**Operators and Types**

Operators are special symbols that tells the compiler to perform specific mathematical or logical functions

C++ divides the operators into following groups:

* Arithmetic operators
* Comparison or Relational operators
* Logical operators
* Assignment operators
* Bitwise operators

1. **Arithmetic Operators:**

Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication, division, and modulus.

The following are arithmetic operators in C++:

* Addition (+): Adds two operands.
* Subtraction (-): Subtracts the second operand from the first.
* Multiplication (\*): Multiplies two operands.
* Division (/): Divides the first operand by the second.
* Modulus (%): Returns the remainder of the division operation.
* Increment (++): Increases the value by one
* Decrement (--): Decreases the value by one

1. **Comparison or Relational Operators:**

Comparison operators are used to compare two operands. They return a Boolean value (true or false) based on the comparison. The following are comparison operators in C++:

* Equal to (==): Checks if two operands are equal.
* Not equal to (!=): Checks if two operands are not equal.
* Greater than (>): Checks if the left operand is greater than the right.
* Less than (<): Checks if the left operand is less than the right.
* Greater than or equal to (>=): Checks if the left operand is greater than or equal to the right.
* Less than or equal to (<=): Checks if the left operand is less than or equal to the right.

1. **Logical Operators:** Logical operators are used to perform logical operations on Boolean operands. They return a Boolean value as a result. The following are logical operators in C++:

* Logical AND (&&): Returns true if both operands are true.
* Logical OR (||): Returns true if at least one of the operands is true.
* Logical NOT (!): Returns true if the operand is false and vice versa.

1. **Assignment Operators:**

Assignment operators are used to assign values to variables. They can also perform operations while assigning values. The following are assignment operators in C++:

* Simple assignment (=): Assigns the value of the right operand to the left operand.
* Addition assignment (+=): Adds the value of the right operand to the left operand and assigns the result to the left operand.
* Subtraction assignment (-=): Subtracts the value of the right operand from the left operand and assigns the result to the left operand.
* Multiplication assignment (\*=): Multiplies the left operand by the value of the right operand and assigns the result to the left operand.
* Division assignment (/=): Divides the left operand by the value of the right operand and assigns the result to the left operand.
* Modulus assignment (%=): Calculates the modulus of the left operand with the value of the right operand and assigns the result to the left operand.

1. **Bitwise Operators:**

Bitwise operators are used to perform operations on individual bits of integer operands. They are used in low-level programming and bitwise manipulation. The following are bitwise operators in C++:

* Bitwise AND (&): Performs a bitwise AND operation.
* Bitwise OR (|): Performs a bitwise OR operation.
* Bitwise XOR (^): Performs a bitwise XOR (exclusive OR) operation.
* Bitwise NOT (~): Performs a bitwise NOT (complement) operation.
* Left shift (<<): Shifts the bits of the left operand to the left by the number of positions specified by the right operand.
* Right shift (>>): Shifts the bits of the left operand to the right by the number of positions specified by the right operand.

**Escape Sequence**

In C++, an escape sequence is a combination of characters that represents a special character or a control sequence. Escape sequences are used to perform actions such as printing special characters like newline (\n), tab (\t), or backspace (\b), or to represent characters that are difficult to type or invisible (such as the backslash itself).

Here are some common escape sequences in C++:

* \n: Newline. Moves the cursor to the beginning of the next line.
* \t: Tab. Moves the cursor to the next tab stop.
* \b: Backspace. Moves the cursor back one position.
* \\: Backslash. Prints a backslash character.
* \': Single quote. Prints a single quote character.
* \": Double quote. Prints a double quote character.
* \0: Null character. Represents the null terminator character ('\0').

Escape sequences are typically used in strings, character literals, and character constants.

**Difference between “\n” and “endl”**

Please check the following lines of code:

* cout<<“C++ \n” <<“is a programming language \n”<<“ok”;
* cout<<“C++”<<endl<<“programming “<<endl<<“language”;

These two lines of code will give the same output, the only difference is:

* “endl” is a constant symbol and given in cout stream.
* “\n” is inserted into the constant string in double quotation.

**Cin or Object Input stream or Console Input**

* The “cin” stands for console input, pronounced as “see in”.
* It is used to get input from the keyboard during execution of the program.
* computer waits to receive an input from keyboard
* The keyword cin (pronounced “C in”) is an object, predefined in C++ to correspond to the standard input stream.
* The >> is the extraction or get from operator.
* It takes the value from the stream object on its left and places it in the variable on its right.

**Increment and Decrement Operators**

**Increment Operator (++):**

The increment operator is used to increase the value of a variable by 1.

It can be used in two ways: prefix and postfix.

* Prefix Increment (++var): The value of var is incremented first, and then the updated value is used in the expression.
* Postfix Increment (var++): The current value of var is used in the expression, and then var is incremented.

**Decrement Operator (--):**

The decrement operator is used to decrease the value of a variable by 1.

Like the increment operator, it can be used in two ways: prefix and postfix.

* Prefix Decrement (--var): The value of var is decremented first, and then the updated value is used in the expression.
* Postfix Decrement (var--): The current value of var is used in the expression, and then var is decremented.