While and For Loop in C

while loop

- while(*expression*) *statement*;
- *initialization;*

```
while(conditional-test)
{
        statement;
        increment;
}
```

while loop

```
#include<stdio.h>
int main()
{
    int i=0;
    while(i<=9)
    {
            printf("%d\n", i);
            i++;
    }
    return 0;
}
```

```
#include<stdio.h>
int main()
{
    int i = 0;
    while ( i < 10)
    {
            printf(”*");
            i++;
    }
    return 0;
}
```

for loop

- Allows one or more statements to be repeated
- for (*initialization*; *conditional-test*; *increment*) *statement*;
- Most flexible loop

for loop

- for (*initialization*; *conditional-test*; *increment*) *statement*;
- *initialization*:
  - Give an initial value to the variable that controls the loop
  - *loop-control variable*
  - Executed only once
  - Before the loop begins

for loop

- for (*initialization*; *conditional-test*; *increment*) *statement*;
- *conditional-test:*
  - Tests the *loop-control variable* against a target value
  - If **true** the loop repeats
    - *statement* is executed
  - If **false** the loop stops
    - Next line of code following the loop will be executed

for loop

- for (*initialization*; *conditional-test*; *increment*) *statement*;
- *increment*:
  - Executed at the bottom of the loop

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

1. i is initialized to 1

Initialization part is executed only once

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

2. Conditional test i<3 is true as i is 1, so the loop executes

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

3. The value of i will be printed, which is 1

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

3. The value of i will be incremented, so now i is 2.

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

4. Conditional test i<3 is true as i is 2, so the loop executes

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

5. The value of i will be printed, which is 2

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

6. The value of i will be incremented, so now i is 3.

for loop

```
for(i=1; i<3; i++)
{
        printf("%d\n", i);
}
```

7. Conditional test i<3 is false as i is 3, so the loop stops

for loop

- **for** loop can run negatively
- ***decrement*** can be used instead of *increment*
  - for(i=20; i>0; i--) …
- Can be incremented or decremented by more than one
  - for(i=1; i<100; i+=5)

for loop

- All of the following loops will print 1 to 99
- for(i=1; i<100; i++)

  printf("%d\n", i);
- for(i=1; i<=99; i++)

  printf("%d\n", i);
- for(i=0; i<99; i++)

  printf("%d\n", i+1);
- for(i=0; i<=98; i++)

  printf("%d\n", i+1);
- So selection of initial value and loop control condition is important

for loop

Single Statement

```
for(i=1; i<100; i++)
        printf("%d\n", i);
```

- Prints 1 to 99

```
for(i=100; i<100; i++)
        printf("%d\n", i);
```

- This loop will not execute

Block of Statements

```
sum=0;
prod=1;
for(i=1; i<=5; i++)
{
        sum+=i;
        prod*=i;
}
printf("%d, %d\n", sum, prod);
```