You can pass command-line arguments in a Java program. You can do the same thing in C++. To do so, create a main function with the following header:

```c
int main(int argc, char* argv[])
```

Where `argv` specifies the arguments and `argc` the number of the arguments. The following command line, for example, starts the program `TestMain` with three strings: `arg1`, `arg2`, and `arg3`:

```
TestMain arg1 arg2 arg3
```

`arg1`, `arg2`, and `arg3` are strings and are passed to `argv`. `argv` is an array of C-strings. In this example, `argc` is 4 because three string arguments are passed and the program name `TestMain` also counts as an argument.

The arguments must be strings, but they don’t have to appear in double quotes on the command line. The strings are separated by a space. A string that contains a space must be enclosed in double quotes. Consider the following command line:

```
TestMain "First num" alpha 53
```

It starts the program with three strings: "First num" and `alpha`, and `53`, a numeric string. Note that `53` is actually treated as a string. You can use "53" instead of `53` in the command line.

Listing 1 presents a program that performs binary operations on integers. The program receives three arguments: an integer followed by an operator and another integer. For example, to add two integers, use this command:

```
Calculator 1 + 2
```

The program will display the following output:

```
1 + 2 = 3
```

Figure 1 shows sample runs of the program.
Figure 1

The program takes three arguments (operand1 operator operand2) from the command line and displays the expression and the result of the arithmetic operation.

Here are the steps in the program:
1. Check \texttt{argc} to determine whether three arguments have been provided in the command line. If not, terminate the program using \texttt{exit(0)}.
2. Perform a binary arithmetic operation on the operands \texttt{args[1]} and \texttt{args[3]} using the operator specified in \texttt{args[2]}.

Listing 1 Calculator.cpp

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

int main(int argc, char * argv[])
{
    // Check number of strings passed
    if (argc != 4)
    {
        cout << "Usage: Calculator operand1 operator operand2";  
        exit(0);  
    }
```
// The result of the operation
int result = 0;

// Determine the operator
switch (argv[2][0])
{
    case '+':
        result = atoi(argv[1]) + atoi(argv[3]);
        break;
    case '-':
        result = atoi(argv[1]) - atoi(argv[3]);
        break;
    case '*':
        result = atoi(argv[1]) * atoi(argv[3]);
        break;
    case '/':
        result = atoi(argv[1]) / atoi(argv[3]);

    // Display result
}
In the sample run, "*" had to be used instead of * for the command

    Calculator 12 "*" 3

In C++, the * symbol refers to all the files in the current directory when it is used on a command line. Therefore, in order to specify the multiplication operator, the * must be enclosed in quote marks in the command line. The following program displays all the files in the current directory when issuing the command Test *:

Listing 2 Test.cpp

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

int main(int argc, char * argv[]) {
    for (int i = 0; i < argc; i++)
        cout << argv[i] << endl;
}
```