**DIFFERENCE BETWEEN C AND C++**

The key differences include:

|  |  |
| --- | --- |
| **C** | **C++** |
| It is a structural or procedural programming language. | It is an object oriented programming language. |
| Emphasis is on procedure or steps to solve a problem | Emphasis is on objects rather than procedure |
| Functions are the fundamental building blocks. | Objects are the fundamental building blocks. |
| In C, the data is not secured. | Data is hidden and can’t be accessed by external functions. |
| C uses scanf() and printf() functions for standard input and output. | C uses cin>> and cout<< functions for standard input and output. |
| In C, namespace feature is absent. | In C++, namespace feature is present. |
| C program file is saved with .C extension. | C++ program file is saved with .CPP extension. |

***Table 1: Difference between C and C++***

|  |  |  |
| --- | --- | --- |
| **Character** | **Name** | **Description** |
| // | double slash | Marks the beginning of a comment |
| # | Pound sign | Marks the beginning of a preprocessor directive |
| < > | Opening and closing brackets | Encloses a filename when used with the #include directive |
| ( ) | Opening and closing parenthesis | Used in naming a function, as in int main () |
| { } | Opening and closing braces | Encloses a group of statements, such as the contents of a function. |
| " " | Opening and closing quotation marks | Encloses a string of characters, such as a message that is to be printed on the screen |
| ; | Semicolon | Marks the end of a complete programming statement |

***Table 2: Mandatory symbols in basic program***

**COMMON ESCAPE SEQUENCES**

|  |  |  |
| --- | --- | --- |
| **Escape Sequence** | **Name** | **Description** |
| \n | Newline | Causes the cursor to go to the next line for subsequent printing |
| \t | Horizontal tab | Causes the cursor to skip over to the next tab stop |
| \b | Backspace | Causes the cursor to back up, or move left one position |
| \r | Return | Causes the cursor to go to the beginning of the current line, not the next line |
| \\ | Backslash | Causes a backslash to be printed |
| \' | Single quote | Causes a single quotation mark to be printed |
| \" | Double quote | Causes a double quotation mark to be printed |

***Table 3: Escape Sequence***

**DATATYPES**

There are many different types of data.

Variables are classified according to their data type, which determines the kind of information that may be stored in them. Integer variables only hold whole numbers.

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Size** | **Range** |
| short | 2 bytes | -32,768 to +32,767 |
| unsigned short | 2 bytes | 0 to +65,535 |
| int | 4 bytes | -2,147,4833,648 to +2,147,4833,647 |
| unsigned int | 4 bytes | 0 to 4,294,967,295 |
| long | 4 bytes | -2,147,4833,648 to +2,147,4833,647 |
| Unsigned long | 4 bytes | 0 to 4,294,967,295 |

***Table 4: Data types and size***

***Other Data Types***

* ***Char Data Type***
* Usually 1 byte long
* Internally stored as an integer
* ASCII character set shows integer representation for each character
* ‘A’ == 65, ‘B’ == 66, ‘C’ == 67, etc
* Single quotes denote a character, double quotes denote a string
* ***Boolean Data Type***

Boolean variables are set to either true or false

**OPERATORS**

There are many operators in C++ for manipulating data which include arithmetic Operators, Relational Operators, Logical operators and many more which will be discussed accordingly.

* ***Arithmetic Operators***

|  |  |
| --- | --- |
| **Operator** | **Description** |
| + | Addition |
| - | Subtraction |
| \* | Multiplication |
| / | Division |
| % | Modulo |

***Table 5: Arithmetic Operators***

* ***Relational Operators***

|  |  |
| --- | --- |
| **Operator** | **Description** |
| = = | Equals to |
| != | Not Equals to |
| < | Less than |
| > | Greater than |
| <= | Less than or equal to |
| >= | Greater than or equal to |

***Table 6: Relational Operators***

* ***Logical Operators***

|  |  |
| --- | --- |
| **Operator** | **Description** |
| && | Logical AND |
| | | | Logical OR |
| ! | NOT |

***Table 7: Logical Operators***

* ***Increment and Decrement Operators***

C++ introduces increment and decrement operators which are ++ and – respectively. These operators increment/decrement 1 in the operand’s value.

For example: x++ will be equivalent to x=x+1 or x+=1.

The special characteristic of these operators is that they can be used for pre-increment as well as post-increment. To understand, consider the following statements:

A=b++; *//The statement will assign the contents of b to A and then increments the value of b by 1*

A=++b; *//The statement will first increment the value of b by 1 and then assign the new value to A.*