FACULTY OF ENGINEERING AND TECHNOLOGY
UNIVERSITY OF LUCKNOW
LUCKNOW

Computer System and Programming in 'C'

CS-101/201

Er. Zeeshan Ali Siddiqui
Assistant Professor
Deptt. of C.S.E.

ITERATIVE STATEMENTS

Iterative Statement

Iterative statements are used to *repeat* the execution of a list of statements, depending on the value of an integer expression.

Three variants:

1. while loop
2. do-while loop
3. for loop

while LOOP

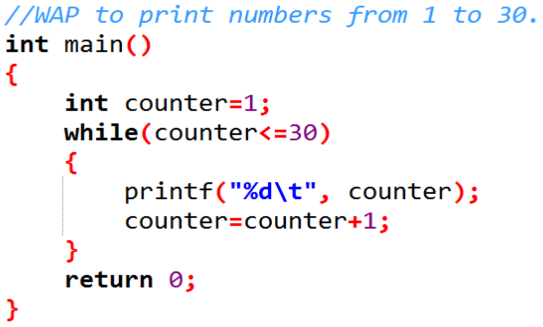
while Loop

In this the test condition is evaluated at the beginning of the loop. That is, *the body of the loop stop executing as soon as the condition is evaluated to false.*

Syntax:

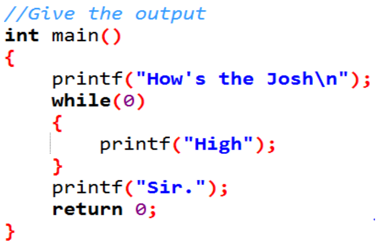
```
Statement x;
while(condition)
{
        Statement Block;
}
Statement y;
```

while Loop: Example



```
//WAP to print numbers from 1 to 30.
int main()
{
    int counter=1;
    while(counter<=30)
    {
        printf("%d\t", counter);
        counter=counter+1;
    }
    return 0;
}
```

while Loop: Quiz



```
//Give the output
int main()
{
    printf("How's the Josh\n");
    while(0)
    {
        printf("High");
    }
    printf("Sir.");
    return 0;
}
```

do-while LOOP

do-while Loop

In this the test condition is evaluated at the end of the loop. That is, *the body of the loop gets executed at least one time* (even if the condition is false).

Syntax:

Statement x;

do

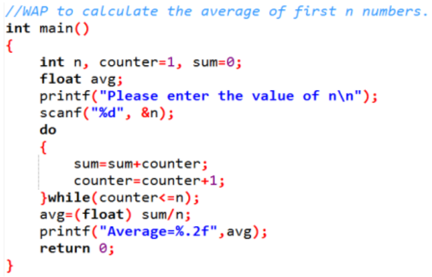
{

Statement Block;

} **while**(condition);

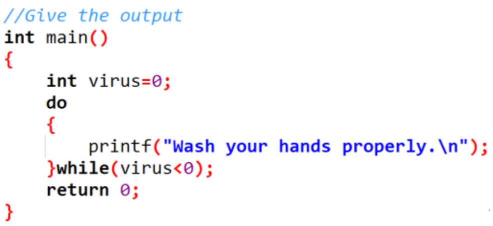
Statement y;

do-while Loop: Example



```
//WAP to calculate the average of first n numbers.
int main()
{
    int n, counter=1, sum=0;
    float avg;
    printf("Please enter the value of n\n");
    scanf("%d", &n);
    do
    {
        sum=sum+counter;
        counter=counter+1;
    }while(counter<=n);
    avg=(float) sum/n;
    printf("Average=%.2f",avg);
    return 0;
}
```

while Loop: Quiz



```
//Give the output
int main()
{
    int virus=0;
    do
    {
        printf("Wash your hands properly.\n");
    }while(virus<0);
    return 0;
}
```

for LOOP

for Loop

Syntax:

Statement x;

for(initialization; condition; updation)

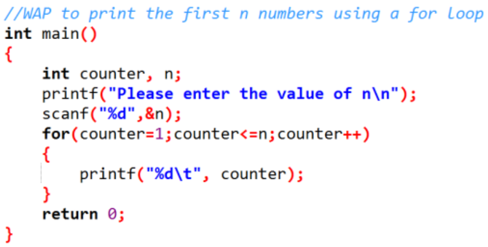
{

    Statement Block;

}

Statement y;

