Logical Operators

Logical operators operate only on Boolean values (or expressions like relational operators that return Boolean values) and yield a Boolean result of their own. The operators used for logical computation in C++ are !, &&, and ||.

Using Logical Operators in C++?

As we'll see, logical operators are well suited for checking the validity of two (or more) comparative operations. The operators then output a specific response based on the nature of the operator and whether one or both operands are true. In C++, we often see this in the form of an if/else statement.

Before we can take a closer look at where logical operators often show up in code, we'll first need to understand the syntax behind them. There are a total of three logical operators:

The "and" (&&) Operator

The logical "and" operator looks at the operands on either of its sides and returns "true" only if both statements are true. If even just one of the two statements is false, the logical "and" will return false.

Below is a practical example of how we can use the && operator in C++:

```
#include <iostream>
using namespace std;


int main()
{
   cout << "Enter a number: ";
   int num {};
   cin >> num;


   if (num > 0 && num <= 10)
      cout << "Your number is between 1 and 10";
   else
      cout << "Your number is not between 1 and 10";
   return 0;
}
```

The “or” (||) Operator

The logical “or” operator works similarly to the” and” operator above. The difference is that “or” will return “true” if either the left or the right operand is true. The || operator will only return a false value if both operands are false.

Consider a scenario where picking one of two lucky numbers between 1 and 10 in a game will win us a prize. For this example, we’ll set the lucky numbers to four and eight. We’ll need to create a program in C++ to check for winners:

```
#include <iostream>
using namespace std;

int main() {

 cout << "Enter a number: ";
   int num {};
   cin >> num;
   if(num == 4 || num == 8)
      cout << "You chose a winning number!";
   else
      cout << "Sorry, better luck next time.";
   return 0;
}
```

The “not” (!) Operator

The “not” logical operator is used to convert a value from true to false, or from false to true. Similarly, if an operand evaluates to true, a logical “not” would cause it to evaluate to false. If an operand evaluates to false, its logical “not” equivalent would be true.

The following example reveals one possible use for the logical “not” operator:

```
#include <iostream>
using namespace std;

int main()
{
   cout << "Enter a number: ";
   int x {};
   cin >> x;

   if (!x == 0)
      cout << "You typed a number other than 0";
   else
      cout << "You typed zero";

   return 0;
}
```

Q/determine whether a number is even

```
#include <iostream>
bool isEven(int x)
{
   // if x % 2 == 0, 2 divides evenly into our number, which means it must be an even number
   return (x % 2) == 0;
}
int main()
{
   std::cout << "Enter an integer: ";
   int x{};
   std::cin >> x;

   if (isEven(x))
      std::cout << x << " is even\n";
   else
      std::cout << x << " is odd\n";
 return 0;
}
```

Switch Selection Statement (Selection)

The switch statement is an alternate syntax for performing actions based on the value of an expressions. In C++, the expression of a switch statement must be of an integral type, a type convertible to an integral type, an enumerated type and must be compared to constants. Each constant value represents a “case”. If the expression matches the case, the subsequent lines of code are executed until a break statement is reached. You can also provide a default case, which is matched if none of the other cases match. The following code shows a common use of the switch statement:

```
Switch(menuItem){
Case OpenMenuItem:

//Code to open a file

break;
Case SavemenuItem:

//Code to save a file

break;
default:
break;
}
```

A switch statement can always be converted into if/else statements. The previous switch statement can be converted as follows:

```
If ( menuItem == OpenMenuItem)

{
//Code to open a file

else if ( MenuItem == SaveMenuItem)

{
```

// Code to save a file

}

else {

// Code to give an error message

}

General Form of Switch Selection statement:

```
switch ( selector )
{
    case label1  :  statement1 ; break;
    case label2  :  statement2 ; break;
    case label3  :  statement3 ; break;
      :
    case label-n :  statement-n ; break;
    default  :  statement-e ;  break;
}
```

```
switch (value) {
 case 0: cout << "grade is A"; break;
case 1: coucout << "grade is B"; break;
 case 2: coucout << "grade is C"; break;
default: cout << "grade is X"; break;
)
```

