

# Programming Languages

Programming Languages



**CENTRAL  
UNIVERSITY**

FAITH • INTEGRITY • EXCELLENCE

**PRESENTED BY:  
DR. K. KISSI MIREKU**

# Computer Programming – Loops



**CENTRAL  
UNIVERSITY**

FAITH • INTEGRITY • EXCELLENCE

A red vertical bar with a crest featuring a white eagle and a star on a red shield.

# Computer Programming - Loops

Let's consider a situation when you want to print **Hello, World!** five times. Here is a simple C program to do the same –

```
#include <stdio.h>

int main() {
    printf( "Hello, World!\n");
    printf( "Hello, World!\n");
    printf( "Hello, World!\n");
    printf( "Hello, World!\n");
    printf( "Hello, World!\n");
}
```

When the above program is executed, it produces the following result –

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

# Computer Programming - Loops

- Almost all the programming languages provide a concept called **loop**, which helps in executing one or more statements up to a desired number of times. All high-level programming languages provide various forms of loops, which can be used to execute one or more statements repeatedly.
- Let's write the above C program with the help of a **while loop** and later, we will discuss how this loop works

```
#include <stdio.h>

int main() {
    int i = 0;

    while ( i < 5 ) {
        printf( "Hello, World!\n");
        i = i + 1;
    }
}
```

When the above program is executed, it produces the following result –

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

# While Loops

- The program makes use of a **while loop**, which is being used to execute a set of programming statements enclosed within {...}. Here, the computer first checks whether the given condition, i.e., variable "a" is less than 5 or not and if it finds the condition is true, then the loop body is entered to execute the given statements. Here, we have the following two statements in the loop body –

- *First statement is printf() function, which prints Hello World!*
- *Second statement is  $i = i + 1$ , which is used to increase the value of variable  $i$*

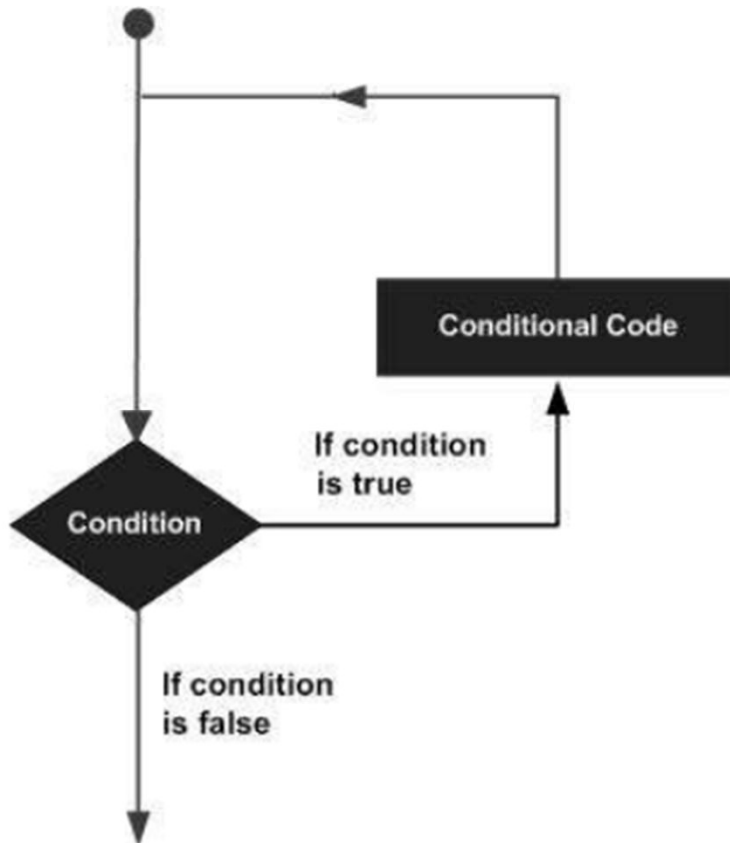
After executing all the statements given in the loop body, the computer goes back to while(  $i < 5$  ) and the given condition, (  $i < 5$  ), is checked again, and the loop is executed again if the condition holds true. This process repeats till the given condition remains true which means variable "a" has a value less than 5.

- To conclude, a loop statement allows us to execute a statement or group of statements multiple times.



