*C++ requirements*

* You are not allowed to use any global variables. Use of global variables will result in a loss of 25 points.
* Your program must have at least 4 functions. You will need your main function. This will be the driver of your program. You must also have the calculateFutureValue. In addition you need a read function and a display function.
* All of your functions, except main need to have function prototypes.
* The calculateFutureValue function must have the following signature:

double calculateFutureValue(double presentValue, double interestRate, int months)

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

*General overview*

In lab lesson 8 part 2 you calculated the present value. In this lab we are going to be calculating the future value. You will need to read in the present value, the monthly interest rate and the number of months for the investment. The formula is going to compute compounded interest (by month). There are a number of required functions that you will be writing, so do not start programming before you have read all of the instructions.

*You must use function prototypes for all of the functions (except main).*

Here is the formula:

*F = P \* pow((1 + i),t)*

Where *F* is the future value, *P* is the present value, *i* is the monthly interest rate and *t* is the number of months.

Your program will need to make sure of better variable names that those used above.

The input for the program will be read in from a file.

You will read in the file name from cin.

You will then read in the present value, monthly interest rate, and the number of months from the input file.

There may be multiple sets of these values. You need to process them all.

The program should end when you reach the end of file on the input file.

The output from the program will be written to two different places.

Any error messages will be written to cout.

You will also be writing out to a file called *output.xls*.

The first thing you will write out are four "headings" separated by tabs (using the \t escape sequence).

Here is an output statement (this example is using cout). Your program will instead write to the output file.

cout << "Future Value\tPresent Value\tMonthly Interest\tMonths" << endl;

This outputs the four headings separated by tabs. If you were to try and open this file with Excel it would give you a warning about the file format being incorrect, but would then determine that it is a tab file and would open it up properly. You could then save it as a read Excel file. This is one way you can get output from your programs into Excel.

The numbers read in must be positive values greater than 0. If they are not you need to output the three read in values to cout and display an error message saying the values are not all greater than 0. Note that all double values should be output in fixed format with two digits to the right of the decimal point. This is true for output to cout and for output to the *output.xls* file.

Assuming the following input in the input file:

-10000 1.1 48

The following would be written to cout:

-10000.00 1.10 48

One or more of the above values are not greater than zero

Your program must have at least four functions (including main).

*Function to read data from the file*

One function will read in the input values from the input file. Each call will read in the present value, monthly interest rate, and number of months.

Note that the monthly interest rate will be a number such as 10 or 12.5. These are to be read in as percentages (10% and 12.5%). You will need to divide these values by 100 to convert them into the values needed in the function (.1 and .125 for the above values). You need to do this conversion before you call the calculateFutureValue function (see below).

The function will return an unsigned int value that indicates if the call worked or not. A return value of 1 indicates that the values were read properly, a return value of 2 indicates that the input data was invalid, and a return value of 0 indicates that we have reached the end of file. The calling function (main) will use this return value to indicate how to process any data read in by the function. Note that we have to read three values from a file. The input file has already been opened. That means you will have to pass the input file as a parameter to the read function. You *MUST* pass the input file by reference (a reference to the ifstream object you created for your input file). Since we are reading in three values you will also have to pass these variables to the function by reference as well. That way the function can update the variables passed to it with the values read in from the file. If the read from the file fails due to end of file you will return a value of 0. If the input data is valid (all of the values are greater than 0) you will return back a value of 1. If any of the input values are invalid (0 or less) you need to return back a value of 2.

*Function to calculate the future value*

This function must have the following signature:

double calculateFutureValue(double presentValue, double interestRate, int months)

This function will be called by unit tests, so you have to have this signature.

The function needs to calculate the future value and return it to the calling function.

The formula for calculating the future value is show at the start of this lab lesson.

*Function to write the values to the output file*

You will need to pass the output file (by reference) to this function. You will also need to pass the future value, present value, interest rate (the value before conversion), and number of months to the function (these do not have to be passed by reference).

This function will write the four values to the output file. The values must be separated by tabs (using the escape sequence for the tab character). The numbers must be written to the file in fixed format with two digits to the right of the decimal point.

There is no return value from the function

*Finally the main function*

The logic for your main will be as follows:

1. Read in the file name from the console
2. Open the input file
3. If the input file did not open write out an error message and stop processing
4. Open the output file
5. If the open failed close the input file, display an error message and stop processing
6. Write the headings to the output file.
7. Loop while we do not have an end of file condition
   1. Call an input function that reads in the present value, interest rate and number of months
   2. If the input is valid calculate the future value and display the results.
   3. If the input is not valid display the data and an error message
8. End of loop
9. Close the input and output files

See *Sample run # 3* below for the syntax of the error message that needs to be written to cout when either of the input or output files cannot be opened.

*Sample run # 1 (valid input):*

Assume the following is read in from cin

input.txt

The *input.txt* file contains:

100 .99 36

The contents of *output.xls* after the run:

Future Value Present Value Monthly Interest Months

142.57 100.00 0.99 36

*Sample run # 2 (invalid input)*

The following is read in from cin

input.txt

The contents of *input.txt* are:

-10000 1.1 48

10000 -1.0 12

10000 1.1 0

0 0 0

The following would be written to cout:

-10000.00 1.10 48

One or more of the above values are not greater than zero

10000.00 -1.00 12

One or more of the above values are not greater than zero

10000.00 1.10 0

One or more of the above values are not greater than zero

0.00 0.00 0

One or more of the above values are not greater than zero

*Sample run # 3 (invalid input file name)*

Assume the following is read in from cin

more.txt

The *more.txt* file does not exist

The contents of cout after the run:

File "more.txt" could not be opened

*Expected output*

There are nine tests. Tests 3 through 5 are unit tests of function calulateFutureValue. The rest of the tests have various input values. For these tests you have to match the expected output.

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".

*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.

*If you see extra output*

If you are seeing extra output you are probably not properly handling the end of file condition.

The easiest way to handle this is to do the following:

if (inputFile >> var1 >> var2 >> var3)

{

// the read worked. Do your processing here

}

else

{

// you have reached end of file