Example 1

Write C++ program to read integer number, and print the name of the day in a week:

```
#include<iostream.h>
void main( )
{
        int day;
         cout << "Enter the number of the day \n";
        cin >> day;
        switch (day)
{
   case 1:   cout << "Sunday";        break;
   case 2:   cout << "Monday";        break;
   case 3:   cout << "Tuesday";       break;
   case 4:   cout << "Wednesday";     break;
   case 5:   cout << "Thursday";      break;
   case 6:   cout << "Friday";        break;
   case 7:   cout << "Saturday";      break;
   default:  cout << "Invalid day number";    break;
}
}
```

Example 2

Write C++ program to read two integer numbers, and read the operation to perform on these numbers:

```
#include<iostream.h>
void main( )
{
        int a, b;
         char x;

         cout << "Enter two numbers \n";
        cin >> a >> b;

          cout << "+  for addition \n";
          cout << "-   for subtraction \n";
          cout << "*  for multiplication \n";
          cout << "/  for division \n";
          cout << "enter your choice \n";
        cin >> x;

        switch ( x )
        {
            case '+': cout << a + b;
                      break;
```

```
            case '-': cout << a - b;
                      break;
            case '*': cout << a * b;
                      break;
            case '/': cout << a / b;
                      break;
            default: break;
        }
}
```

Write c++ program to read integer number and print the equivalent string

#include<iostream>

Int Main() {

Int x;

Cin>> x;

Switch(x)

Case0:cout<<"Zero"; break;

Case1:cout<<"one"; break;

.

.

.

Default: cout<<"invalid number";}

Number Validation Program using Switch Case Statement in C++:

```
#include <iostream>
using namespace std;
void main()
{
	int x;
	cin >> x;
	switch (x)
	{
	case 1:
		cout << "One"<<endl;
		break;
	case 2:
		cout << "Two" << endl;
		break;
	default:
		cout << "Out of Range" << endl;
		break;
	}
	system ("pause");
}
```

```
#include <iostream>
using namespace std;
void main()
{
	char ch;
	cin >> ch;
	switch (ch)
	{
	case '+':
		cout << "pluse"<<endl;
		break;
	case '-':
		cout << "minuse" << endl;
		break;
	default:
		cout << "Out of Range" << endl;
		break;
	}
	system ("pause");
}
```

Create a Calculator using the switch Statement

```
Program to build a simple calculator using switch Statement
#include <iostream>
using namespace std;

int main() {
    char oper;
    float num1, num2;
    cout << "Enter an operator (+, -, *, /): ";
    cin >> oper;
    cout << "Enter two numbers: " << endl;
    cin >> num1 >> num2;

    switch (oper) {
        case '+':
            cout << num1 << " + " << num2 << " = " << num1 + num2;
            break;
        case '-':
            cout << num1 << " - " << num2 << " = " << num1 - num2;
            break;
        case '*':
            cout << num1 << " * " << num2 << " = " << num1 * num2;
            break;
        case '/':
            cout << num1 << " / " << num2 << " = " << num1 / num2;
            break;
        default:
            // operator is doesn't match any case constant (+, -, *, /)
            cout << "Error! The operator is not correct";
            break;
    }

    return 0;
}
Run Code
```

Output 1

```
Enter an operator (+, -, *, /): +
Enter two numbers:
2.3
4.5
2.3 + 4.5 = 6.8
```

Output 2

```
Enter an operator (+, -, *, /): -
Enter two numbers:
2.3
4.5
2.3 - 4.5 = -2.2
```

Output 3

```
Enter an operator (+, -, *, /): *
Enter two numbers:
2.3
4.5
2.3 * 4.5 = 10.35
```

Output 4

```
Enter an operator (+, -, *, /): /
Enter two numbers:
2.3
4.5
2.3 / 4.5 = 0.511111
```

Output 5

```
Enter an operator (+, -, *, /): ?
Enter two numbers:
2.3
4.5
Error! The operator is not correct.
```

```cpp
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string name;
    cout << "Enter name:";
    getline (cin, name);
    cout << "Hello " << name;
    return 0;
}
```

Rewrite the following code fragment so that a switch is used instead of the if/else statements.

```
#include <iostream>
using namespace std;
int main()
{
  int value;
  char ch;
  cin >> ch;
    switch(ch)
    {
    case 'A':
      value =10;
      break;
    case 'P':
      value = 20;
      break;
    case 'T':
      value = 30;
      break;
    case 'V':
      value = 40;
      break;
    default:
      value = 50;

    return 0;
    }
    cout<< "value= " << value;
}
```