ABOUT THE TUTORIAL

Computer Prgramming Tutorial

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. This tutorial will take you through simple and practical approach while learning computer programming.

Audience

This tutorial has been prepared for the people who are struggling to learn computer programming but they are unable to learn it due to lack of proper guidance and approach. I'm confident that after completing this tutorial you will be at a level where you can code in C Programming language and will have basic understanding of Java and Python programming languages as well and you can continue further from that point onwards.

If you are completely new to Computer Programming, then my recommendation is to read this tutorial twice or even thrice. First reading will not give you much idea, but during your second reading, you will start grasping most of the concepts and you will enjoy while writing great computer programs.

Prerequisites

I do not expect much from your as pre-requisites to teach you computer programming but I assume that at-least you have basic knowledge about computer and its peripherals like keyboard, mouse, screen, printer, etc. If you know more than this like word editor, good typing speed, internet search, etc., then it will help you a lot while doing your computer programming.

Copyright & Disclaimer Notice

same sequence could have been given in Spanish Language, Hindi Language, Arabic or any other human language provided someone, who is asking about the way, knows about such languages.

Now, let's go back and try to understand about a computer program, which is a sequence of instructions written in a Computer Language to perform a specified task by the computer. Following is a simple program written in **Python** programming Language:

```
print "Hello, World!"
```

Above computer program instructs computer to print "Hello, World!" on computer screen.

- A computer program is also called a **computer software**, which can range from two lines to millions of lines of instructions.
- Computer program instructions are also called program source code and **computer programming** is also called **program coding**.
- A computer machine without a computer program is just a dump box and thus computer program brings a computer machine to live state.

Like human has several languages to communicate their message, computer scientists have developed several computer-programming languages to provide instructions to the computer (i.e., to write computer programs). We will see several computer programming languages in subsequent chapters.

What is Computer Programming?

If you understood what is **computer program**, then I will say *the act of writing computer programs is called computer programming.*

As I mentioned earlier, there are 1000+ of programming languages, which can be used to write computer programs and following are few of them:

- Java
- C
- C++
- Python
- PHP
- Perl
- Ruby

What Computer Program can do?

Today computer programs are being used in almost every field, household, agriculture, medical, entertainment, defense, communication, etc. Following are few applications of computer programs:

- MS Word, MS Excel, Adobe Photoshop, Internet Explorer, Chrome, etc., are example of computer programs.
- Computer programs are being used to develop graphics and special effects in movie making.
- Computer programs are being used to perform Ultrasounds, X-Rays, and other medical examinations.

Conclusion

Not sure if you understood what I taught you above in this chapter, but if you did not understand then I will suggest to go through it once again and make sure you understood all the above concepts. But if you understood these concepts, then you are almost done and let's proceed to the next chapter, which you are going to enjoy a lot.

CHAPTER

7

Programming - Keywords

So far, you have covered two important concepts called variables and their data types. You have seen how we have used **int, long** and **float** keywords to specify different data types. You also have seen how we named our variables to store different values.

Though this chapter is not required separately because reserved keywords are part of basic programming syntax but I kept it separate to explain it right after data types and variables to make it easy to understand.

Like int, long, and float, there are many other keywords supported by C programming language which we will use for different purpose. Different programming languages provide different set of reserved keywords, but there is one important & common rule in all the programming languages that we cannot use a reserved keyword to name our variables, which means we cannot name our variable like **int** or **float** rather these keywords can only be used to specify a variable data type.

For example, if you will try to use any reserved keyword for the purpose of variable name, then you will get syntax error, as follows:

```
#include <stdio.h>

main()
{
   int float;

   float = 10;

   printf( "Value of float = %d\n", float);
}
```

When you compile above program, it produces the following error:

```
main.c: In function 'main':
main.c:5:8: error: two or more data types in declaration specifiers
    int float;
......
```

But now let's give proper name to our integer variable, then above program should compile and execute successfully:

```
#include <stdio.h>

main()
{
   int count;
```

CHAPTER

8

Programming - Operators

An operator in a programming language is a symbol that tells the compiler or interpreter to perform specific mathematical, relational or logical operation and produce final result. This chapter will explain you what are the **operators** and will take you through important arithmetic and relational operators available in C, Java and Python programming languages.

Arithmetic Operators

Computer programs are widely used for mathematical calculations. We can write a computer program which can do simple calculation like adding two numbers (2 + 3) and we can also write a program, which can solve a complex equation like $P(x) = x^4 + 7x^3 - 5x + 9$. If you have been even a poor student, you must be aware that in first expression 2 and 3 are operands and + is an operator. A similar concept exists in Computer Programming.

Here we took following two mathematics examples:

```
2 + 3

P(x) = x^4 + 7x^3 - 5x + 9.
```

These two statements are called arithmetic expressions in a programming language and **plus**, **minus**used in these expressions are called arithmetic operators and values used in these expressions like 2, 3 and x, etc., are called operands. In their simplest form such expressions produce numerical results.