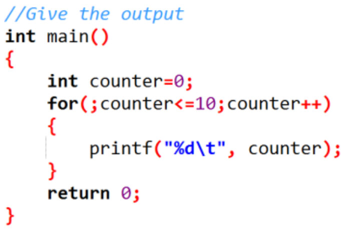
for Loop: Example



```
//WAP to print the first n numbers using a for loop
int main()
{
    int counter, n;
    printf("Please enter the value of n\n");
    scanf("%d",&n);
    for(counter=1;counter<=n;counter++)
    {
        printf("%d\t", counter);
    }
    return 0;
}
```

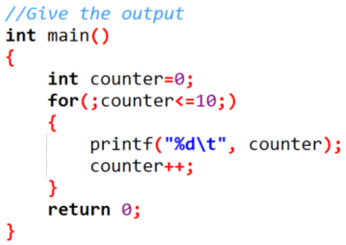
for LOOP Analysis

for Loop: no initialization



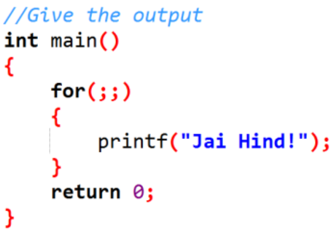
```
//Give the output
int main()
{
    int counter=0;
    for(;counter<=10;counter++)
    {
        printf("%d\t", counter);
    }
    return 0;
}
```

for Loop: updation omission



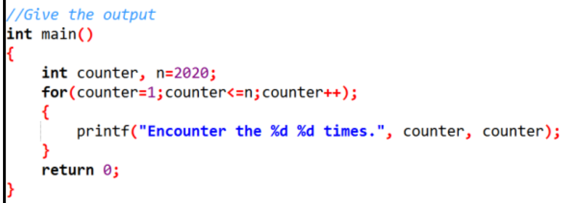
```
//Give the output
int main()
{
    int counter=0;
    for(;counter<=10;)
    {
        printf("%d\t", counter);
        counter++;
    }
    return 0;
}
```

for Loop: only skeleton



```
//Give the output
int main()
{
    for(;;)
    {
        printf("Jai Hind!");
    }
    return 0;
}
```

for Loop: Quiz



```
//Give the output
int main()
{
    int counter, n=2020;
    for(counter=1;counter<=n;counter++);
    {
        printf("Encounter the %d %d times.", counter, counter);
    }
    return 0;
}
```

Selecting an appropriate Loop

Loop

- Entry controlled or pre-test (*Condition is tested before the loop starts*)
- Exit controlled or post-test (*Condition is tested after the loop is executed*)

- Therefore, if your requirement is to have a *pre-test loop* then choose either *for loop or while loop*.

- In case, you need to have a *post-test loop* then choose a *do-while loop*.

NESTED LOOPs

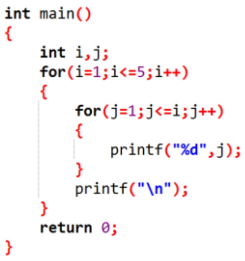
Nested Loops

- Loops can be placed inside other loops.
- A for loop can be used to control the number of times that a particular set of statements will be executed.
- Another outer loop could be used to control the number of times that a whole loop is repeated.
- In C, loops can be nested to any desired level.

Nested Loops: Example

- WAP to print the following pattern-

```
1
12
123
1234
12345
```



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

Programming Exercise

- Study multiple loop variables in for loop. Examine its uses and utility.
- WAP in C which accepts a number and display it in words. (e.g. 123 : One Two Three).
- WAP to find the sum of digits of the entered number.
- WAP to find the factorial of a given number.
- WAP to print the following pattern:

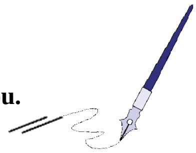
```
*****
****
**
*
```

- WAP to print the Fibonacci series. (0,1,1,2,3,5,8,... upto N)
- WAP to convert decimal number into binary number (Integer part only).
- WAP to convert binary number into decimal number (Integer part only).

Interesting Exercise

- Give some *real world* examples of while, do-while and for loops.
- Differentiate between while and do-while loop.
- Can a for loop behaves like a while loop and vice-versa. Give your reasoning.
- How to create a infinite loop in C? Give as many distinct solutions (programs) as you can.
- Give some programming examples where we need an infinite loop.
- Can you think about some real world examples of infinite loop?

Thank You.



BTQ

BTQ: Brain Teaser Question

There are 30 horse in a field, and 28 chickens.
How many didn't?

☺