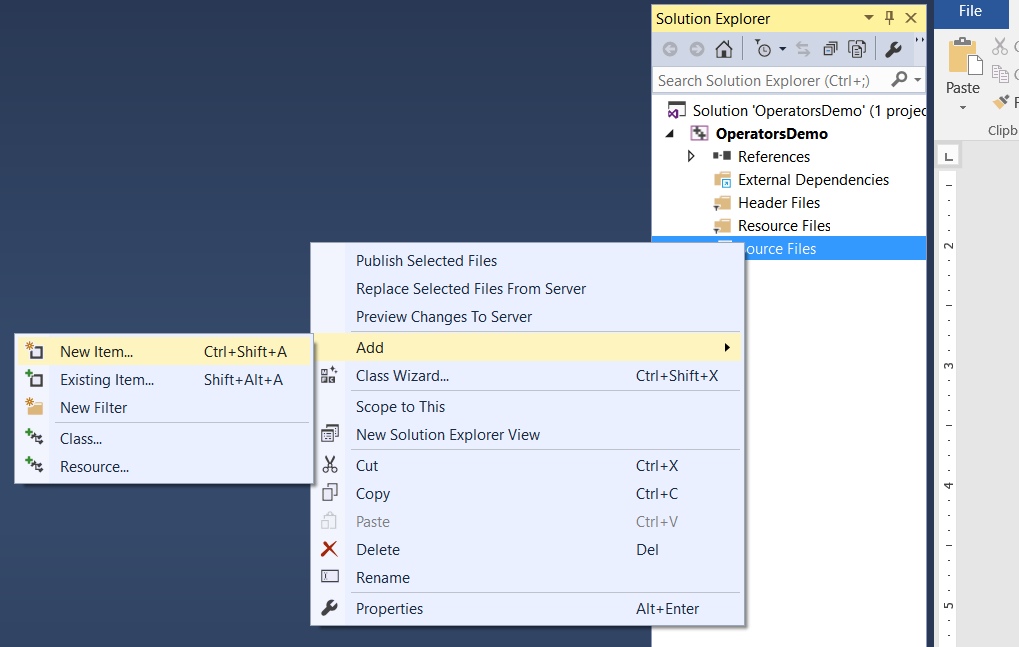
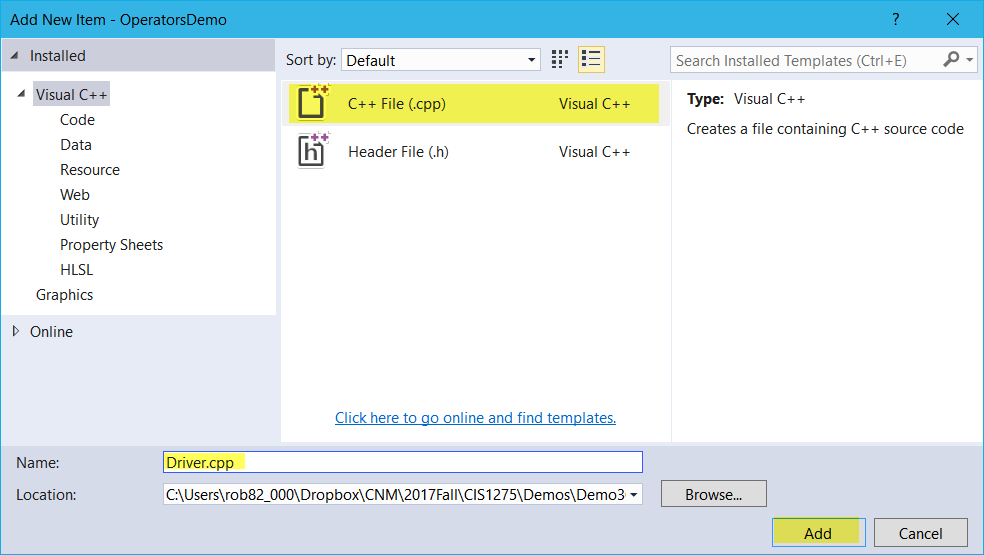
OPERATORS Demo

