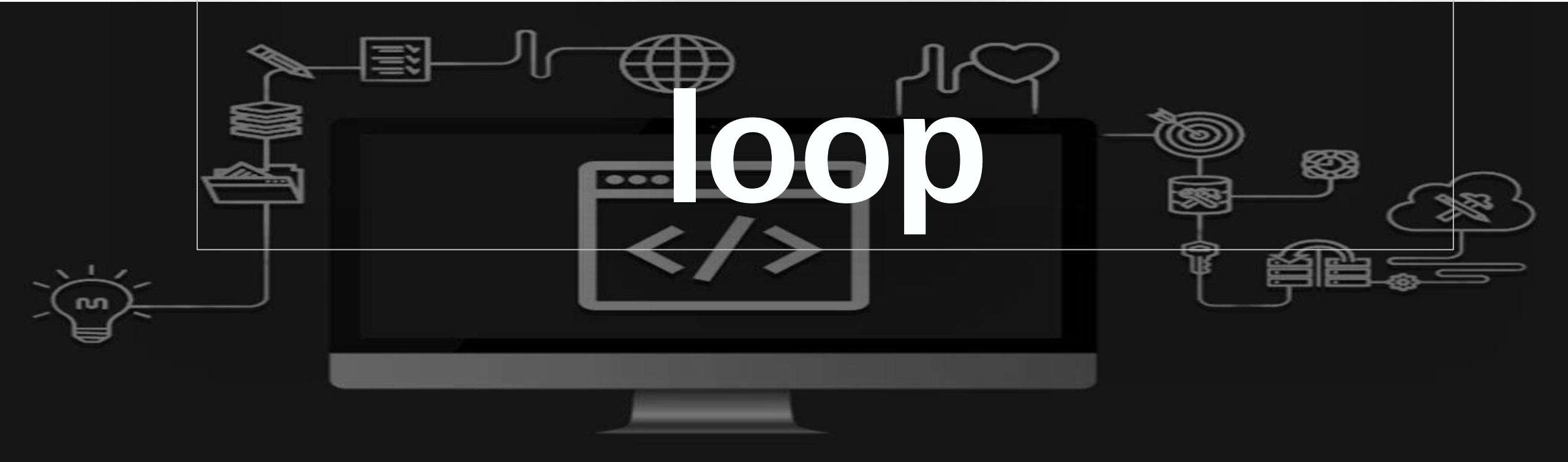
for (initialisation; condition; modification)
{ ... }

Examples

```pascal
program Exemple_pour;

var
        a, b, p, i : Integer;

begin
    ReadLn(a, b);   // lecture
    p := 1;         // initialisation

    For i:= 1 to b do
      begin
        p := p * a;
      end;

    WriteLn('La puissance = ', p);

end.
```

```c
#include <stdio.h>

int main (){

    int a, b, i; // déclaration

    int p = 1; // déclaration + initialisation
    scanf("%d %d", &a, &b);

    for (i = 1; i <= b; i++)
    {
        p = p * a;
    }

    printf("La puissance = %d", p);

    return 0;
}
```

while loop

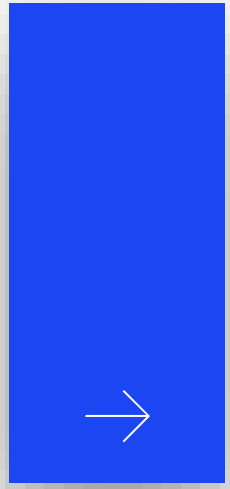


the WHILE loop

- ✓ the WHILE loop executes the body of the loop when the execution ***condition*** is **met**; we will stop as soon as the condition is no longer verified.

Syntax

```
loop variable initialization
while   execution test/condition   do
                    Statements Block
    update loop variable
endwhile
...
```



Running While

1st step: **Initialization** of the Loop Variable before the loop.

Step 2: Evaluate and test the **execution condition**.

