ECE 1305, Ian Scott-Fleming

Lab assignment 1

Objective:

Before we get into the meat of the course, we need to be confident that you know how to create Visual Studio projects, know where Visual Studio puts the source code files you create, and know how to find the executables (.exe files) that it creates. This lab is designed to make sure you know all that, and get you started on logical and arithmetic expressions.

Grading on this will be "untuitive" – does your submission demonstrate that you've got an 'A's grasp of the material? mostly got it (B); seem a little shaky (C). Students who format the code sloppily will probably end up with a minus or lose a grade…students who format the code nicely, and add comments with their name, section, etc., and a brief description, will probably end up with a plus…

Note for the curve-busters in the class: if parts 1 & 2 seem too trivial for you (you know Windows & have worked with VS already, e.g.), see the "Babylonian Challenge" below!

What to submit:

(See below, but please read through the assignment first) If you work with others in the class, each of you must do your own submission, and you **must** put, in the header comments, the names of people you worked with on this assignment.

Remember to put your name, the course, and the assignment in comments at the top of each program, and to format and indent your code as demonstrated in class.

Please use the specified filenames for your source code. This makes it much easier for the grader to streamline the grading process. It allows her to create one VS project and reuse it without having to modify it for each student. This gives her more time to spend grading and giving you useful feedback.

Code Examples

Here are some example programs for you to look at. These may help you solving your assignments.

Inventory application – C++ arithmetic

A college bookstore receives cartons of textbooks. In a shipment, each carton contains the same number of textbooks. The inventory manager wants to use a computer to calculate the total number of textbooks arriving at the bookstore for each shipment, from the number of cartons and the number of textbooks in each carton. The inventory manager will enter the number of cartons received and the fixed number of textbooks in each carton for each shipment; the application then will calculate the total number of textbooks in a shipment

Example solution for inventory

C++ logic

To help a bouncer at an entertainment place, you are supposed to write a program that makes a decision based on the age of the client: he/she can go in and drink (21+), go in but not drink (18-20), or is too young to go in (under 18). Your program should ask for the age and make a decision.

Example solution for bouncer

Part 1

Create a Visual Studio project for each of the above programs, compile and run them. You can cut and paste the code…no need to type it. The goal here is to be comfortable creating VS projects, compiling them, and running them. Use the source-code filename (without extension) as the project name in each case.

Part 2

Use the Windows file browser (NOT Visual Studio) to locate the project(s) you just created on the file system. The goal is to be sure you know where on the file system your projects are stored.

You should turn off Windows' feature "Hide Extensions for known File Types" in the browser, so you can tell which files are the CPP files, which are the executables (.exe's), the object files (.obj), etc.. Otherwise, you'll likely end up submitting the wrong file (like hello.txt or hello.log instead of hello.cpp).

in Win7, this is under

Organize | Folder and Search Options | View

in the file browser ("Explore") and uncheck the box for "Hide Extensions for known File Types". It's different on Windows 8 file browser -- "View" pane, "File Name Extensions".

Using a separate text editor (such as Windows' notepad app), document where Visual Studio is saving your files, list the folders and files created by Visual Studio, and how to navigate to your executables and run them. Document how to find the source code file (e.g. inventory.cpp) and the executable file (inventory.exe). You will submit this text file for part 2. You do not need to write much for this...just enough to help yourself (and others) navigate both Windows and VS's file hierarchy. VS creates such a monstrous set of files for even the simplest of projects that it will be very helpful to have this properly documented for future students, so the "best" documented explanation will get turned into a help file for posting on Blackboard.

Be aware that VS puts your executable in a folder called "Debug", but rather obnoxiously creates 2 or more debug folders within each project/solution, so you may need to poke around a bit to find it.