Video at <https://youtu.be/PI7DwjYJK9Q>

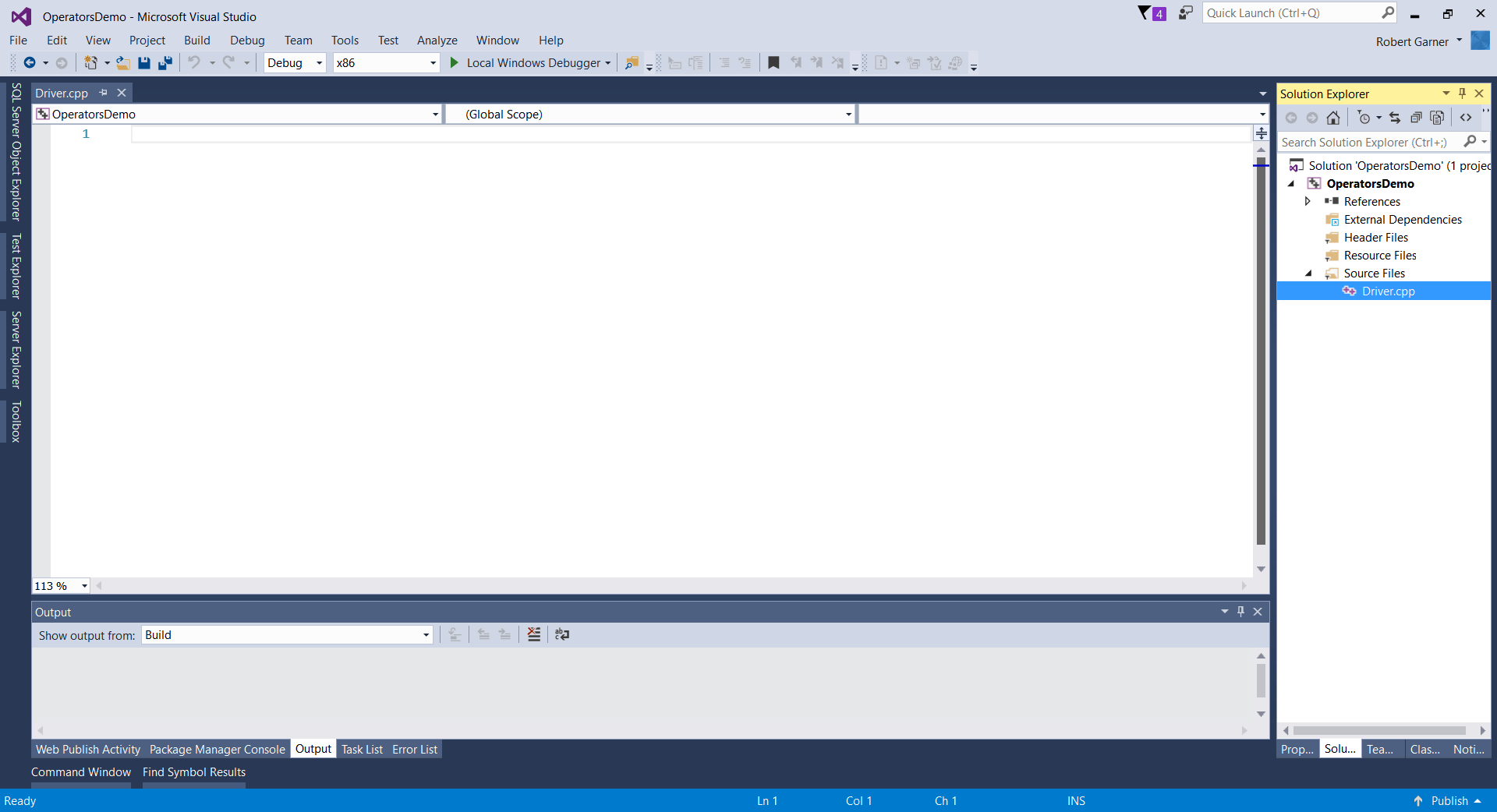
1. Start Visual Studio
2. Create a new empty C++ project and call it “OperatorsDemo”.
3. Click on Source Files and Add|New Item:



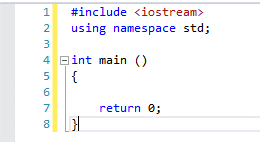
1. Select Visual C++, C++ File (.cpp), call it “Driver.cpp”, then click “Add”:



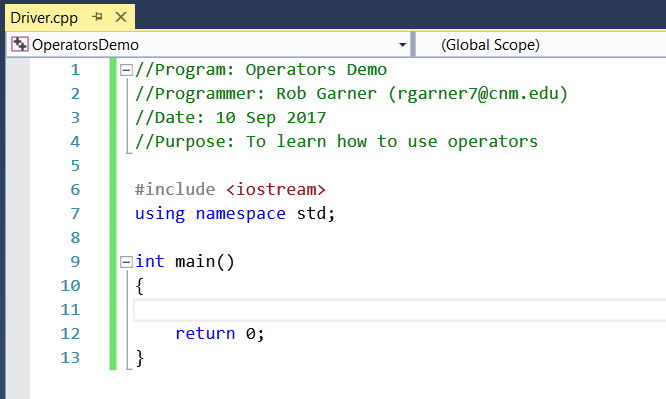
1. You should now have “Driver.cpp” in the solution explorer and it should be open in the main window:



1. Enter the following code to start our program:

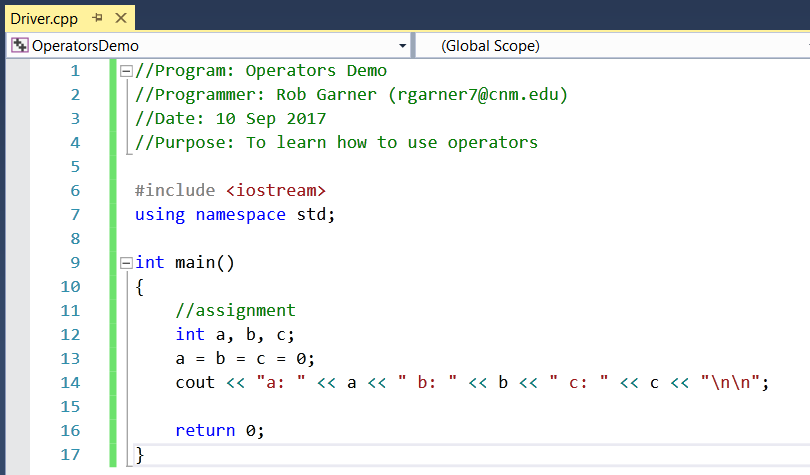


1. Take a look at this code and memorize it. We will be starting all of our programs this way. We #include the iostream library so we can send text to the screen and read text from the keyboard. We use namespace std so we can deconflict with other files we may create later. We then have to have a main function so the operating system will know where to start our program. Main will return an int of 0 so the operating system knows our program executed OK. Let’s add some comments:

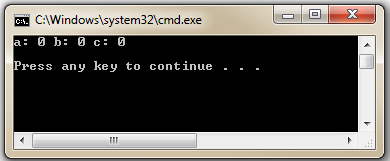


* 1. Comments can be either block comments that start with “/\*” and end with “\*/” or single line comments that just start with “//”. The picture above highlights the characters that designate the beginning and end of the block comment at the top and the beginning of the single line comment at the bottom. Single line comments don’t need anything specific at the end because everything to the end of the line is considered part of the comment.
  2. Comments do not affect execution of the program. They are just there to help you and, more importantly programmers who may have to modify your code in the future, to understand how your code works. You must comment you code!
  3. You should have a comment block at the start of the program with:
     1. Name of the program
     2. Your name and email address
     3. Date
     4. Purpose of program
  4. Points will be taken off of programs that are not adequately commented.

