

Project 2: Reverse Polish Notation

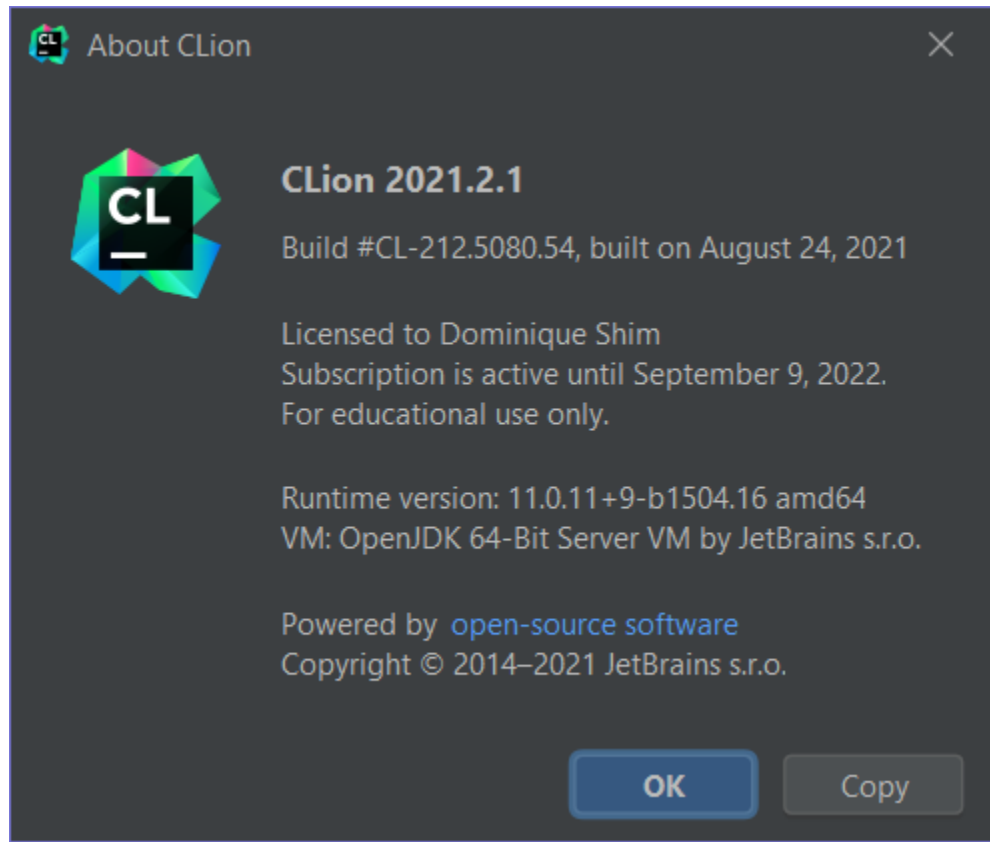
Name: Dominique Shim

Section: 02

IDE Used

IDE: CLion

IDE Version: 2021.2.1



Project 2: Reverse Polish Notation

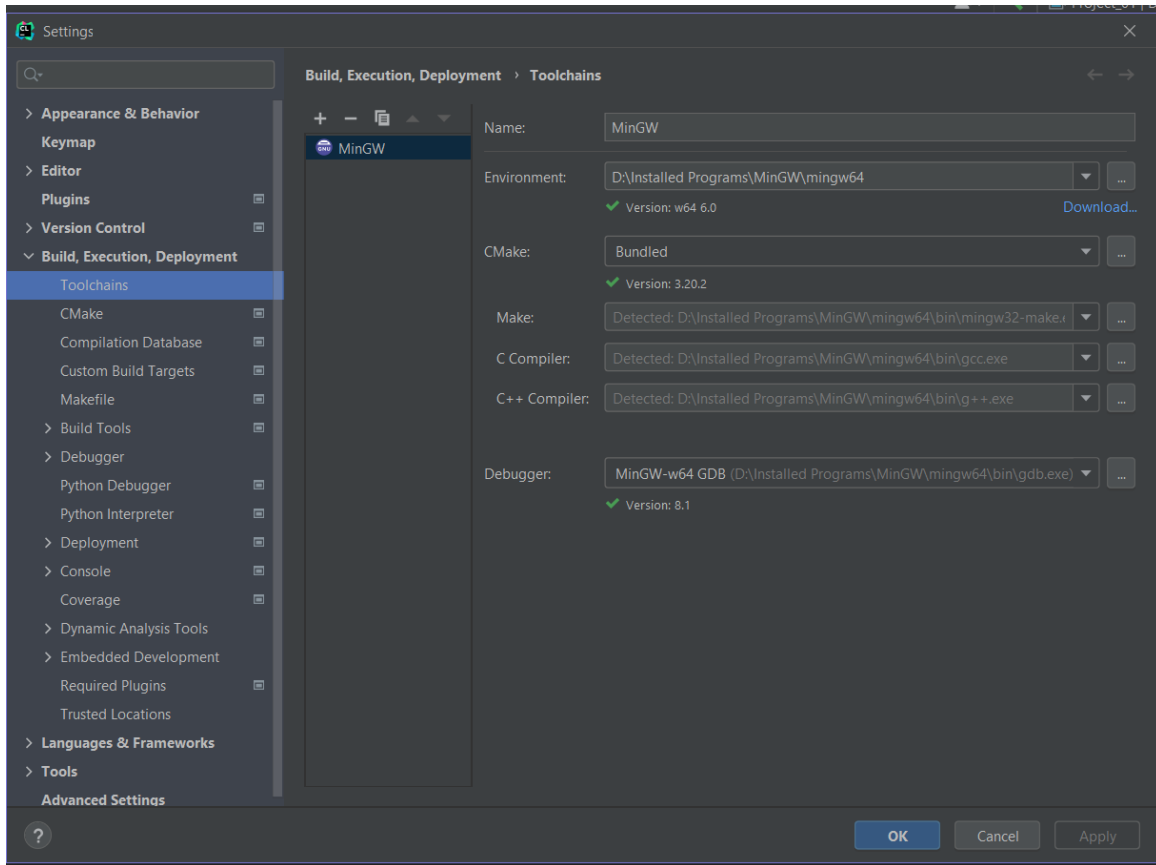
Name: Dominique Shim

Section: 02

Environment

Environment for IDE: MinGW

Compile: G++



Environment for command line: MinGW

Compiler: gcc

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Dominique> gcc --version
gcc.exe (MinGW.org GCC-6.3.0-1) 6.3.0
Copyright (C) 2016 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

