Calculator Version 1

My first calculator ever in C++. Basic division, addition, multiplication, and subtraction functions. It'll prompt the user to enter values for x and y, using cout and cin.

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

float x, y;

void division(){
      float quotient = x / y;
      cout << "Quotient: " << quotient <<end1;
}

void addition(){
      float sum = x + y;
      cout << "Sum: " << sum <<end1;
}

void multiplication(){
      float product = x * y;
      cout << "Product: " << product <<end1;
}

void subtraction(){
      float difference = x - y;
      cout << "Difference: " << difference <<end1;
}

int main(){

      cout << "Enter the value of x: ";
      cin >> x;
      cout << "Enter the value of y: ";
      cin >> y;

      division();
      addition();
      multiplication();
      subtraction();

      return 0;
}
```

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

`#include <iostream>  //library`

- Include the input/output stream library in your program.(Syntax)
- The “iostream” library provides the functionality for basic input and output operations in C++.

`using namespace std;`

- Tells the compiler (program that translates source code into binary) to use the std namespace by default.
- std namespace contains all the standard C++ library functions and objects.
- Without this, you would have to prefix `std::` to every standard function or object, Ex: `std::cout << “Hello, World!” << std: :end1;`
  `std::cin >> x;`

```
float x, y;
```

`float`

- It is a data type in C++ that stores floating-point numbers (#’s with decimals). Examples are: 3.14 and -0.5
- Used in operations that can result in non-integer values (10/4 = 2.5)

`x, y;`

- Are variables of a type of float
- Variables are name storage locations in memory that hold a value.
- Purpose: used to store the two numbers entered by the user.
- When they are outside of any function, they become **global variables**. They can be accessed and modified by any function in the program. Ex: division(), addition()

`float x, y;`

- Declares two floating-point variables, x and y.
- Are global variables, meaning they can be accessed by all functions!

```
void division(){
      float quotient = x / y;
      cout << "Quotient: " << quotient <<end1;
}
```

void

- A return type function, which means it does not return any value.
- Instead of returning a value, it performs an action. (In this case, it's calculating and printing the quotient, difference, sum, and product)

Parentheses **()**

- After the function name, to indicate that this function does not take any parameters (inputs), it relies on the global variables (x and y) for its calculations.

quotient

- The name of the variable that will hold the result of the division.
- It's a **local variable**, meaning it exists within the **division()** function and cannot be accessed outside of it.

x / y

- The division operation, which divides the value of **x** and **y**.

float quotient = x / y;

- This line performs the division of **x** and **y** and stores the result in a variable named **quotient**.

cout

- Stands for "console output" and is used to print text or values to the terminal. (A part of the iostream library)

<<

- The **stream insertion operator**, which sends data to the **cout** stream for display.

"Quotient: "

- This is a string that provides context for the output. It tells the user that the value being displayed is the quotient.

`endl`

- Stands for “end line” and moves the cursor to the next line in the terminal after printing the output.

`cout << “Quotient: “ << quotient <<endl;`

- Outputs the result of the division to the console with the label “Quotient: “.

```
int main(){

     cout << “Enter the value of x: “;
     cin >> x;
     cout << “Enter the value of y: “;
     cin >> y;

     division();
     addition();
     multiplication();
     subtraction();

     return 0;
}
```

`int main(){`

- An entry point of the program.
- Every C++ program starts with execution from the `main()` function.
- `int` returns an integer (usually `0`), which signals to the operating system that the program was executed successfully.

`cout << “Enter the value of x: “;`

- Outputs the message to the console and prompts the user to input the variable for `x`.

`cin`

- Stands for “console input” and takes the input from the user.

`cin >> x;`

- This line reads the value from the user and stores it in the global variable x.
- Extracts the input from the `cin` stream and assigns it to the variable `x`.

`division();`

- Calls the `division()` function, which calculates the quotient between `x` and `y`, and prints the result.

`return 0;`

- This line ends the `main()` function and returns the value `0` to the operating system. Returning `0` indicates it was successful.