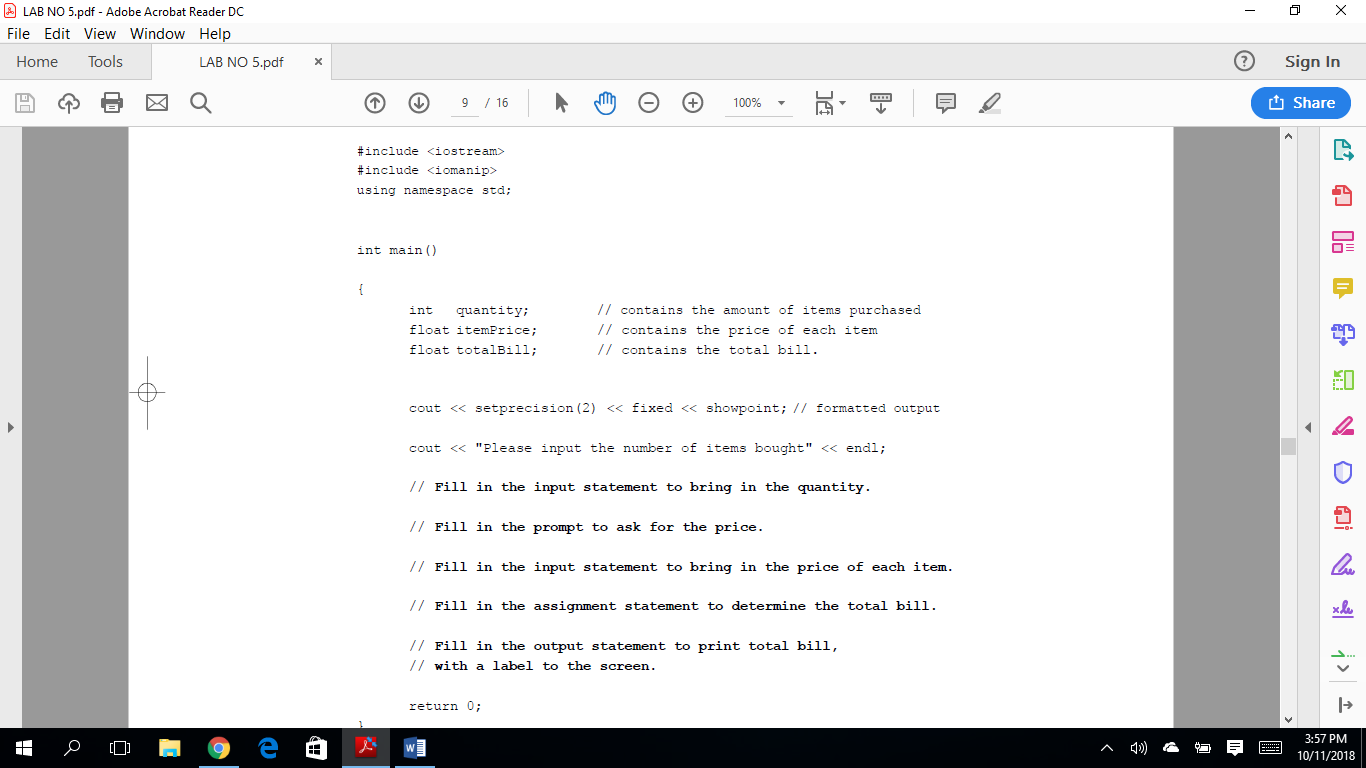
**Supporting File for Lab No. 6**

**LAB 6.1 Working with the cin Statement**

Compile the code is listed below:



Exercise 1: Complete the program so that a sample run inputting 22 for the number of items bought and 10.98 for the price of each item will produce the results below.

Sample run of the program.

**Please input the number of items bought**

22

**Please input the price of each item**

**10.98**

**The total bill is $241.56**

Exercise 2: Once you have the program working, change the instruction:

cout << setprecision (2) << fixed << showpoint;

to

cout << setprecision(2) << showpoint;

Rerun the program with the same data given in Exercise 1 above and record your results. What do you think the fixed attribute in the cout statement does?