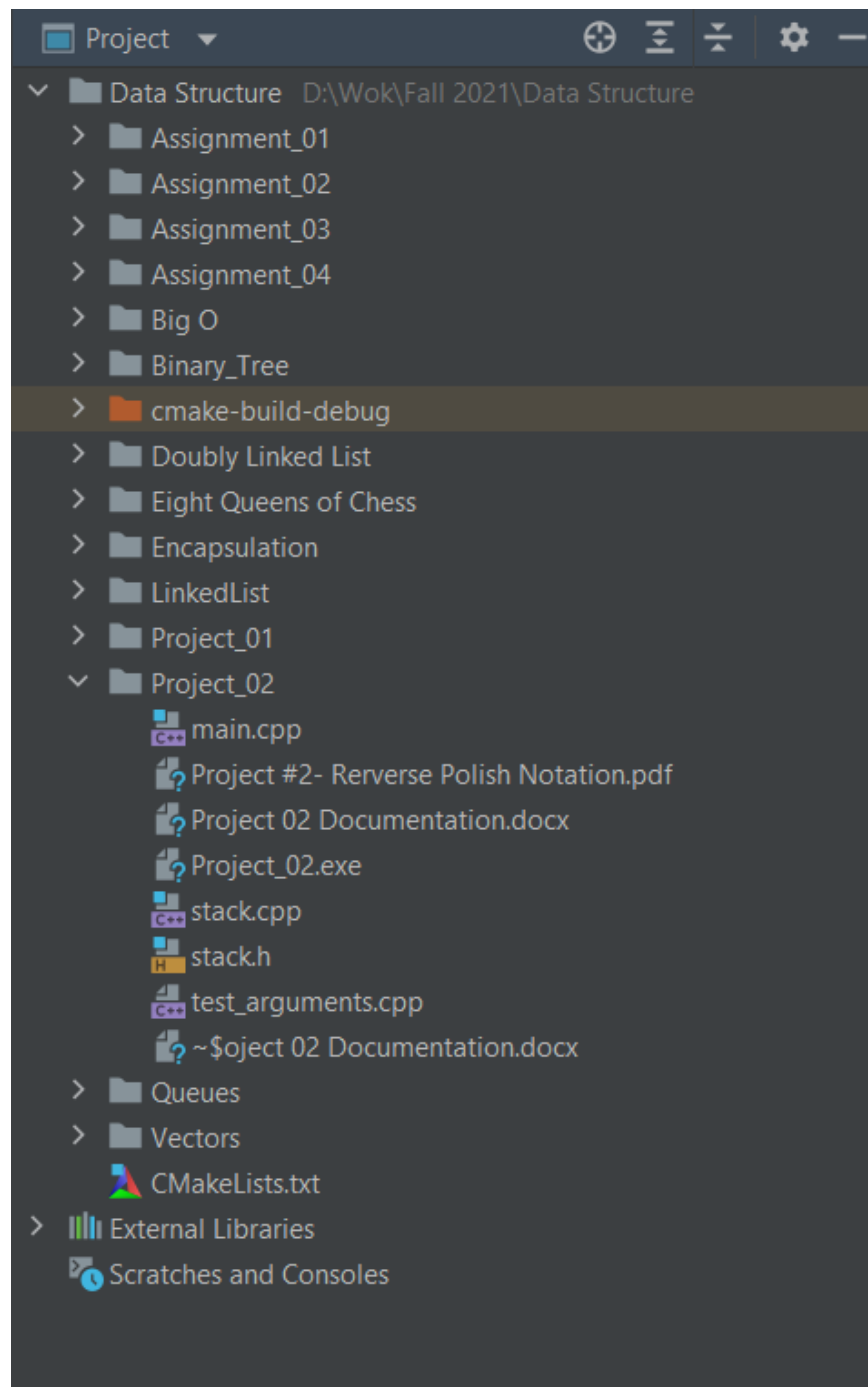
PS C:\Users\Dominique>
```

Project 2: Reverse Polish Notation

Name: Dominique Shim

Section: 02

File Directory

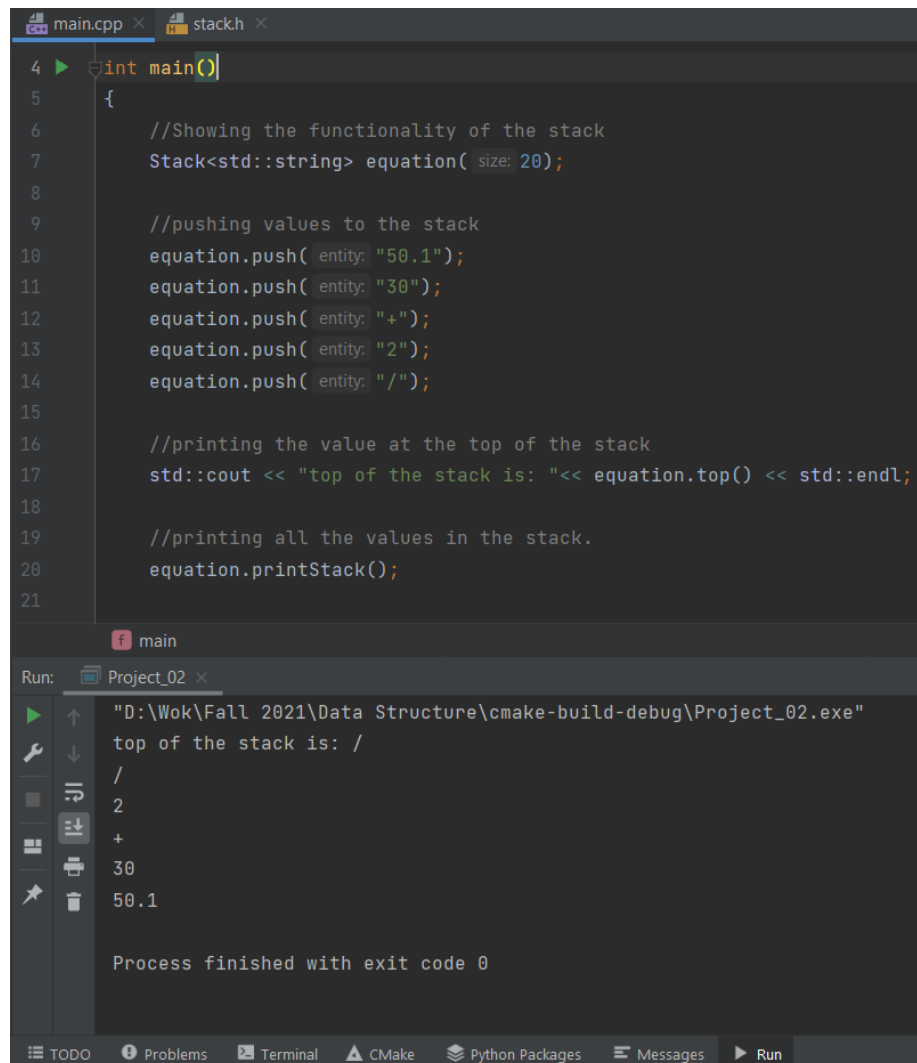


Project 2: Reverse Polish Notation

Name: Dominique Shim

Section: 02

Testing



The screenshot displays a C++ IDE with two tabs: `main.cpp` and `stack.h`. The `main.cpp` file contains the following code:

```
4 int main()
5 {
6     //Showing the functionality of the stack
7     Stack<std::string> equation( size: 20);
8
9     //pushing values to the stack
10    equation.push( entity: "50.1");
11    equation.push( entity: "30");
12    equation.push( entity: "+");
13    equation.push( entity: "2");
14    equation.push( entity: "/");
15
16    //printing the value at the top of the stack
17    std::cout << "top of the stack is: "<< equation.top() << std::endl;
18
19    //printing all the values in the stack.
20    equation.printStack();
21 }
```

Below the code editor, the `Run` panel shows the output of the program:

```
"D:\Wok\Fall 2021\Data Structure\cmake-build-debug\Project_02.exe"
top of the stack is: /
/
2
+
30
50.1

Process finished with exit code 0
```

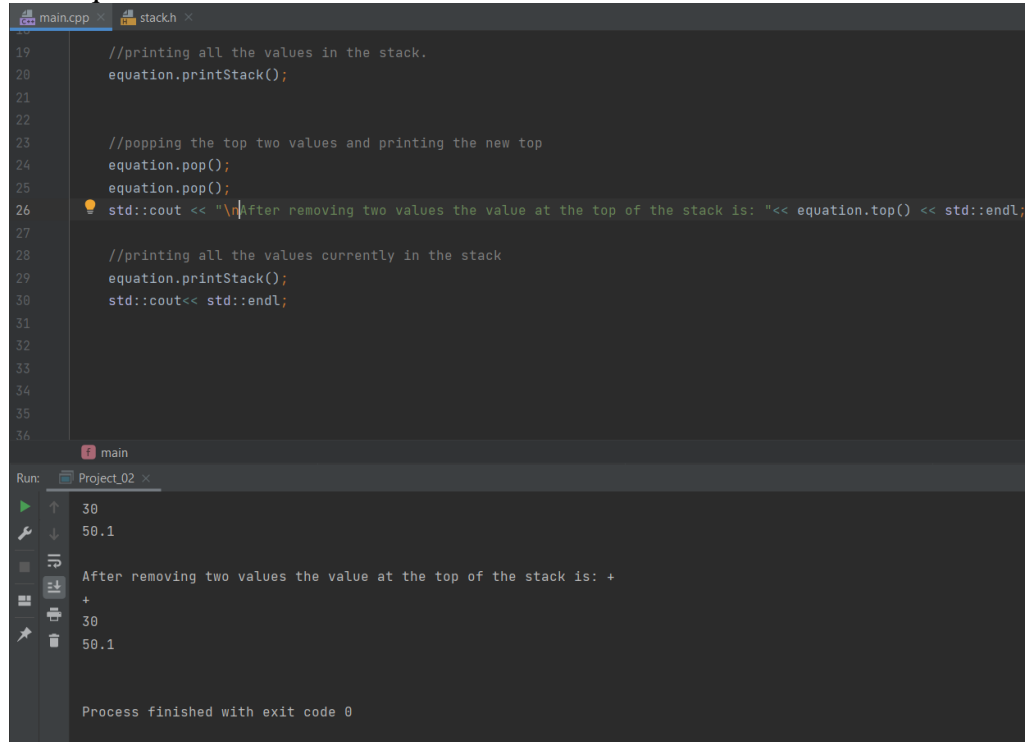
The IDE interface includes a sidebar with icons for navigation and a bottom status bar with tabs for `TODO`, `Problems`, `Terminal`, `CMake`, `Python Packages`, `Messages`, and `Run`.