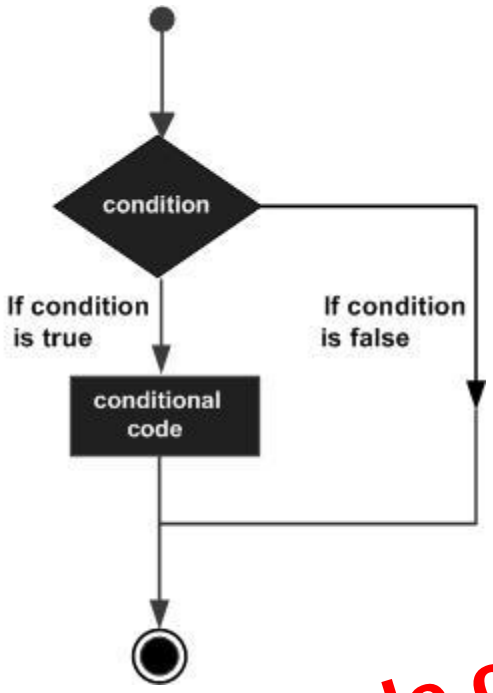
Similar way, a programming language provides various arithmetic operators. Following table lists down few of the important arithmetic operators available in C programming language. Assume variable A holds 10 and variable B holds 20, then:

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiplies both operands	A * B will give 200
/	Divides numerator by de-numerator	B / A will give 2
%	This gives remainder of an integer division	B % A will give 0

Following is a simple example of C Programming to understand above mathematical operators:

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

```
c = a / b
print "Value of c = ", c

c = a % b
print "Value of c = ", c

if( a == 10 ):
   print "a is equal to 10"
```



Let's write a C program with the help of **if conditional statements** to convert above given situation into programming code:


```
#include <stdio.h>

main()
{
   int x = 45;

   if( x > 95)
   {
      printf( "Student is brilliant\n");
   }

   if( x < 30)
   {
      printf( "Student is poor\n");
   }

   if( x < 95 && x > 30 )
   {
      printf( "Student is average\n");
   }
}
```

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

```
Student is average
```

Above program makes use of **if conditional statements**. Here, first if statement checks whether given condition i.e., variable x is greater than 95 or not and if it finds condition is true, then the conditional body is entered to execute given statements. Here we have only one *printf()* statement to print a remark about the student.

Similar way, second if statement works. Finally, third if statement is executed, here we have following two conditions:

```
   {
      printf( "Student is brilliant\n");
   }
   else
   {
      printf( "Student is not brilliant\n");
   }
}
```

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

```
Student is not brilliant
```

if...elseif...else statement

An **if** statement can be followed by an optional **else if...else** statement, which is very useful to test various conditions using single if...else if statement.

When using if , else if , else statements, there are few points to keep in mind:

- An if can have zero or one else's and it must come after any else if's.
- An if can have zero to many else if's and they must come before the else.
- Once an else if succeeds, none of the remaining else if's or else's will be tested.

The syntax of an **if...else if...else** statement in C programming language is:

```
if(boolean_expression 1)
{
   /* Executes when the boolean expression 1 is true */
}
else if( boolean_expression 2)
{
   /* Executes when the boolean expression 2 is true */
}
else if( boolean_expression 3)
{
   /* Executes when the boolean expression 3 is true */
}
else
{
   /* Executes when the none of the above condition is true */
}
```

Now with the help of **if...elseif...else** statement, very first program can be coded as follows:

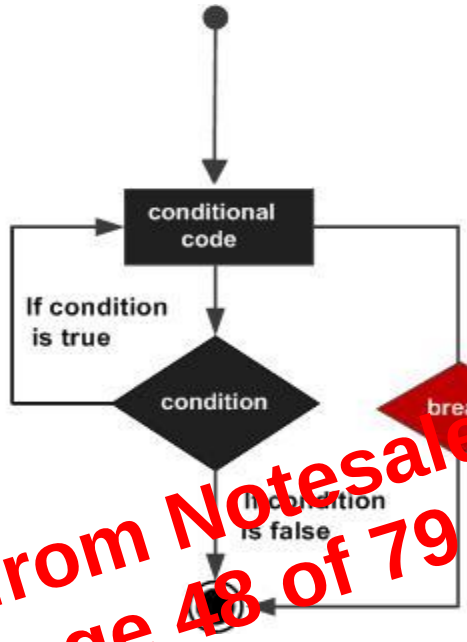
```
#include <stdio.h>

main()
{
   int x = 45;

   if( x > 95)
   {
      printf( "Student is brilliant\n");
   }
   else if( x < 30)
```

The break statement

When the **break** statement is encountered inside a loop, the loop is immediately terminated and 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 below:



For example, is a variant of the above program, but it will come out after printing Hello World! only three times:

```
#include <stdio.h>

main()
{
   int i = 0;

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

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

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

Characters in Python

Python does not support any character data type but all the characters are treated as string, which is a sequence of characters and we will study strings in a separate chapter. But you do not need to have any special arrangement while using a single character in Python.

Following is the equivalent program written in Python:

```
ch1 = 'a';
ch2 = '1';
ch3 = '$';
ch4 = '+';

print "ch1: ", ch1
print "ch2: ", ch2
print "ch3: ", ch3
print "ch4: ", ch4
```

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

```
ch1:  a
ch2:  1
ch3:  $
ch4:  +
```

Python also supports escape sequence in very similar way you have used them in C programming.

