Fundamentals of Programming

Lecture 6

Chamila Karunatilake

Department of Information and Communication Technology

Faculty of Technology

University of Sri Jayewardenepura

chamilakarunatilake@sjp.ac.lk

Looping in C

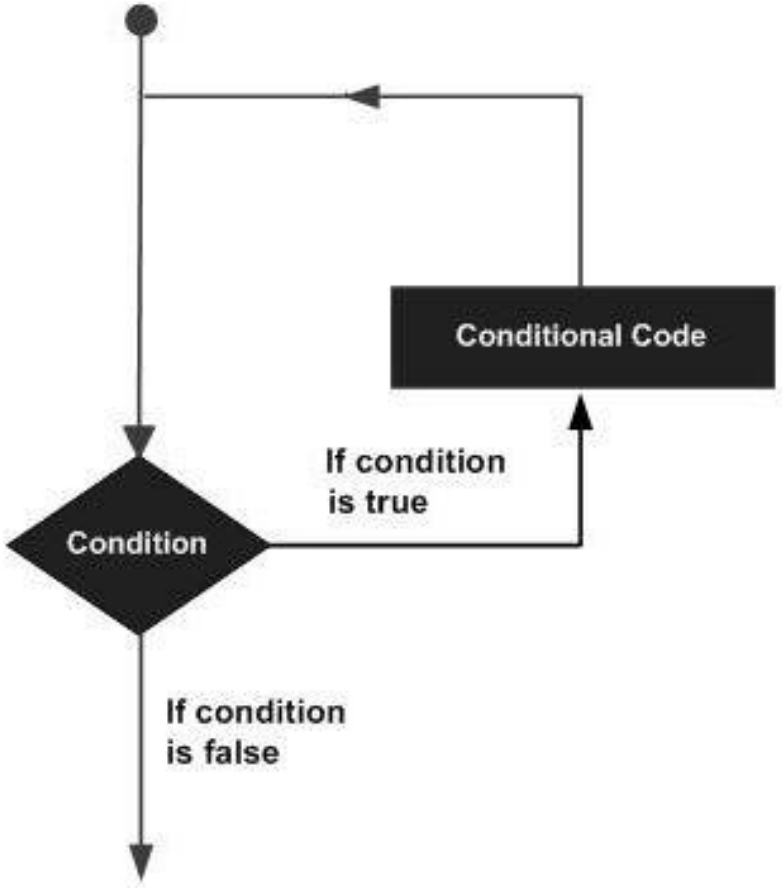
Lecture Outline

- while Loop
- do while Loop
- for Loop

Introduction to Looping in C

- In programming, there are situations, when a block of code needs to be executed several times, repeatedly.
- This is facilitated by C and many other programming languages with a mechanism called **looping**.
- A loop statement allows executing **a statement or group of statements** multiple times.
- There are three types of looping in C:
  - while loop
  - do while loop
  - for loop

Introduction to Looping in C



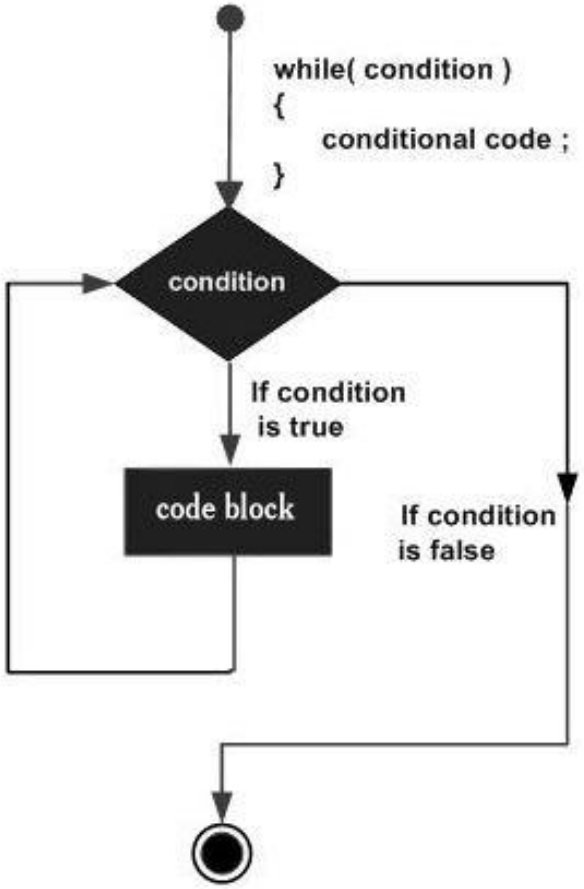
while Loop

- A **while** loop in C programming repeatedly executes a target statement as long as a given **condition is true**(expression is non-zero).
- It evaluates the test expression before every loop, so it can execute zero times if the condition is initially false.

```
while(expression )
{
      statement(s);
}
```

- The expression is evaluated.
- If it is non-zero, statement is executed, and expression is re-evaluated.
- This cycle continues until expression becomes zero, at which point execution resumes after statement.

while Loop




```
int a = 10;

while( a < 20 )
{
      printf("value of a: %d\n", a);
      a++;
 }
```

while Loop : Example

```c
#include <stdio.h>

int main ()
{
    int a = 0;
    while( a < 256 )
    {
      printf("%d\t%c\n",a,a);
      a++;
    }
    return 0;
}
```