Adding values to the stack and first printing the value at the top then printing all the values in the stack.

Project 2: Reverse Polish Notation

Name: Dominique Shim

Section: 02



The screenshot shows a C++ IDE with two tabs: 'main.cpp' and 'stackh'. The 'main.cpp' tab is active, displaying the following code:

```
19 //printing all the values in the stack.  
20 equation.printStack();  
21  
22  
23 //popping the top two values and printing the new top  
24 equation.pop();  
25 equation.pop();  
26 std::cout << "\nAfter removing two values the value at the top of the stack is: "<< equation.top() << std::endl;  
27  
28 //printing all the values currently in the stack  
29 equation.printStack();  
30 std::cout<< std::endl;  
31  
32  
33  
34  
35  
36
```

Below the code editor, the 'Run' tab is active, showing the output of the program:

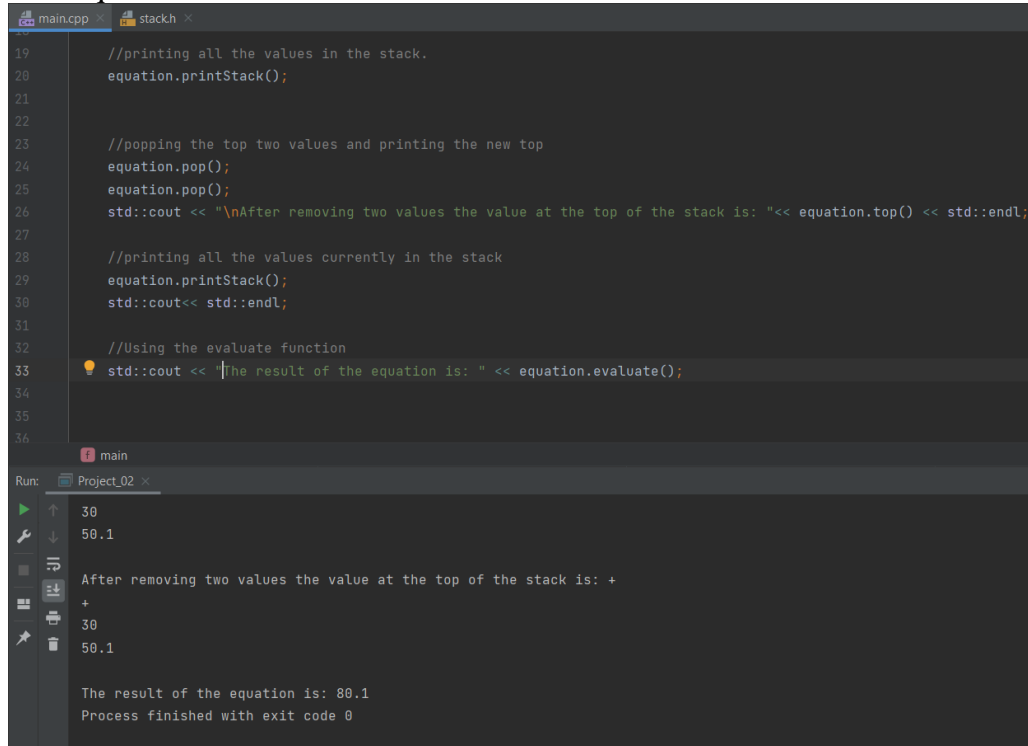
```
Run: Project_02 x  
30  
50.1  
  