Exercise 3: Now put the fixed attribute back in and change the instruction to make the precision 4. Rerun the program with the same data given in Exercise 1 and record your results. What do you think the setprecision( ) attribute in the cout statement does?

**The attribute showpoint forces all floating point output to show a decimal point even if the values are whole numbers. In some environments this is done automatically.**

**LAB 6.2 Formatting Output**

Look at the following table:

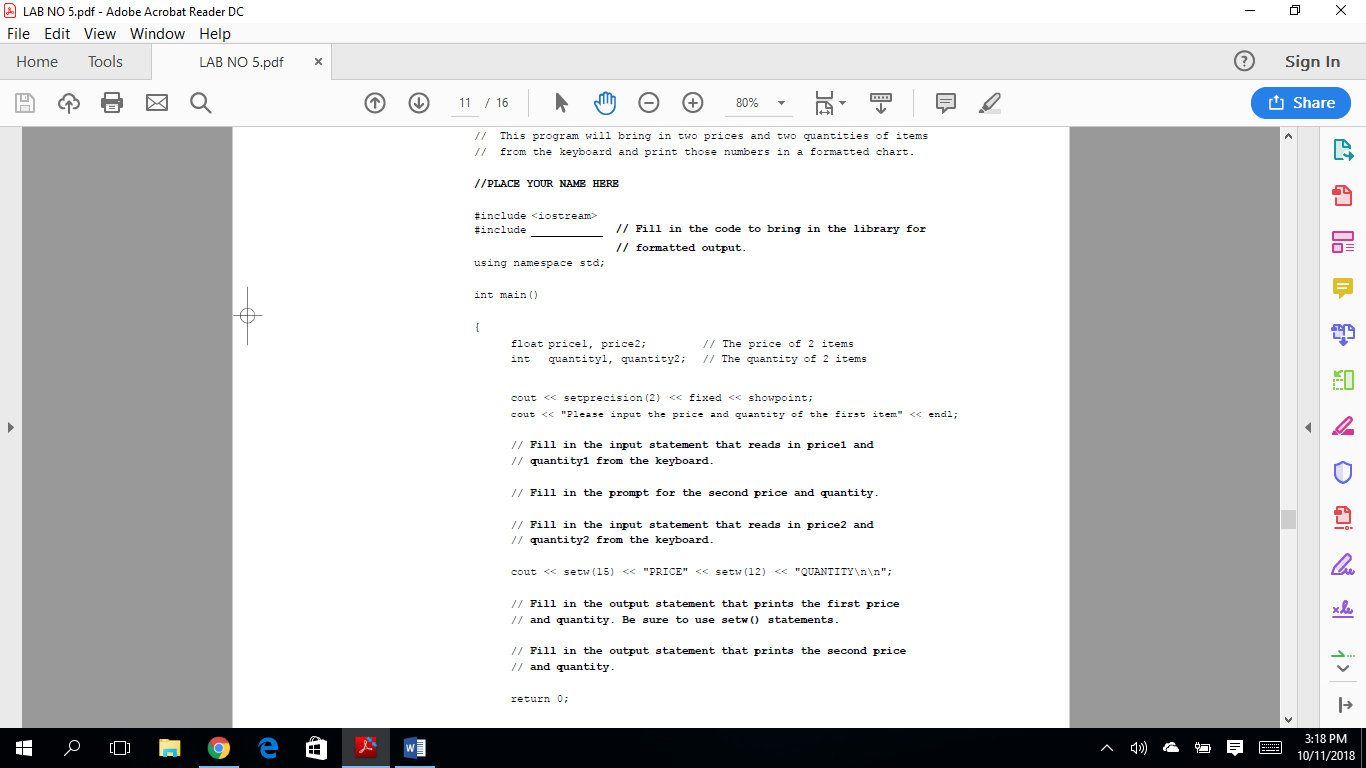
PRICE QUANTITY

1.95 8

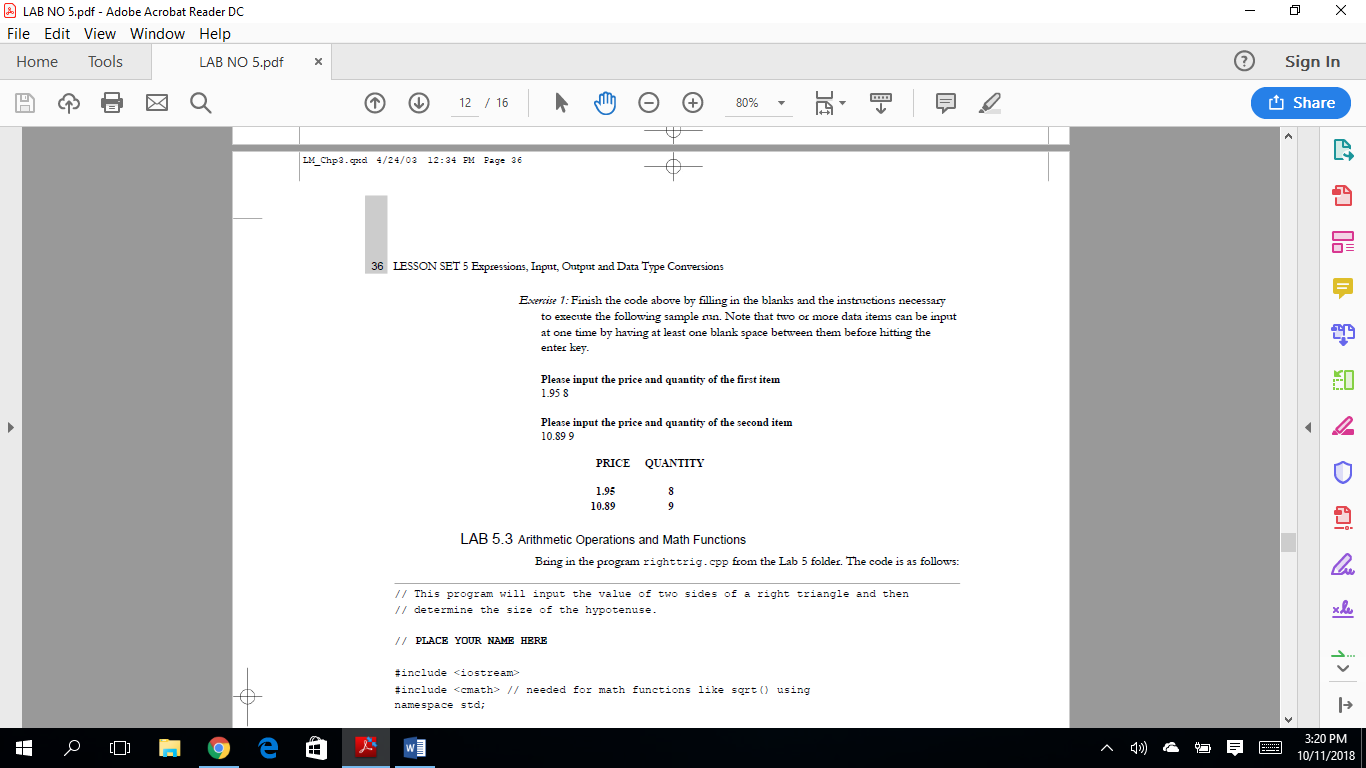
10.89 9

Assume that from the left margin, the price takes up fifteen spaces. We could say that the numbers are right justified in a 15-width space. Starting where the price ends, the next field (quantity) takes up twelve spaces. We can use the formatted output from Lab 6.1 and the statement setw(n) where n is some integer to indicate the width to produce such tables.