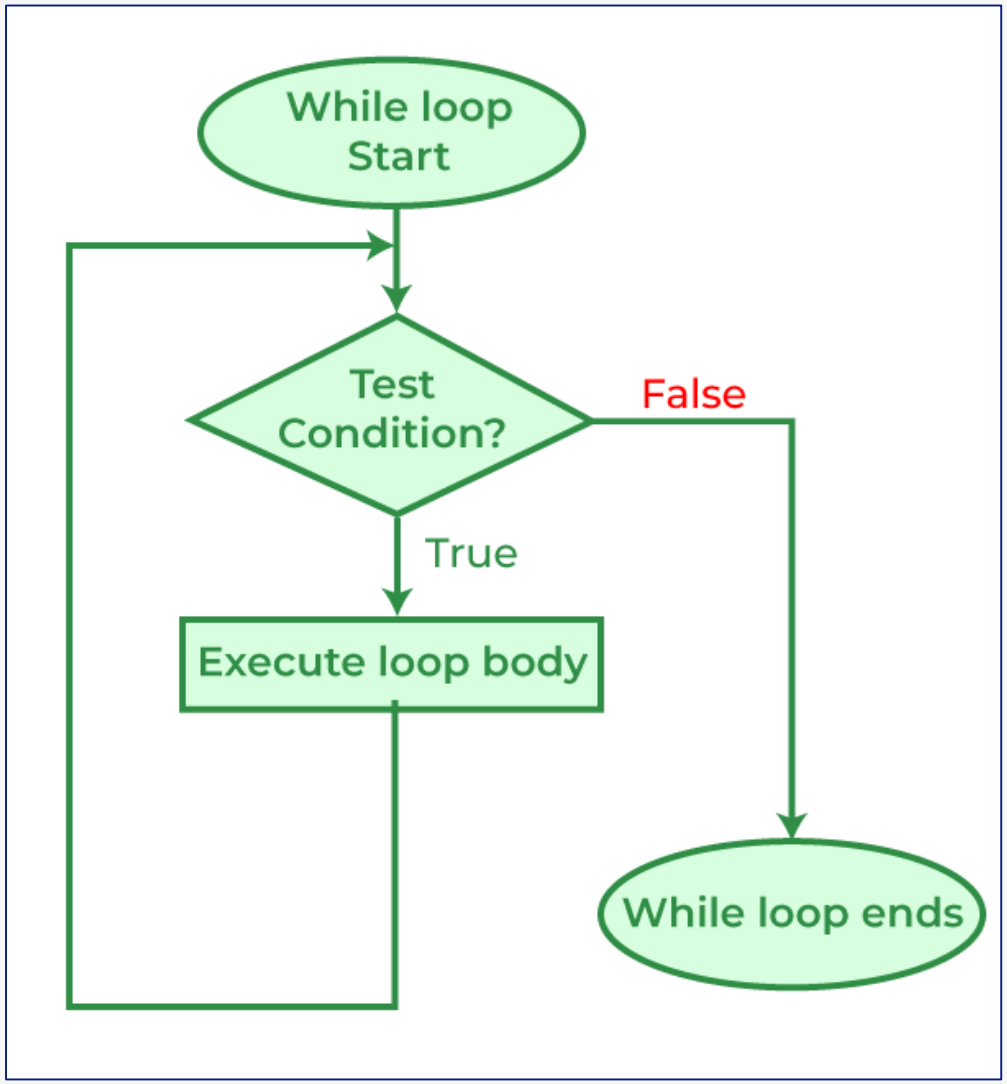
If it is verified then go to the ***3rd step***, otherwise go to the ***5th step***.

3rd step: Execution of the loop **statements block.**

4th step: Changing (**updating**) the value of the Loop Variable and returning to the ***2nd step***.

5th step: Exit from the loop and continue execution of the program starting from the first instruction which comes after ***endwhile***.

while loop Flow Diagram



Example1 : Multiplication table of a number

Multiplication table

Write an algorithm which asks the user for an integer N (1 < N < 10), and which then writes the multiplication table of this number

Analyse :

```
Algorithm table_multiple;

Var N, i, p: integer ;

begin
    read (N);
    i := 1;
    while i <= 10 do
        p := N * i
        write (N, 'x', i, ' = ', p)
        i := i + 1
    endwhile

end.
```

Exemple2 : GCD calculation

GCD calculation

Write an algorithm that calculates the GCD of two integers a and b by applying the recurrence relation GCD(a,b) = GCD(b, a MOD b) until the remainder of the division of a by b is zero .

Analyze :

PGCD(18,12) = PGCD(12, 6) = PGCD(6, 0) = 6

```
Algorithm calcul_pgcd;

Var a, b, r: integer ;

begin
    read (a,b);

    while b <> 0 do
        r := a MOD b
        a := b
        b := r
    endwhile
    write ('Le PCGD de a et b =', a)
end.
```

the WHILE loop

Déclaration

PASCAL

Syntaxe:

WHILE condition **DO**
Begin ... End

C

Syntaxe:

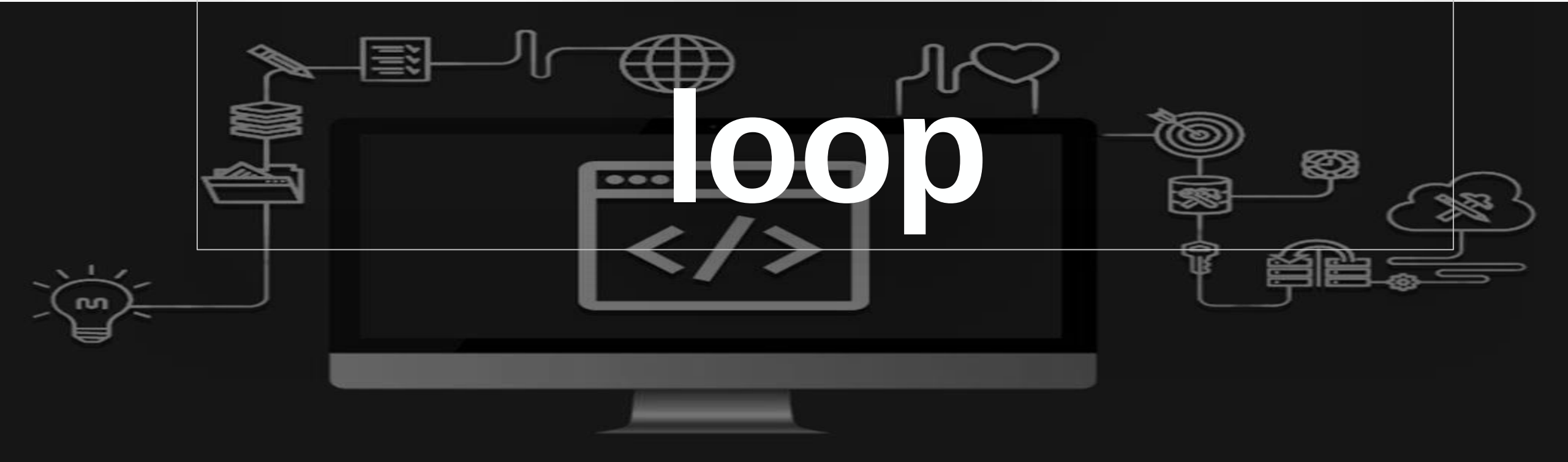
while (condition) { ... }

Exemples

```pascal
program Exemple_tq;

var
        a, b, r : Integer;

begin
    ReadLn(a, b);   // lecture

    While b <> 0 do
      begin
        r := a mod b;
        a := b;
        b := r;
      end;

    WriteLn('Le PGCD de a et b est ', a);

end.
```

```c
#include <stdio.h>

int main (){

    int a, b, r; // déclaration
    scanf("%d %d",&a, &b);

    while (b != 0)
    {
        r = a%b;
        a = b;
        b = r;
    }

    printf("Le PGCD de a et b = %d", a);

    return 0;
}
```

repeat loop



the REPEAT loop

- ✓ La boucle **Répéter** permet de rentrer dans la boucle quelque soit la condition et réitère l'exécution ***jusqu'à ce*** que la ***condition*** soit vérifiée.

Syntax

```
Repeat
        Block of statements
        update loop variable
until  stop condition
...
```

Running REPEAT

Step 1: Go inside the “Repeat” loop and execute the associated **block**.

Step 2: Update the variables involved in the **stopping condition**.