After removing two values the value at the top of the stack is: +  
+  
30  
50.1  
  
Process finished with exit code 0
```

Continuing from the previous instance, pop was called twice and the top value was printing along with all the values in the stack.

Project 2: Reverse Polish Notation

Name: Dominique Shim

Section: 02



The screenshot shows a C++ IDE with two tabs: 'main.cpp' and 'stackh'. The 'main.cpp' tab is active, displaying code from line 19 to 36. The code includes comments and function calls for printing the stack, popping values, and evaluating the equation. The 'Run' window at the bottom shows the output of the program, which includes the stack contents before and after popping, and the final result of the equation evaluation.

```
19 //printing all the values in the stack.  
20 equation.printStack();  
21  
22  
23 //popping the top two values and printing the new top  
24 equation.pop();  
25 equation.pop();  
26 std::cout << "\nAfter removing two values the value at the top of the stack is: " << equation.top() << std::endl;  
27  
28 //printing all the values currently in the stack  
29 equation.printStack();  
30 std::cout << std::endl;  
31  
32 //Using the evaluate function  
33 std::cout << "The result of the equation is: " << equation.evaluate();  
34  
35  
36
```

main

Run: Project_02

30
50.1
+
30
50.1
The result of the equation is: 80.1
Process finished with exit code 0

Printing the result after running the evaluate function.

Project 2: Reverse Polish Notation

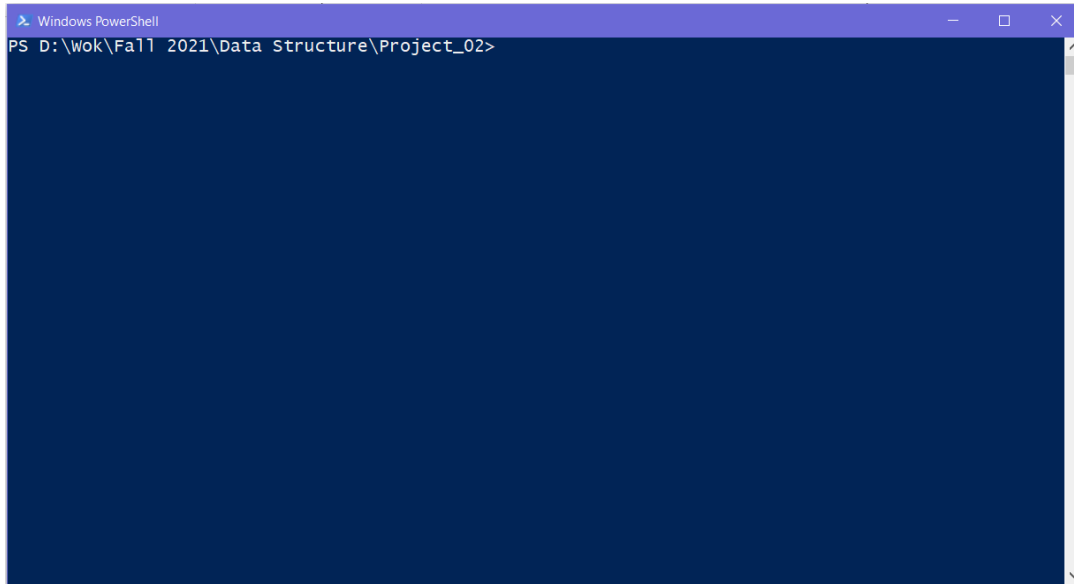
Name: Dominique Shim

Section: 02

How to Use Command Line to Run Program

Step 1.

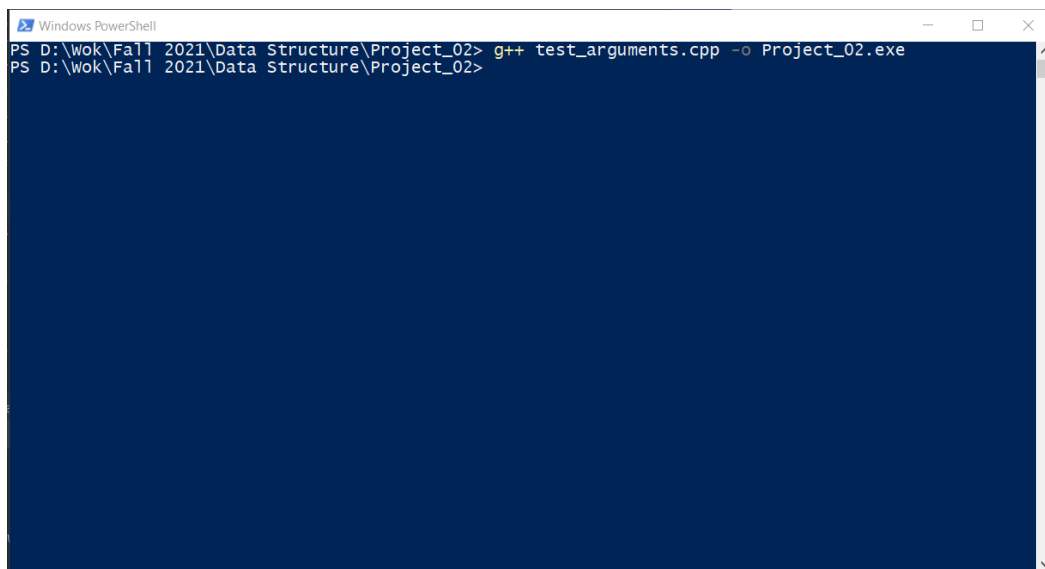
Open the command line in the same folder as the files



