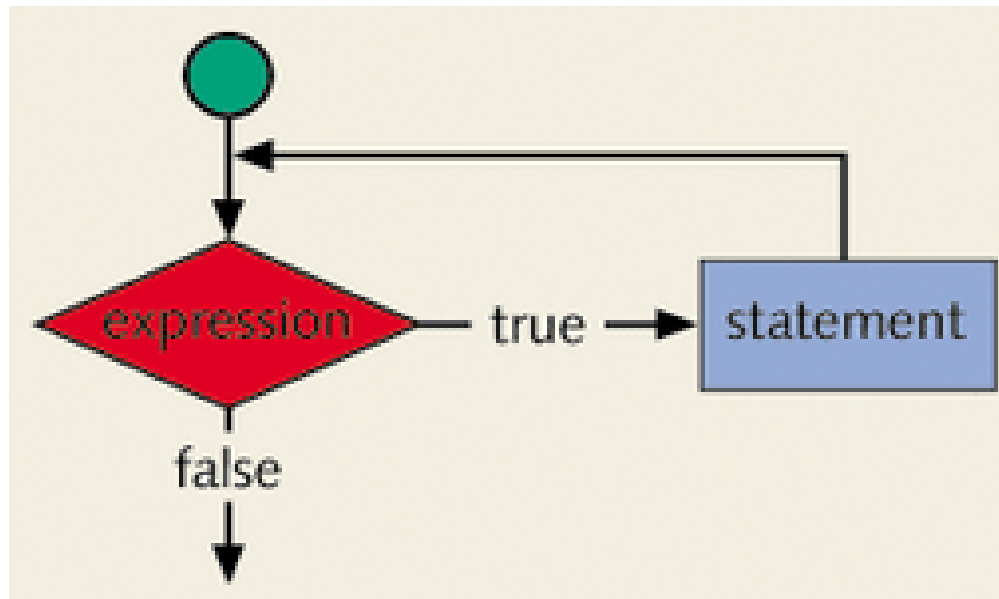# Control Structure/ Iterative /Repetition Structure

By: Atiya Jokhio

Concept of Looping (Repetition) Structure

- One of the basic structured programming concepts
- The real power of computers is in their ability to repeat an operation or a series of operations many times
- When action is repeated many times, the flow is called a loop

Why Is Repetition Needed?

- Suppose we want to add five numbers (say to find their average).
- From what you have learned so far, you could proceed as follows.

  - ```
    scanf(%d %d %d %d  %d,
    &num1,&num2,&num3,&num4,&num5);
    ```
  - ```
    sum = num1+num2+num3+num4+num5;
    ```
  - ```
    average = sum/5;
    ```

- Suppose we wanted to add and average `100`, or `1000`, or more numbers.
- We would have to declare that many variables, and list them again in `scanf` statement, and perhaps, again in the output statement.

Repetition

- There is two means of repetition
  1. Counter-controlled repetition
  2. Sentinel-controlled repetition

- 1. **Counter-controlled repetition:** is sometimes called definite repetition because we know in advance exactly how many times the loop will be executed.

- In counter-controlled repetition, a control variable is used to count the number of repetitions. The control variable is incremented (usually by 1) each time the group of instructions is performed. When the value of the control variable indicates that the correct number of repetitions has been performed, the loop terminates and execution continues with the statement after the repetition statement.

Repetition [Cont.]

2. **<u>Sentinel-controlled repetition</u>**

- Sentinel-controlled repetition is sometimes called indefinite repetition because it's not known in advance how many times the loop will be executed.

- Sentinel values are used to control repetition when:

  - The precise number of repetitions isn't known in advance, and

  - The loop includes statements that obtain data each time the loop is performed.

Repetition [Cont.]

Counter-controlled repetition

```c
sum = 0;

n = 1;

while (n <= 10)

{

sum = sum + n*n;

n = n+ 1;

}
```

Sentinel-controlled repetition

```c
do

{

printf("Input a number.\n");

scanf("%d", &num);

}

while(num>0);
```

Counter-controlled repetition

Counter-controlled repetition requires:

- The name of a control variable (or loop counter).

- The initial value of the control variable.

- The increment(or decrement) by which the control variable is modified each

time through the loop.

- The condition that tests for the final value of the control variable (i.e., whether looping should continue).

Loop Types

- There are three repetition or looping structures in C that lets us repeat statements over and over until certain conditions are met.

  - **while** Looping Structure

  - **for** Looping Structure

  - **do…while** Looping Structure

Loop Types [Cont.]

✓ **<u>While Loop:</u>**

In while loop, a condition is evaluated before processing a body of the loop. If a condition is true then and only then the body of a loop is executed. After the body of a loop is executed then control again goes back at the beginning, and the condition is checked if it is true, the same process is executed until the condition becomes false. Once the condition becomes false, the control goes out of the loop.

In while loop, if the condition is not true, then the body of a loop will not be executed, not even once.

Loop Types [Cont.]

Syntax

The syntax of a while loop is –

initialize loop counter;

    while(**condition**) {
        statement(s);
        loop counter
   }

Loop Types [Cont.]

Example:

```c
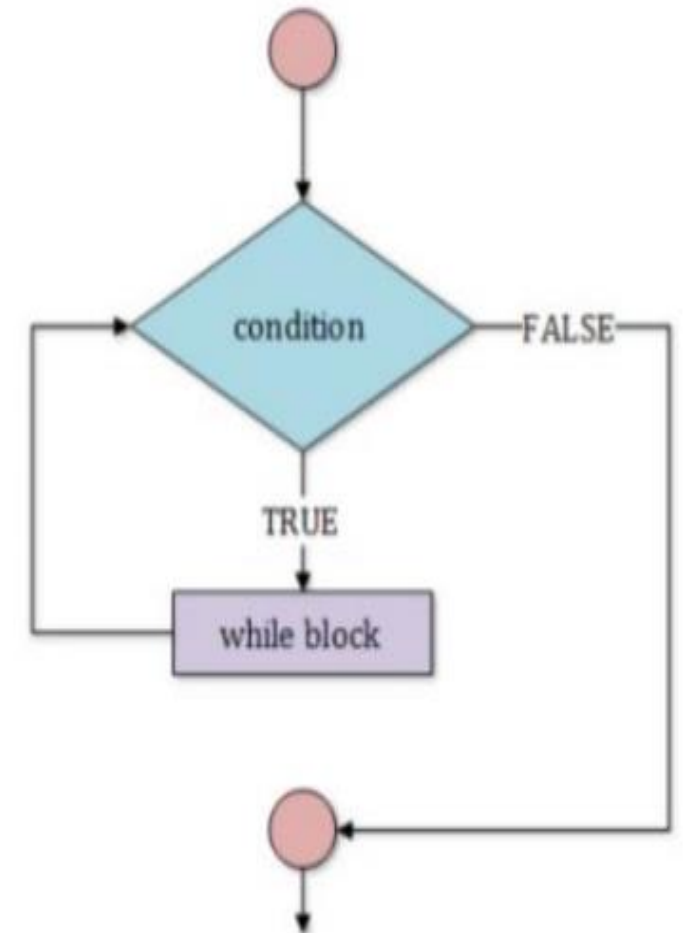#include <stdio.h>

// function main begins program execution
int main( void )
{
    unsigned int counter = 1; // initialization

    while ( counter <= 10 ) { // repetition condition
        printf ( "%u\n", counter ); // display counter
        ++counter; // increment
    } // end while
} // end function main
```