# While Loops

The general form of a loop statement in most of the programming languages –

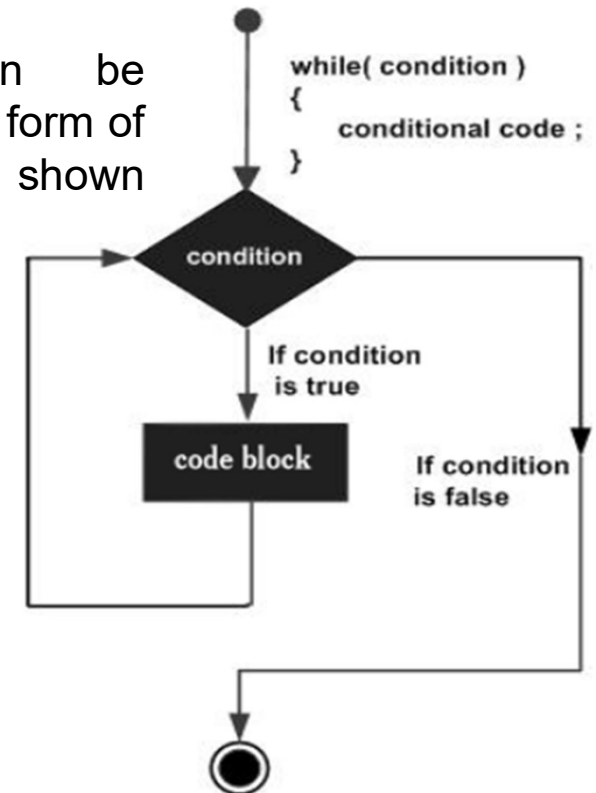


# While Loops

- A **while loop** available in C Programming language has the following syntax –

```
while ( condition ) {  
    /*....while loop body ....*/  
}
```

The code can be represented in the form of a flow diagram as shown beside –



# While Loops

The following important points are to be noted about a while loop –

- A while loop starts with a keyword **while** followed by a **condition** enclosed in ( ).
- Further to the while() statement, you will have the body of the loop enclosed in curly braces {...}.
- A while loop body can have one or more lines of source code to be executed repeatedly.
- If the body of a while loop has just one line, then its optional to use curly braces {...}.
- A while loop keeps executing its body till a given **condition** holds true. As soon as the condition becomes false, the while loop comes out and continues executing from the immediate next statement after the while loop body.
- A condition is usually a relational statement, which is evaluated to either true or false. A value equal to zero is treated as false and any non-zero value works like true.



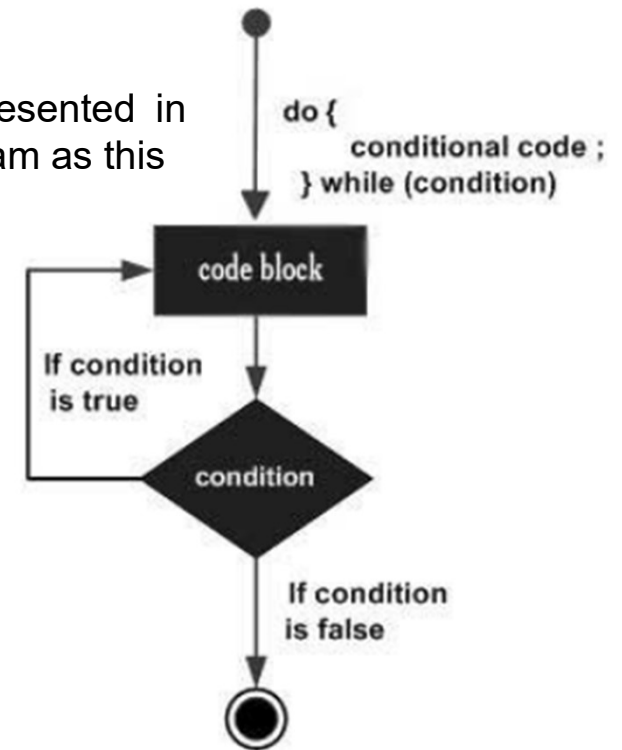


# The do...while Loop

- A while loop checks a given condition before it executes any statements given in the body part.
- C programming provides another form of loop, called **do...while** that allows to execute a loop body before checking a given condition. It has the following syntax –

```
do {  
    /*.....do...while loop body ....*/  
}  
while ( condition );
```

The code can be represented in the form of a flow diagram as this



# The do...while Loop

- If you will write the above example using **do...while** loop, then **Hello, World** will produce the same result –

```
#include <stdio.h>

int main() {
    int i = 0;

    do {
        printf( "Hello, World!\n");
        i = i + 1;
    }
    while ( i < 5 );
}
```