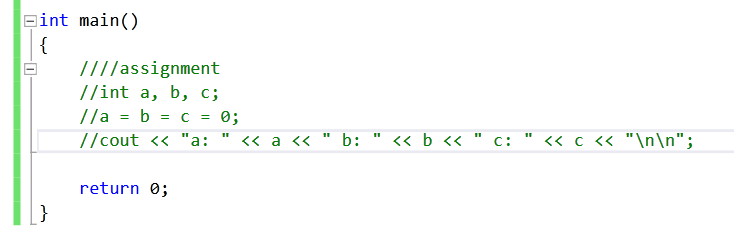
1. Multiple assignments on one line:
   1. Add the following code:



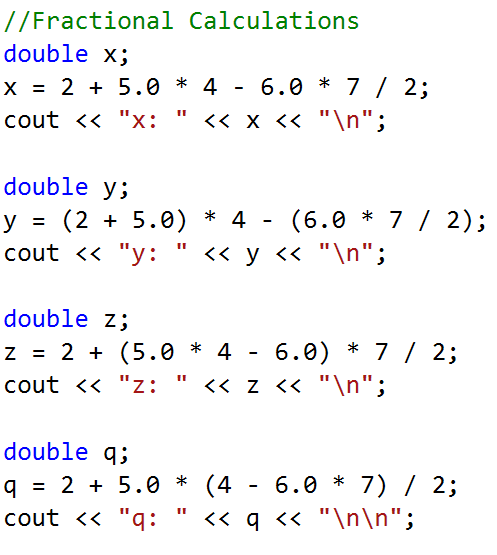
* 1. Test it by clicking on Ctrl-F5. Compile if required.



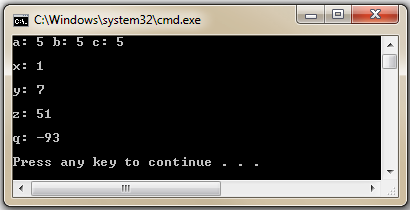
* 1. Notice that all three variables a, b, c are assigned the value 0;
  2. Change the 0 to a 5 in the code and run it again.
  3. Comment out everything in this section in preparation for the next section (to comment out multiple lines highlight the lines you want to comment out then click Ctrl-K-C, to uncomment click Ctrl-K-U)).



1. Fractional Calculations:
   1. Add the following code:



* 1. Try it with Ctrl-F5



* 1. Comment out everything in this section in preparation for the next section (to comment out multiple lines highlight the lines you want to comment out then click Ctrl-K-C, to uncomment click Ctrl-K-U)).

1. Wrong and Right Way:
   1. Enter the following code:

//Wrong and Right ways

double v,

a = 1,

b = 2,

c = 3,

d = 4;

v=(a+b)(c-d);

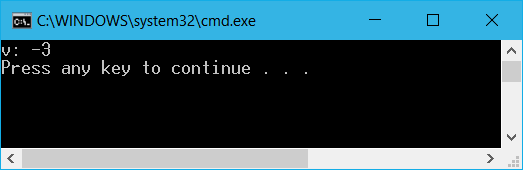
cout << "v: " << v << "\n";

* 1. Notice the error you get!
  2. Comment out the assignment operation and change it to:

//wrong: v=(a+b)(c-d);

v=(a+b)\*(c-d);

* 1. Try it with Ctrl-F5



* 1. Compare the difference between the following two formulas:

v=a+b/c-d;

v=(a+b)/(c-d);

Why are they different?