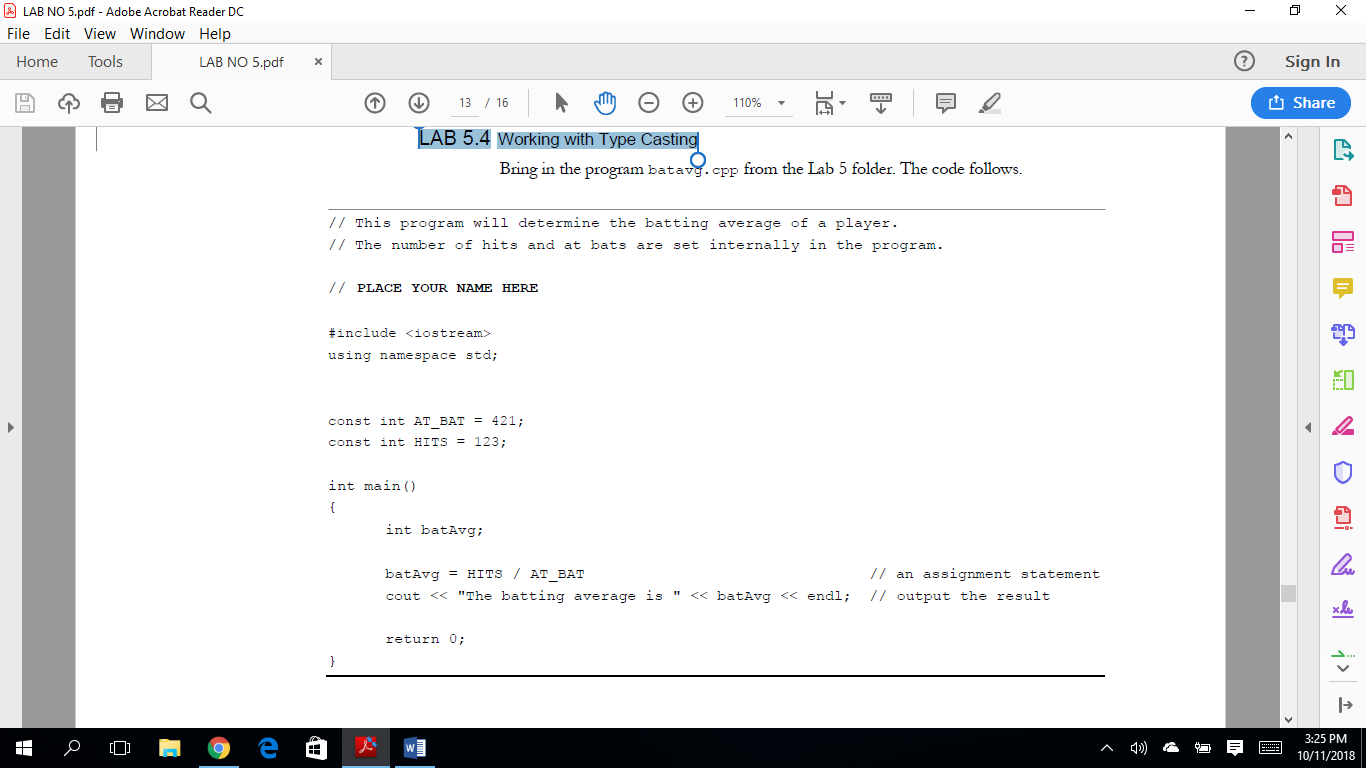
The code is as follows:



*Exercise 1:* Finish the code above by filling in the blanks and the instructions necessary to execute the following sample run. Note that two or more data items can be input at one time by having at least one blank space between them before hitting the enter key.



**LAB 6.3 Working with Type Casting**



*Exercise 1:* Run this program and record the results. The batting average is .

*Exercise 2:* There is a logic error in this program centering around data types. Does changing the data type of batavg from int to float solve the problem? Make that change and run the program again and record the result.

The batting average is .

*Exercise 3:* Continue to work with this program until you get the correct result. The correct result should be 0.292162. Do not change the data type of the two named constants. Instead, use a typecast to solve the problem.