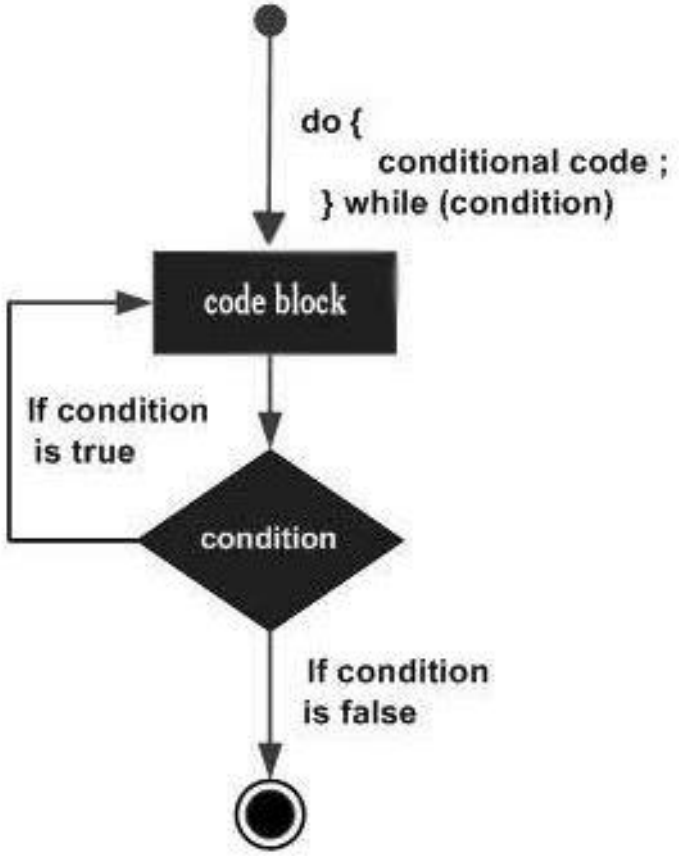
do while Loop

- A **do while** loop is almost like a while loop however the **condition(expression)** is **checked** at the **bottom of the loop**.
- Hence, despite the condition is true or false, the loop statements are executed **at least once**.

```
do{
   statement(s);
} while(expression );
```

- Statement is executed for the first time.
- Expression is evaluated at the bottom. If it is non-zero, statement is re-executed.
- This cycle continues until expression becomes zero, at which point execution resumes after statement.

do while Loop




```
int a = 10;
do
{
   printf("value of a: %d\n", a);
   a = a + 1;
} while( a < 20 );
```

do while Loop : Example

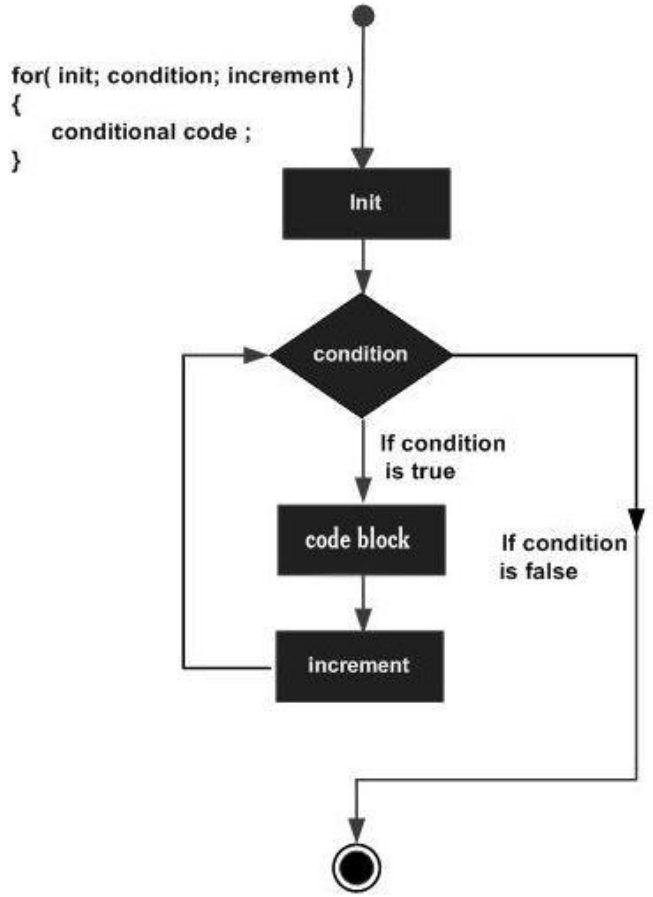
```c
#include <stdio.h>

int main ()
{
    int input;
    do
    {
        printf("Enter a positive Integer : " );
        scanf("%d",&input);
    }
    while(input < 0);
    printf("The square root is  : %.2f" ,sqrt(input));
}
```

for Loop

```
for ( initialization; condition; increment )
{
    statement(s);
}
```

- The for loop in C is the most general looping construct.
- A **for** is used in the situations where the loop statements should be executed **specific number of times**.
- The execution order is similar to while loop where the condition is checked at the loop header.
- The loop header contains three parts: an **initialization**, a **condition**, and an **increment**.
- First, **initialization** declares and initialize any loop control variables.
- Next, the **condition** is evaluated. If it is true, the body of the loop is executed.
- Next, **increment** statement updates any loop control variables.

for Loop




```
for(int a = 10; a < 20; a++)
{
   printf("value of a: %d\n", a);
}
```

for Loop : Example

```
#include <stdio.h>

int main ()
{
    for(int a=0;a<256;a++)
    {
      printf("%d\t%c\n",a,a);
    }
    return 0;
}
```

for Loop : Example

```c
#include <stdio.h>

int main ()
{
    for(int a=1,b=9;a<10,b>0;a++,b--)
    {
      printf("%d\t%d\n",a,b);
    }
    return 0;
}
```

Questions?