Loops

Prepared by: Lec Tasmiah Tamzid Anannya, CS Dept, AIUB

Repetitive Operation

- The default order of execution in a C program is top-down.
- Execution starts at the beginning of the main() function and progresses, statement by statement, until the end of main() is reached.
- What if we need to execute some portion of the program either a specified number of times or until a particular condition is being satisfied?
- In other word, how do we **repeat** (execute over and over) some portion of the program?
- A repetitive operation is done through a **loop** control instruction in C.
- In C, there are three ways by which we can repeat a part of a program, they are **- while**, **do...while**, and **for** statements

Loops in C

while loop syntax:

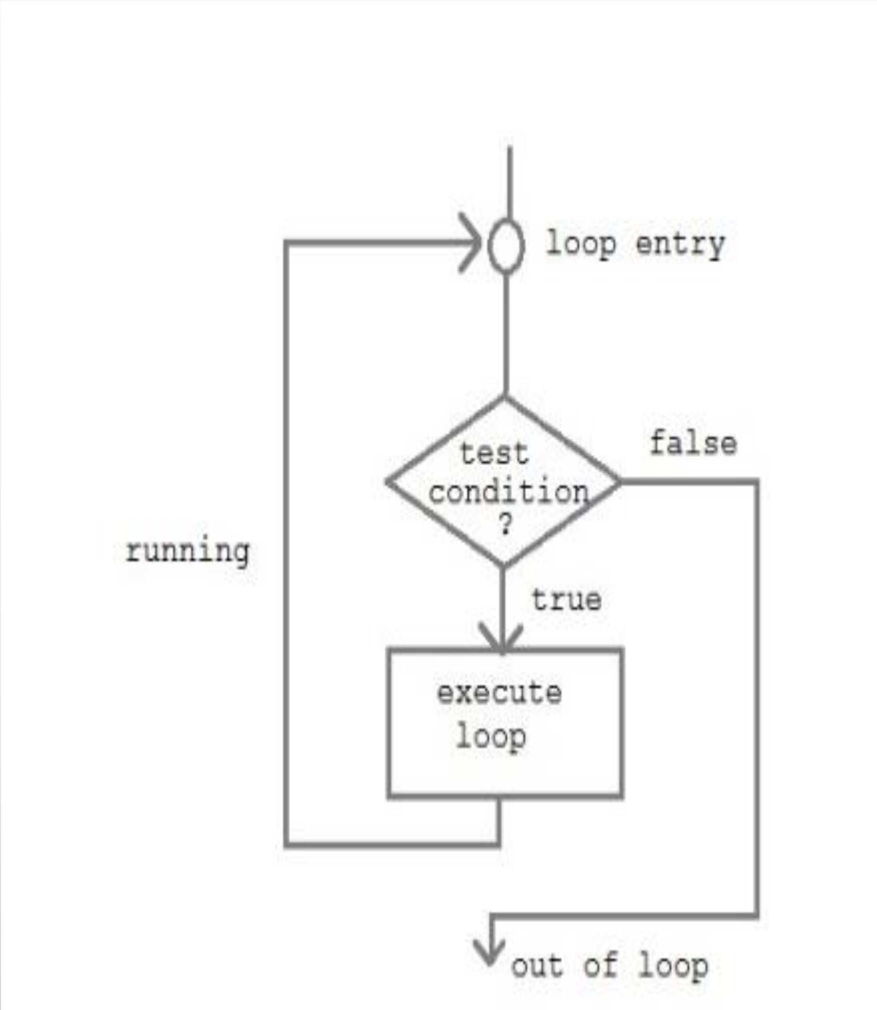
```
while(condition) {
statement;
…
}
```