```
Windows PowerShell
PS D:\Wok\Fall 2021\Data Structure\Project_02>
```

Step 2.

Run the command “g++ test_arguments.cpp -o Project_02.exe” to create an .exe file called Project_02.exe. This file will appear in the folder that the command line is currently located









```
Windows PowerShell
PS D:\Wok\Fall 2021\Data Structure\Project_02> g++ test_arguments.cpp -o Project_02.exe
PS D:\Wok\Fall 2021\Data Structure\Project_02>
```

Project 2: Reverse Polish Notation

Name: Dominique Shim

Section: 02

Name	Date modified	Type	Size
 main.cpp	11/8/2021 6:49 PM	JetBrains CLion	2 KB
 Project #2- Rerverse Polish Notation.pdf	10/26/2021 3:33 PM	Adobe Acrobat D...	292 KB
 Project_02.exe	11/8/2021 6:56 PM	Application	123 KB
 stack.cpp	11/2/2021 2:37 PM	JetBrains CLion	1 KB
 stack.h	11/8/2021 6:56 PM	JetBrains CLion	7 KB
 test_arguments.cpp	11/8/2021 6:51 PM	JetBrains CLion	1 KB

Step 3.

In the command line type “.\\Project_02.exe argument1 argument2 ... argumentn”. The arguments should be in the order of polish notation.

```
Windows PowerShell
PS D:\Wok\Fall 2021\Data Structure\Project_02> .\\Project_02.exe 400 40 100 x + 30 + 60 4 / -
4415
PS D:\Wok\Fall 2021\Data Structure\Project_02>
```


Project 2: Reverse Polish Notation

Name: Dominique Shim

Section: 02

NOTE

When using multiply, use lower case x (x) rather than asterisk (*). Using an asterisk will be treated as a wild card in the command line and return all the file names as an argument