```
    char ch[5] = {'H', 'e', 'l', 'l', 'o'};
    int i = 0;

    while( i < 5 )
    {
        printf("ch[%d] = %c\n", i, ch[i] );
        i = i + 1;
   }
}
```

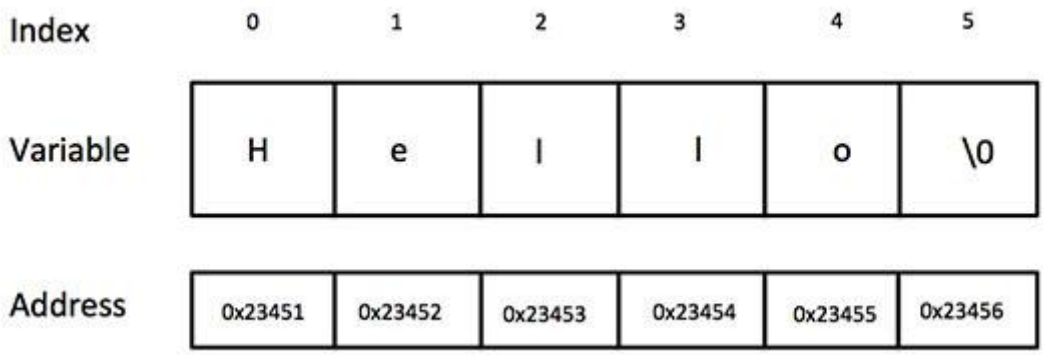
Here, we used %c to print character value. When the above code is compiled and executed, it produces the following result:

```
ch[0] = H
ch[1] = e
ch[2] = l
ch[3] = l
ch[4] = o
```

If you are done with the above example, then I think you understood about strings in C programming, because **strings in C are represented as arrays of characters**. C programming simplified the assignment and printing of strings. Let's check same example once again with simplified syntax:

```
#include <stdio.h>

main()
{
    char ch[5] = "Hello";
    int i = 0;

    /* Print as a complete string */
    printf("String = %s\n", ch);

    /* Print character by character */
    while( i < 5 )
    {
        printf("ch[%d] = %c\n", i, ch[i] );
        i = i + 1;
   }
}
```


Here, we used %s to print full string value using array name **ch**, which is actually beginning of the memory address holding ch variable as shown below:



Though it's not visible from the above examples, but internally C program assigns null character **'\0'** as the last character of every string. This indicates the end of the string and it means if you want to store a 5 character string in an array then you must define array size of 6 as a good practice, though C does not complain about it.

```
String = Hello
```

Strings in Python

Creating strings in Python is as simple as simply assigning a string into a Python variable using single or double quote as shown below:

Following is a simple program, which creates two strings and print them using print() function:

```
var1 = 'Hello World!'
var2 = "Python Programming"

print "var1 = ", var1
print "var2 = ", var2
```

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

```
var1 =  Hello World!
var2 =  Python Programming
```

Python does not support a character type; these are treated as strings of length one, thus also considered a substring.

To access substrings, use the square brackets for slicing along with the index or indices to obtain your substring. Following is a simple example:

```
var1 = 'Hello World!'
var2 = "Python Programming"

print "var1[0]: ", var1[0]
print "var2[1:5]: ", var2[1:5]
```


When the above code is executed, it produces the following result:

```
var1[0]:  H
var2[1:5]:  ytho
```

CHAPTER

17

Programming - Summary

Thanks for your patience and going through this lengthy tutorial though I tried my best to keep it concise but because this subject is really very interesting to me so, I could not stop myself to write such a big content.

If you did not understand any of the concepts, then I highly recommend to go through the tutorial once again and once you are comfortable with the concepts explained in this tutorial, then you are fully charged and can proceed for detailed tutorial.

There are many other subjects related to computer programming which I did not cover intentionally to avoid your confusion, but I'm sure those concepts will not give you a pain as long as you make yourself comfortable with the concepts explained in this tutorial.

At tutorialspoint, we have put lot of effort to prepare nice and detailed tutorial on C, Java and Python programming languages and recommend you to start either of them just immediate after complete Computer Programming Fundamentals tutorial.

Kindly share with me your experience with this tutorial mentioning tutorial name in the subject line at **webmaster et tutorialspoint.com**. If you have further suggestion to improve this tutorial further, then I will welcome your suggestions.