When the program is executed, it produces the following result –

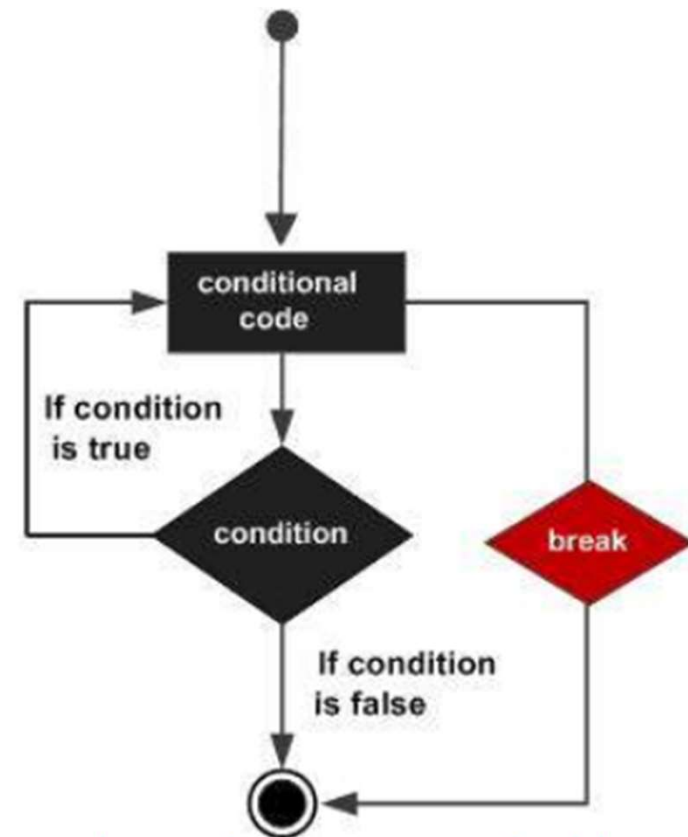
```
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
```

# The break statement

- When the **break** statement is encountered inside a **loop**, the **loop** is immediately terminated and the program control resumes at the next statement following the **loop**.

The syntax for a **break** statement in C is as follows –

- **break;**



A break statement can be represented in the form of a flow diagram as shown above –

# The break statement

An example of the **break** Statement in a program, but it will work out after printing Hello World! only three times –

```
#include <stdio.h>

int main() {
    int i = 0;
    do {
        printf( "Hello, World!\n");
        i = i + 1;

        if( i == 3 ) {
            break;
        }
    }
    while ( i < 5 );
}
```

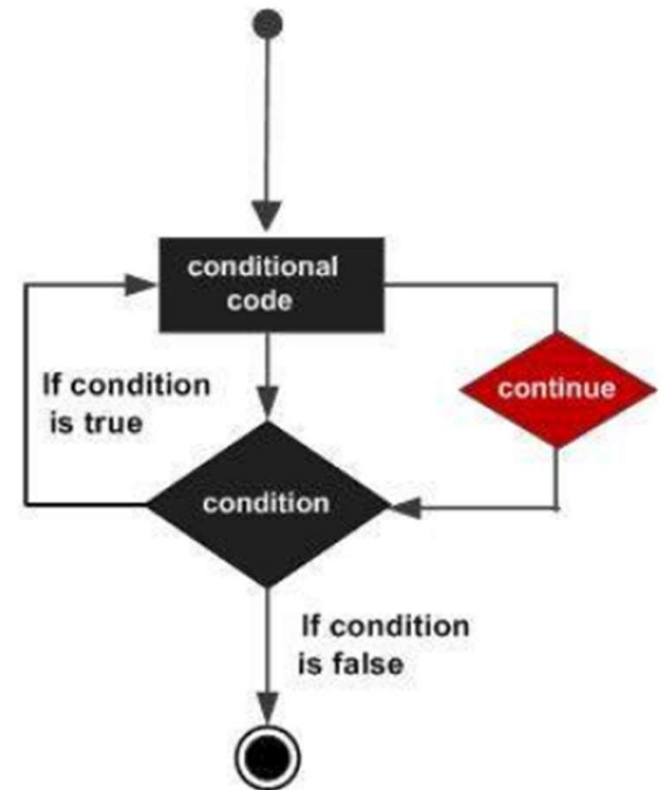
When the program is executed, it produces the following result –

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

# The Continue statement

The continue statement in C programming language works somewhat like the break statement. Instead of forcing termination, continue forces the next iteration of the loop to take place, skipping any code in between.

