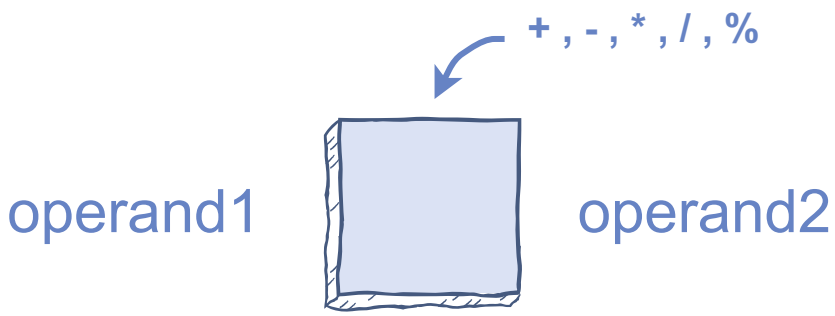
Arithmetic Operators

In this lesson, we will discuss arithmetic operators.

We'll cover the following

Introduction to arithmetic operators

Arithmetic operators are used to perform numeric operations on operands.



Here is the list of arithmetic operators available in C++:

Operator	Operation	Use
+	Addition	Adds operand1 and operand2
_	Subtraction	Subtracts operand2 from operand1
*	Multiplication	Multiplies operand1 and operand2
/	Division	Divides operand1 by operand2
%	Modulus	Returns remainder after dividing operand1 by operand2

%	Modulus	Returns remainder after dividing operand1 by opeand2

Example program with `int` operands

Consider two operands of type `int`: The value of `operand1` is `50`, and the value of `operand2` is `26`. Let's apply each arithmetic operator on them.

Run the code below and see the output!

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

int main() {
  // Initilaize operand1 and operand2
  int operand1 = 50;
  int operand2 = 26;
  // Prints value of operand1 and operand2
  cout << "Values of operands are:" << endl;
  cout << "operand1 = " << operand1 << " , operand2 = " << operand2 << endl;
  // Adds operand1 and operand2; and print their result
  cout << "Addition = " << operand1 + operand2 << endl;
  // Subtracts operand1 and operand2; and print their result
  cout << "Subtraction = " << operand1 - operand2 << endl;
  // Multiplies operand1 and operand2; and print their result
  cout << "Multiplication = " << operand1 * operand2 << endl;
  // Divides operand1 and operand2; and print their result
  cout << "Division = " << operand1 / operand2 << endl;
  // Returns remainder of operand1 and operand2; and print it
  cout << "Modulus = " << operand1 % operand2 << endl;
  return 0;
}
```

Result of `/` operator

All the operators in C++ show the same results as expected by the calculator except the division operator. If you put `50/26` in a calculator, it returns `1.92307692308` in output. Whereas, our C++ program is returning `1` in the output. So, why is the C++ division operator showing different behavior from the calculator's division operator?

The reason is the data type of our operands is `int`, so our output is of type `int`. Therefore, C++ only gives you the whole number part of the quotient, excluding the remainder to keep the type consistent. If you want a quotient with a fractional part, use the operands of `float` or `double` data type.

Example program with `float` operands

Consider two operands of type `float`: The value of `operand1` is `50.0`, and the value of `operand2` is `26.0`.

Press the **RUN** button and see the output!

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

int main() {
  // Initilaize operand1 and operand2
 float operand1 = 50.0;
 float operand2 = 26.0;
  // Prints value of operand1 and operand2
  cout << "Values of operands are:" << endl;
  cout << "operand1 = " << operand1 << " , operand2 = " << operand2 << endl;
  // Adds operand1 and operand2; and print their result
  cout << "Addition = " << operand1 + operand2 << endl;
  // Subtracts operand1 and operand2; and print their result
  cout << "Subtraction = " << operand1 - operand2 << endl;
  // Multiplies operand1 and operand2; and print their result
  cout << "Multiplication = " << operand1 * operand2 << endl;
  // Divides operand1 and operand2; and print their result
  cout << "Division = " << operand1 / operand2 << endl;
  // Returns remainder of operand1 and operand2; and print it
  //cout << "Modulus = " << operand1 % operand2 << endl;
  return 0;
}
```

If you run the code above, we get the same results as expected by the calculator.

Using `%` with `float` operands

Uncomment `line No 20` in the above code and see the output. It generates an error.

Using a `mod` operator with floating-point operands generates an error. We can only use the `mod` operator with operands of integer type.

> We can also apply arithmetic operators to the operands of the `char` data type. In this case, operators operate upon the ASCII value of the characters.

Q

What is the output of the following code?

```
int main() {
  char operand1 = 'n';
  char operand2 = 'C';
  cout << "Division = " << operand1 / operand2 << endl;
  cout << "Modulus = " << operand1 % operand2 << endl;
  return 0;
}
```

See this link for ASCII values.

Retake Quiz

This sums up our discussion of arithmetic operators. Let’s discuss the assignment and compound assignment operators in the next lesson.

See you there!