*C++ requirements*

*You are not allowed to use any global variables. Use of global variables will result in a grade of zero for part 1.*

Make sure you exactly match any function signatures that are required by the exercise. The function signatures are:

double readSeconds()

double calculateDistance(double seconds)

void displayResults(double seconds, double distance)

The program must use type *double* for calculations.

Failure to follow the C++ requirements could reduce the points received from passing the tests.

*General overview*

Your program will calculate the distance an object travels (in meters) on Earth for a specified number of seconds.

Your program must have the main function and three additional functions. The signatures for these functions must be as follows:

double readSeconds()

double calculateDistance(double seconds)

void displayResults(double seconds, double distance)

The readSeconds function will be an input function that will read in a double value from cin and return that value back to main.

ThecalculateDistance function will calculate the distance an object falls (on earth) during the specified number of seconds.

The displayResults function that will display both the number of seconds an object has fallen as well as the distance the object has fallen.

Here is a summary of the processing that is required in the various functions:

*double readSeconds()*

This function reads in a value from cin. If the value is less than zero the function should output a message.

The value read in (valid or not) should be returned to the calling function.

The prompt from the function should be:

Enter the time (in seconds)

If the value is less than zero you should output the following message.

The time must be zero or more

*double calculateDistance(double seconds)*

This function calculates the distance traveled (on earth) during the number of seconds pass in as a parameter. The distance is calculated in meters and is returned to the calling function.

The formula is:

*d = 0.5 \* g \* pow(t, 2)*

Where *d* is distance (in meters), *t* is time (in seconds) and *g* is 9.8 (the acceleration due to gravity on the earth).

Use good variable names and not just *d*, *g* and *t*. Use double values for your calculations.

*void displayResults(double seconds, double distance)*

The displayResults function takes two parameters of type double. The first is the number of seconds and the second is the distance traveled.

The output is the text:

The object traveled xx.xxx meters in yy.yy seconds

Note that the distance is output with three digits to the right of the decimal point while seconds is output with two digits to the right of the decimal point. Both are in fixed format.

Assume that the number of seconds is 10.5, the output would be:

The object traveled 540.225 meters in 10.50 seconds

*int main()*

The main function will be the driver for your program.

First you need a loop that will process input values until you get an input value that is equal to 0. You will get this input value by calling the readSeconds function.

If the value is greater than zero the main function needs to call the calculateDistance and displayResults functions.

If the value is zero the loop should end and your program should then end.

Note that all of the non-main functions are called from main.

For the following sample run assume the input is as follows:

-12.5

-3.5

10.5

4.2

0

Your program should output the following:

Enter the time (in seconds)

The time must be zero or more

Enter the time (in seconds)

The time must be zero or more

Enter the time (in seconds)

The object traveled 540.225 meters in 10.50 seconds

Enter the time (in seconds)

The object traveled 86.436 meters in 4.20 seconds

Enter the time (in seconds)

Note that a zero is used to terminate the loop, but a value of zero is actually a valid value for seconds. We just aren't using that value in this program.

Failure to follow the requirements for lab lessons can result in deductions to your points, even if you pass the validation tests. Logic errors, where you are not actually implementing the correct behavior, can result in reductions even if the test cases happen to return valid answers. This will be true for this and all future lab lessons.

*Expected output*

There are seven tests. The first three tests will run your program with input and check your output to make sure it matches what is expected. The next three tests are unit tests. The unit tests will directly call the calculateDistance function. The compilation of the unit test could fail if your calculateDistance function does not have the required signature. The last test will only have invalid data to make sure your program works properly in that environment.

You will get yellow highlighted text when you run the tests if your output is not what is expected. This can be because you are not getting the correct result. It could also be because your formatting does not match what is required. The checking that zyBooks does is very exacting and you must match it exactly. More information about what the yellow highlighting means can be found in course "How to use zyBooks" - especially section "1.4 zyLab basics".

Finally, do not include a system("pause"); statement in your program. This will cause your verification steps to fail.

*Note:* that the system("pause"); command runs the pause command on the computer where the program is running. The pause command is a Windows command. Your program will be run on a server in the cloud. The cloud server may be running a different operating system (such as Linux).

*Error message "Could not find main function"*

Now that we are using functions some of the tests are unit tests. In the unit tests the zyBooks environment will call one or more of your functions directly.

To do this it has to find your main function.

Right now zyBooks has a problem with this when your int main() statement has a comment on it.

For example:

If your main looks as follows:

int main() // main function

You will get an error message:

Could not find main function

You need to change your code to:

// main function

int main()

If you do not make this change you will continue to fail the unit tests.