The syntax for a continue statement in C is as follows –





# The continue statement

An example of the **continue** statement in a program, but it will skip printing when the variable has a value equal to 3 –

```
#include <stdio.h>

int main() {
    int i = 0;
    do {
        if( i == 3 ) {
            i = i + 1;
            continue;
        }
        printf( "Hello, World!\n");
        i = i + 1;
    }
    while ( i < 5 );
}
```

When the program is executed, it produces the following result –

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

# Loops in Java

Following is the equivalent program written in Java that too supports **while** and **do...while** loops.

The following program prints **Hello, World!** five times as we did in the case of C Programming –


```
public class DemoJava {  
    public static void main(String []args) {  
        int i = 0;  
  
        while ( i < 5 ) {  
            System.out.println("Hello, World!");  
            i = i + 1;  
        }  
    }  
}
```

The **break** and **continue** statements in Java programming work quite the same way as they work in C programming.

# Loops in Python

Following is the equivalent program written in Python that too supports **while** and **do...while** loops.

Python does not make use of curly braces for the loop body, instead it simply identifies the body of the loop using indentation of the statements.



```
i = 0

while (i < 5):
    print "Hello, World!"
    i = i + 1
print "Loop ends"
```

When the program is executed, it produces the following result –

```
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Loop ends
```

The **break** and **continue** statements in Python programming work quite the same way as they work in C programming.



# Computer Programming – numbers



**CENTRAL  
UNIVERSITY**

FAITH • INTEGRITY • EXCELLENCE

# Computer Programming - numbers

Every programming language provides support for manipulating different types of numbers such as simple whole integers and floating point numbers. C, Java, and Python categorize these numbers in several categories based on their nature.

Type	Keyword	Value range which can be represented by this data type
Number	int	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
Small Number	short	-32,768 to 32,767
Long Number	long	-2,147,483,648 to 2,147,483,647
Decimal Number	float	1.2E-38 to 3.4E+38 till 6 decimal places
Type	Keyword	Value range which can be represented by this data type

These data types are called primitive data types and you can use these data types to build more data types, which are called user-defined data types.



# Numbers

```
#include <stdio.h>
```

```
int main() {  
    short  s;  
    int    i;  
    long   l;  
    float  f;  
    double d;  
  
    s = 10;  
    i = 1000;  
    l = 1000000;  
    f = 230.47;  
    d = 30949.374;  
  
    printf( "s: %d\n", s);  
    printf( "i: %d\n", i);  
    printf( "l: %ld\n", l);  
    printf( "f: %.3f\n", f);  
    printf( "d: %.3f\n", d);  
}
```

Printing various types of numbers available in C programming language –

Rest of the coding is very obvious, but we used **%.3f** to print float and double, which indicates the number of digits after the decimal to be printed. When the above program is executed, it produces the following result –

```
s: 10  
i: 1000  
l: 1000000  
f: 230.470  
d: 30949.374
```

# Numbers in Java

When the program is executed, it produces the following result –

```
public class DemoJava {  
    public static void main(String []args) {  
        short s;  
        int i;  
        long l;  
        float f;  
        double d;  
  
        s = 10;  
        i = 1000;  
        l = 1000000L;  
        f = 230.47f;  
        d = 30949.374;  
  
        System.out.format( "s: %d\n", s);  
        System.out.format( "i: %d\n", i);  
        System.out.format( "l: %d\n", l);  
        System.out.format( "f: %f\n", f);  
        System.out.format( "d: %f\n", d);  
    }  
}
```

```
s: 10  
i: 1000  
l: 1000000  
f: 230.470001  
d: 30949.374000
```



# Numbers in Python

```
s = 10
i = 1000
l = 1000000
f = 230.47
d = 30949.374

print "s: ", s
print "i: ", i
print "l: ", l
print "f: ", f
print "d: ", d
```

When the program is executed, it produces the following result –

```
s: 10
i: 1000
l: 1000000
f: 230.47
d: 30949.374
```

# Computer Programming - Functions



**CENTRAL  
UNIVERSITY**

FAITH • INTEGRITY • EXCELLENCE

# Computer Programming - Functions

- A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for your application and a high degree of code reusing. You have already seen various functions like **printf()** and **main()**.

These are called built-in functions provided by the language itself, but we can write our own functions as well and this tutorial will teach you how to write and use those functions in C programming language.

- Good thing about functions is that they are famous with several names. Different programming languages name them differently, for example, functions, methods, sub-routines, procedures, etc.