* 1. Enter the following code:

double SA, pi = 3.14, rad = 5.5;

SA = (pi)(rad)(rad)

cout << "SA: " << SA << "\n";

* 1. What errors do you get?
  2. Fix the code and make it work.
  3. Which of the following work?:

SA = pi\* rad \*\* 2;

SA = pi \* rad ^ 2;

SA = pi \* rad \* rad;

SA = pi \* pow(rad,2);

* 1. Next add the following code:

double m,

x = 1,

y = 2,

w = 3;

m = sqrt(x.3y)/w;

cout << "m: " << m << "\n";

* + 1. Do you get errors? Note . is not multiplication in C++. C++ math notation is different than standard algebra notation.
    2. Try fixing the code (hint use \* instead of .)
  1. Next add the following code:

double angle = 0.5 , f(angle);

f(angle) = 2/3 sin (x-0.3);

cout << "angle: " << angle << "\n";

* + 1. What errors do you get?
    2. Try the following:

double angle = 0.5 , fofangle;

fofangle = 2.0 /3.0 \* sin (x-0.3);

cout << " fofangle: " << fofangle << "\n";

* + 1. Why does this work?
  1. Take a look at other formulas available at <http://msdn.microsoft.com/en-us/library/system.math.aspx>
  2. Comment out everything in this section in preparation for the next section.

1. Increment and Decrement Operators
   1. Add the following code:

int i = 5;

i = i + 1;

cout << "i: " << i << "\n";

* 1. Notice i is one more than it started as
  2. Replace the formula with:

i = ++i;

* 1. Notice this works the same way
  2. Try

i = i++;

* 1. Notice this works the same as well but what is the difference?
  2. Replace the code with the following:

int i = 5;

cout << "i: " << i++ << "\n";

cout << "i: " << i << "\n";

* 1. Notice that i++ increments i within the cout command
  2. Now try:

int i = 5;

cout << "i: " << ++i << "\n";

cout << "i: " << i << "\n";

* 1. What’s the difference?
  2. Try:
  3. Now try:

int i = 5;

cout << "i: " << i-- << "\n";

cout << "i: " << i << "\n";

* 1. And
  2. Now try:

int i = 5;

cout << "i: " << --i << "\n";

cout << "i: " << i << "\n";

* 1. Delete or comment out everything in this section in preparation for the next section.

1. Accumulation Operators
   1. Add the following code:

int sum = 7;

sum = sum + 10;

cout << "sum: " << sum << "\n";

* 1. Now try:

int sum = 7;

sum += 10;

cout << "sum: " << sum << "\n";

* 1. Now try:

int diff = 7;

diff = diff - 10;

cout << " diff: " << diff << "\n";

* 1. Now try:

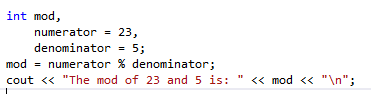
int diff = 7;

diff -= 10;

cout << " diff: " << diff << "\n";

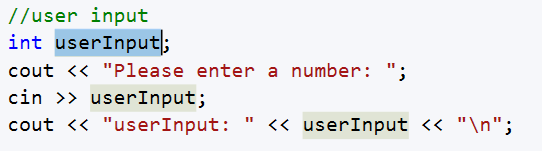
* 1. Comment out everything in this section in preparation for the next section.

1. Modulus Operation
   1. Try the following code:

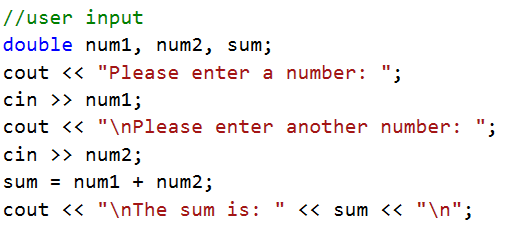


* 1. Why does this return 3?
  2. The modulus is the remainder after a division of two numbers is completed so 23 divided by 5 is 4 with a remainder of 3. 23 mod 5 therefore yields 3. Try other numbers like 20 and 5 or 17 and 2.

1. Get user input
   1. Try the following code:



* 1. What is the output if you enter 7 when it asks you to enter a number?
  2. “cin” allows you to get user input. In this case userinput is brought in as an integer. Try the program with a variety of response both numbers and text. Notice that C++ will take anything in but it doesn’t always make sense.
  3. Make a simple calculator:



* 1. Run it enter integers, doubles and characters. Change double to int and again try integers, double and characters. What happens? Why?