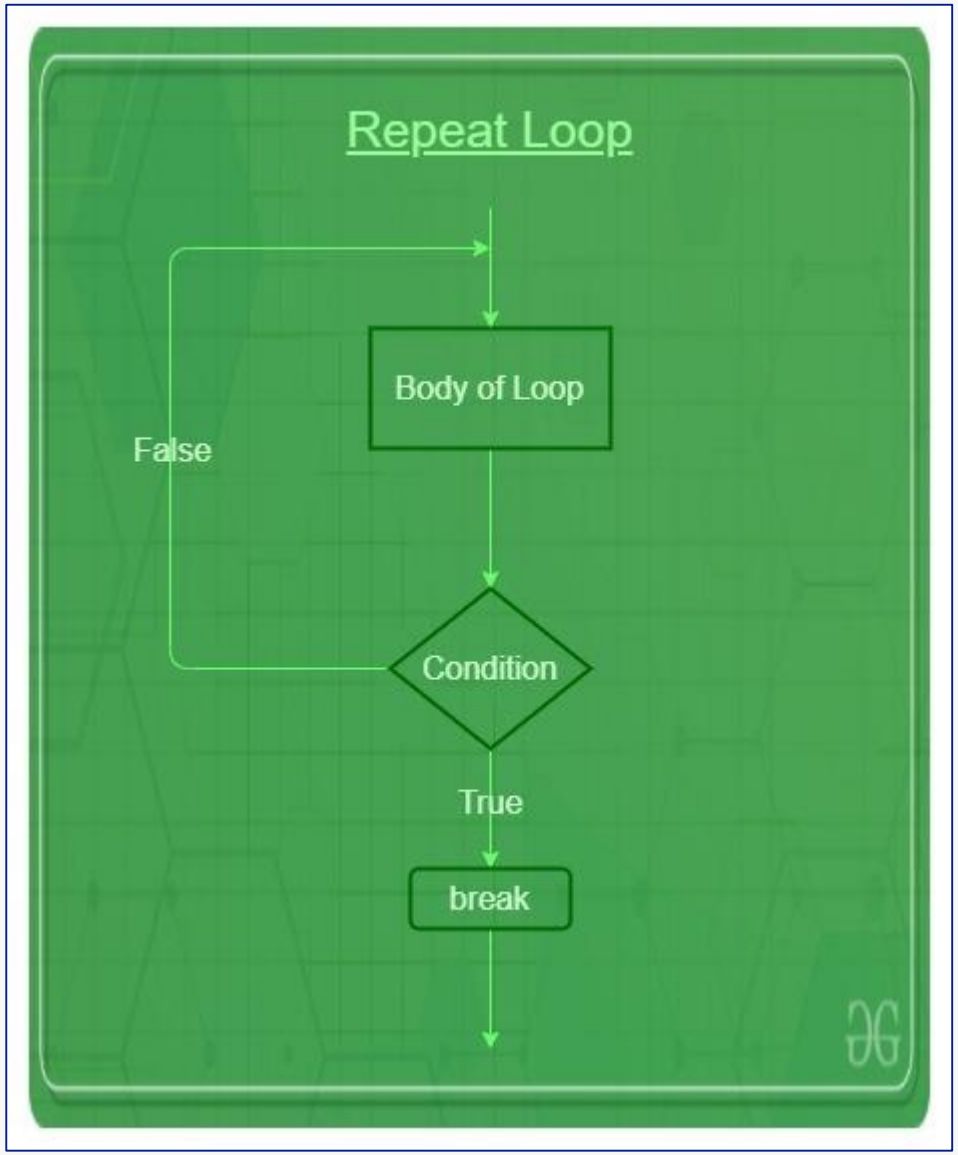
Step 3: Evaluate and test the **stopping condition**.

- If it is checked then go to the ***4th step***, otherwise go back to the ***1st step***.

Step 4 : Changing (**updating**) the value of the Loop Variable and returning to the ***2nd step***.

Step 5 : the **stopping condition** having been reached, we exit the Repeat loop and continue the execution of the program from the first instruction which comes after Until.

REPEAT loop Flow Diagram



Example1 : Multiplication Table

Multiplication table

Write an algorithm which asks the user for an integer N (1 < N < 10), and which then writes the multiplication table of this number

Analyse :

```
Algorithm table_ multiple;

Var  N, i, p: integer ;

begin
    read (N);
     i := 1;
     Repeat
          p := N * i
          write (N, 'x', i, ' = ', p)
          i := i + 1
     Until  (i > 10)

end.
```

Example2 : GCD calculation

GCD calculation

Write an algorithm that calculates the GCD of two integers a and b by applying the recurrence relation GCD(a,b) = GCD(b, a MOD b) until the remainder of the division of a by b is zero .

Analyse :

PGCD(18,12) = PGCD(12, 6) = PGCD(6, 0) = 6

```
Algorithm calcul_pgcd;

Var a, b, r: integer ;

begin
    read (a,b);

    Repeat
        r := a MOD b
        a := b
        b := r
    Until (b = 0)
    write ('Le PCGD de a et b =', a)
end.
```

the REPEAT loop

Déclaration

PASCAL

Syntaxe:

REPEAT bloc **UNTIL** (condition)

C

Syntaxe:

do { ... } **while** (condition)

Exemples

```pascal
program Exemple_repeter;

var
        a, b, r : Integer;

begin
    ReadLn(a, b);   // lecture

    repeat
        r := a mod b;
        a := b;
        b := r;
    until (b = 0);

    WriteLn('Le PGCD de a et b est ', a);

end.
```

```c
#include <stdio.h>

int main (){
    int a, b, r; // déclaration
    scanf("%d %d",&a, &b);

    do
    {
        r = a%b;
        a = b;
        b = r;
    }
    while (b != 0);

    printf("Le PGCD de a et b = %d", a);

    return 0;
}
```

Ministry of Higher Education and Scientific Research
Djilali BOUNAAMA University - Khemis Miliana(UDBKM)
Faculty of Science and Technology
Department of Mathematics and Computer Science

Chapter 4

Iteration/Repetetive Structures : Loops

MI-L1-UEF121 : Algorithms and Data Structures I

Noureddine AZZOUZA

n.azzouza@univ-dbkm.dz