# Functions

```
#include <stdio.h>

int main() {
    int set1[5] = {10, 20, 30, 40, 50};
    int set2[5] = {101, 201, 301, 401, 501};
    int i, max;

    /* Process first set of numbers available in set1[] */
    max = set1[0];
    i = 1;
    while( i < 5 ) {
        if( max < set1[i] ) {
            max = set1[i];
        }
        i = i + 1;
    }

    printf("Max in first set = %d\n", max );

    /* Now process second set of numbers available in set2[] */
    max = set2[0];
    i = 1;
    while( i < 5 ) {
        if( max < set2[i] ) {
            max = set2[i];
        }
        i = i + 1;
    }
    printf("Max in second set = %d\n", max );
}
```

When the program is executed, it produces the following result –

```
Max in first set = 50
Max in second set = 501
```



# Defining a Function

The general form of a function definition in C programming language is as follows –

```
return_type function_name( parameter list ) {  
    body of the function  
  
    return [expression];  
}
```



# Defining a Function

- A function definition in C programming consists of a *function header* and a *function body*. Here are all the parts of a function –
- **Return Type** – A function may return a value. The **return\_type** is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return\_type is the keyword **void**.
- **Function Name** – This is the actual name of the function. The function name and the parameter list together constitute the function signature.
- **Parameter List** – A parameter is like a placeholder. When a function is invoked, you pass a value as a parameter. This value is referred to as the actual parameter or argument. The parameter list refers to the type, order, and number of the parameters of a function. Parameters are optional; that is, a function may contain no parameters.
- **Function Body** – The function body contains a collection of statements that defines what the function does.



# Calling a Function

- While creating a C function, you give a definition of what the function has to do. To use function, you will have to call that function to perform a defined task.

Now, let's write the above example with the help of a function –

```
#include <stdio.h>

int getMax( int set[] ) {
    int i, max;

    max = set[0];
    i = 1;
    while( i < 5 ) {
        if( max < set[i] ) {
            max = set[i];
        }
        i = i + 1;
    }
    return max;
}

main() {
    int set1[5] = {10, 20, 30, 40, 50};
    int set2[5] = {101, 201, 301, 401, 501};
    int max;

    /* Process first set of numbers available in set1[] */
    max = getMax(set1);
    printf("Max in first set = %d\n", max );

    /* Now process second set of numbers available in set2[] */
    max = getMax(set2);
    printf("Max in second set = %d\n", max );
}
```

# Functions in Java

- Java programming names them as **methods**, but the rest of the concepts remain more or less same.

When the program is executed, it produces the following result –

Max in first set = 50

Max in second set = 501

```
public class DemoJava {  
    public static void main(String []args) {  
        int[] set1 = {10, 20, 30, 40, 50};  
        int[] set2 = {101, 201, 301, 401, 501};  
        int max;  
  
        /* Process first set of numbers available in set1[] */  
        max = getMax(set1);  
        System.out.format("Max in first set = %d\n", max );  
  
        /* Now process second set of numbers available in set2[] */  
        max = getMax(set2);  
        System.out.format("Max in second set = %d\n", max );  
    }  
    public static int getMax( int set[] ) {  
        int i, max;  
        max = set[0];  
        i = 1;  
  
        while( i < 5 ) {  
            if( max < set[i] ) {  
                max = set[i];  
            }  
            i = i + 1;  
        }  
        return max;  
    }  
}
```

# Functions in Python

- Once again, if you know the concept of functions in C and Java programming, then Python is not much different.

When the program is executed, it produces the following result –

Max in first set = 50  
Max in second set = 501

```
def getMax( set ):
    max = set[0]
    i = 1

    while( i < 5 ):
        if( max < set[i] ):
            max = set[i]

        i = i + 1
    return max

set1 = [10, 20, 30, 40, 50]
set2 = [101, 201, 301, 401, 501]

# Process first set of numbers available in set1[]
max = getMax(set1)
print "Max in first set = ", max

# Now process second set of numbers available in set2[]
max = getMax(set2)
print "Max in second set = ", max
```

The decorative element on the left consists of a thick vertical red bar. Centered on this bar is a shield-shaped emblem. The emblem has a red background and features a white eagle with its wings spread, perched on a white floral or sunburst-like symbol. Above the eagle's head is a small white star. The shield is outlined in gold.

# Thank you

- Computer programming is the act of writing computer programs, which are a sequence of instructions written using a Computer Programming Language to perform a specified task by the computer.
- Computer Programming is fun and easy to learn provided you adopt a proper approach.
- These presentations attempt to cover the basics of computer programming using a simple and practical approach for the benefit of students.
- Thank you for your time