do..while loop syntax:

```
do {
statement;
…
} while(condition);
```

for loop syntax:

```
for(initial; condition; increment ) {
statement;
…
}
```



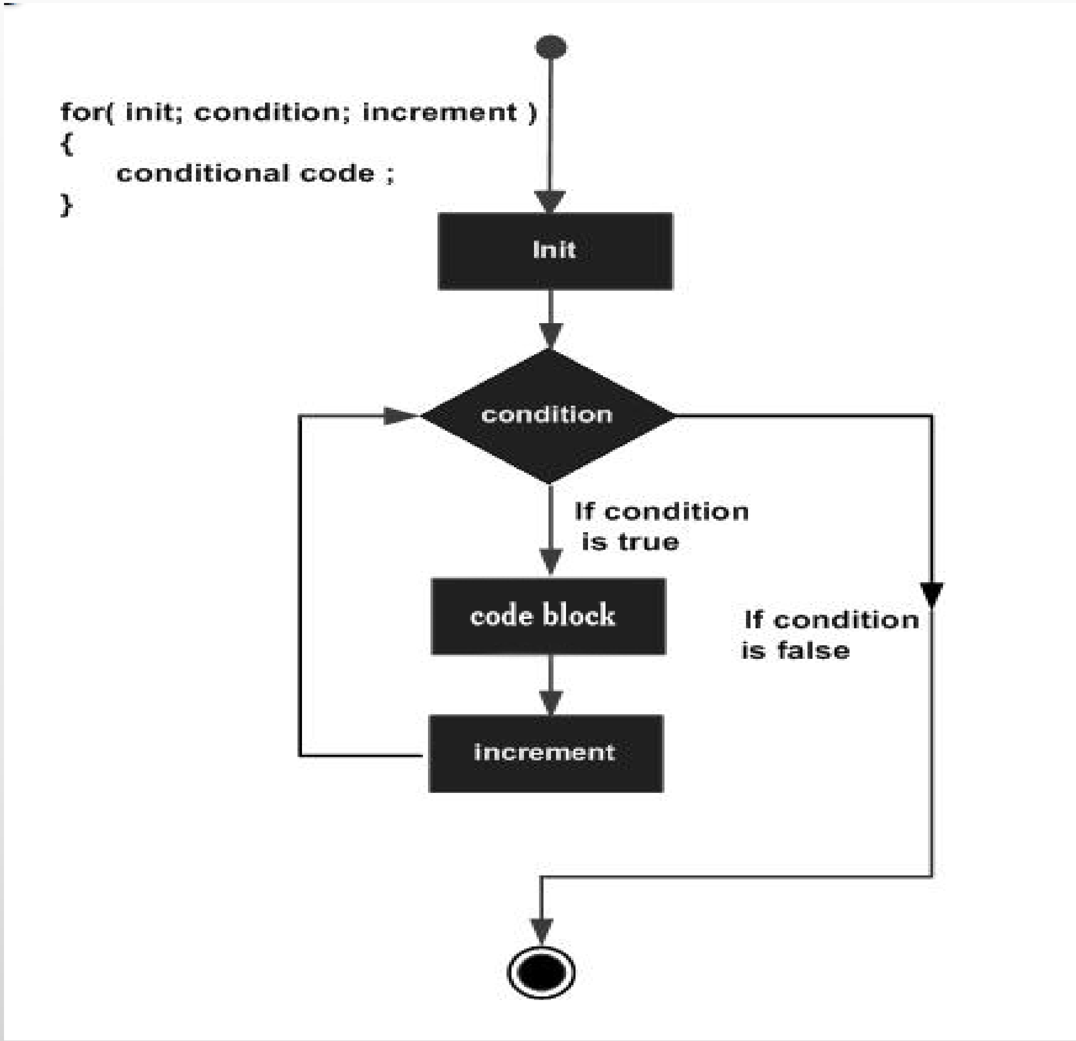
For loop

- We need to how many iterations are done in case of for loop.
- **Syntax**:

  ```
  for(initial; condition; increment ) {
  statement;
  ...
  }
  ```

- Initialization section is used to give an initial value to the variable that controls the loop
- The variable is referred as loop-control variable
- The initialization section executes only once.
- The condition section tests the loop-control variable against a target value.
- If the condition is true, the loop repeats, otherwise the loop stops.

Flowchart



```
for( init; condition; increment )
{
    conditional code ;
}
```

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Initialize i, i=1

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

1<=3 is true

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Print Hello World

```
Hello World!
```

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Increment i, i=2

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

2<=3 is true

Example

```c
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Print Hello World

```
Hello World!
Hello World!
```

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Increment i, i=3

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

3<=3 is true

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Print Hello World

```
Hello World!
Hello World!
Hello World!
```

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Increment i, i=4

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

4<=3 is False

Example

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=3;i++)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

```
Hello World!
Hello World!
Hello World!
```

Exit Loop

- Now write a program that prints numbers from 1 to 100 using a for loop.

- A for loop can run negatively.

```
#include<stdio.h>
int main()
{

    int i;
    for(i=10;i>=0;i--)
    {
        printf("Hello World!\n");
    }
    return 0;
}
```

Guess the output?

```
#include<stdio.h>
int main()
{

    int i;
    for(i=1;i<=100;i=i+5)
    {
        printf("%d\n",i);
    }
    return 0;
}
```

The loop control variable can be incremented or decremented by more than one.

Practice

- Write a program that prints the numbers between 17 and 100 that can be divided by 17.

- Write a program that outputs table of 2.

- Write a program that prompts the user for an integer value. Next, using for loop, make it count down from this value to 0 displaying each number in new line.