|  |  |  |  |
| --- | --- | --- | --- |
| Date: | **22-06-2020** | Name: | **Varun G Shetty** |
| Course: | **C ++ programming** | USN: | **4AL17EC093** |
| Topic: | * Basic concepts * Conditionals | Semester & Section: | **6th & ‘B’** |
| GitHub Repository: | **Varunshetty4** |  |  |

**BASIC CONCEPTS:**

**Introduction:** C++ is a general-purpose programming language. C++ is used to create computer programs. C++ offers various headers, each of which contains information needed for programs to work properly. This particular program calls for the header **<iostream>**. The **number sign (#)** at the beginning of a line targets the compiler's pre-processor. In this case, **#include** tells the pre-processor to include the **<iostream>** header. The **<iostream>** header defines the standard stream objects that input and output data. In our code, the line **using namespace std;** tells the compiler to use the **std**(standard) **namespace**. The **std**namespace includes features of the C++ Standard Library.

Curly brackets { } indicate the beginning and end of a function, which can also be called the function's body. The entry point of every C++ program is **main()**, irrespective of what the program does. **cout**is used in combination with the insertion operator. Write the insertion operator as **<<** to insert the data that comes after it into the stream that comes before. In C++, the **semicolon**is used to terminate a statement. Each statement must end with a **semicolon**. It indicates the end of one logical expression.

A **block**is a set of logically connected statements, surrounded by opening and closing curly braces. You can have multiple statements on a single line, as long as you remember to end each statement with a **semicolon**. Failing to do so will result in an error. The line **return 0;** terminates the **main()** function and causes it to return the value 0 to the calling process. A non-zero value (usually of 1) signals abnormal termination.

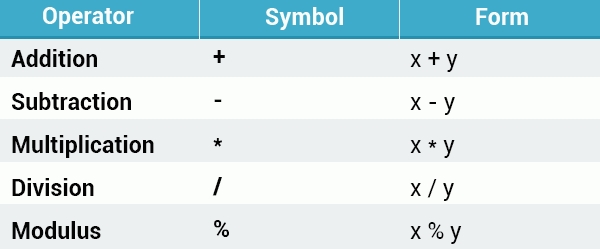
**Newline:** The **cout**operator does not insert a line break at the end of the output.  
One way to print two lines is to use the **endl**manipulator, which will put in a line break. The new line character **\n** can be used as an alternative to **endl**. The backslash (\) is called an **escape character**, and indicates a "special" character. Two newline characters placed together result in a blank line.

**Comments**: A comment beginning with **two slashes (//)**is called a single-line comment. The slashes tell the compiler to ignore everything that follows, until the end of the line. Comments that require multiple lines begin with **/\*** and end with **\*/.** Within a comment marked with /\* and \*/, // characters have no special meaning, and vice versa. This allows you to "nest" one comment type within the other.

**Variable:** C++ offer a rich assortment of built-in as well as user defined **data types**. **Integer**, a built-in type, represents a whole number value. Define integer using the keyword **int**.  
C++ requires that you specify the **type**and the **identifier**for each variable defined. An **identifier**is a name for a variable, function, class, module, or any other user-defined item. An identifier starts with a letter (A-Z or a-z) or an underscore (\_), followed by additional letters, underscores, and digits (0 to 9). The C++ programming language is **case-sensitive**, so **myVariable**and **myvariable**are two different identifiers. Specifying the data type is required just once, at the time when the variable is declared.  
After that, the variable may be used without referring to the data type.**int** a;

**User Input:** To enable the user to input a value, use **cin**in combination with the extraction operator (**>>**). The variable containing the extracted data follows the operator. As with **cout**, extractions on **cin**can be chained to request more than one input in a single statement: cin >> a >> b;

**Basic Arithmetic:** C++ supports these arithmetic operators.



The division operator divides the first operand by the second. Any remainder is dropped in order to return an integer value. If one or both of the operands are floating point values, the division operator performs floating point division.

**Operator Precedence**: If none of the expressions are in parentheses, **multiplicative**(multiplication, division, modulus) operators will be evaluated before **additive**(addition, subtraction) operators.

**Assignment and Increment Operator:** The simple **assignment**operator (=) assigns the right side to the left side. C++ provides shorthand operators that have the capability of performing an operation and an assignment at the same time. The **increment**operator is used to increase an integer's value by one, and is a commonly used C++ operator. **Prefix**increments the value, and then proceeds with the expression. **Postfix**evaluates the expression and then performs the incrementing. The **decrement**operator (--) works in much the same way as the increment operator, but instead of increasing the value, it decreases it by one.

**Relational operators**: Same as C

**Conditionals:**

1. If statement

|  |
| --- |
| if (condition) {  statements  } |

1. If-else statement

|  |
| --- |
| if (condition) {  //statements  }  else {  //statements  } |

You can also use nested if that is if inside if and also nested if else statements.

**Loops:**

1. While loop

|  |
| --- |
| while (condition) {  statement(s);  } |

1. Do-while loop

|  |
| --- |
| do {  statement(s);  } while (condition); |

1. For loop

|  |
| --- |
| for ( init; condition; increment ) {  statement(s);  } |

1. Switch statement

|  |
| --- |
| switch (expression) {  case value1:  statement(s);  break;  case value2:  statement(s);  break;  ...  case valueN:  statement(s);  break;  } |

With no specified **break**statement, the statements continued to run after the matching case. Thus, all the other case statements printed. This type of behavior is called **fall-through**. As the switch statement's final case, the **default**case requires no **break**statement. The **break**statement can also be used to break out of a loop.

Logical Operators:

1. Logical AND &&
2. Logical OR ||
3. Logical NOT !