Part 3: Little siblings' dividing program

Your younger brother and sister are currently learning division in school. Since they keep asking you for help on their homework, you have decided to write a computer program that can help them

to check their homework.
You younger brother is currently in grade school. He needs to divide whole numbers and know the result and the remainder of the division.
Your sister is in the next grade up. She also needs to divide whole numbers, but her teacher wants her to report real-number output with decimal places as needed.
It is now your job to write one program that will read in two numbers and give the output for both of your siblings.
(**goal: whole number division, remainder, and decimal division, conversion between integer and float data types**)

Following is an example of the program design process you should go through

Program design:

- input variables: integer variables for dividend and divisor
- result/output variables:
  - for brother: integer variables for quotient and remainder
  - for sister: floating point (use doubles) for quotient
    - Q: what do you need to do to force C++ to do floating point arithmetic if divisor and dividend are integers?
- Program flow:

1. prompt for input values
2. calculate and output brother's math problem (dividend, divisor, quotient, remainder)
3. calculate and output sister's math problem (dividend, divisor, quotient

Additional requirements:

- Name your program **division.cpp**
- Please include your names in a comment.
- Please indent your program properly.

Once you get that working, add the following:

- *Divisions by zero are a bad thing. Add an extra if statement to catch the error if they enter a zero as the divisor.*
- Hint: = is not the same as = = . See "Comparison Operators" on p. 48, and especially "Pitfall: Using = instead of…" on p. 59

Part 4:

Write a program to calculate some trajectory information for a ball thrown into the air at a given initial velocity. Specifically, calculate the time needed to reach maximum height and the maximum height reached. Assume:

1. *V_0 is much less than escape velocity, so acceleration, gravity = constant = -9.8 m/sec/sec*
   note: gravity has a negative acceleration!
2. *wind resistence is negligible, so basic newtonian physics equations apply.*
3. *y_0 = ground level*

4. *recall from freshman physics book (Table 3.1 in my copy of Halliday & Resnick):*
    a. $v_t = V_0 + a \cdot t$
    b. $y = y_0 + ½ (V_0 + v_t) \cdot t$
       or
       $y = y_0 + v_0 \cdot t + ½ a \cdot t^2$

1. Determine the variables (and data types) needed for inputs, calculations & outputs
2. Determine the logic/program-flow/steps needed to solve the problem
    a. (write it in 'pseudo-code', then use the pseudocode description as comments)
3. You'll want to work out a few sample problems on paper first, to make sure you know what the calculations should be and what the expected answers are for a given velocity and angle.
    a. For example: work out launching straight up @ 1 times the speed of gravity (9.8 m/s^2), 1.5 gravities (1.5 * 9.8 m/s^2) and at 2 gravities (2 * 9.8 m/s^2).
4. Q: what should your program do if the user launches the ball downwards (i.e., negative initial velocity) instead of upwards? I.e., if you've got it doing what it should do (calculating height and time for ball thrown upwards), make sure it doesn't do what it shouldn't do: give an incorrect answer if the inputs aren't valid.

Name your program **gravity.cpp** .

Part 5: Progressive Tax calculator

A country has a progressive tax. Depending on the income, the people have to pay more or less taxes.
Their first $ 5,000 are tax free
For everything over $ 5,000 up to $ 10,000 they have to pay 10%
For everything over $ 10,000 up to $ 20,000 they have to pay 15%
For everything over $ 20,000 up to $ 30,000 they have to pay 20%
For everything over $ 30,000 they have to pay 30%
As you can probably guess, your job is to write a tax calculation program that will calculate they amount of taxes for a given income.

Part 5a

Do not start programming yet!. Sit down and think first. Determine the data (variables and constants) you need, what you need to get from the user, and the steps needed to solve the problem. Write the pseudo-code for the program on the computer (you can create a .txt file in VS). **Work out the answer for several values, so you have some numbers to test your code with. Then trace through your algorithm by hand, and make sure that it produces the same answers.**

The pseudocode you write will make good comments to document your code

Part 5b

Once you have the pseudo-code, you may work on the actual program

Additional requirements:

- Name your program **progressive_taxes.cpp**
- Please include your name, course and a brief description in the header comments.
- Please indent your program properly.
- Since you are outputting a monetary value, please format the result properly (2 digits after the decimal point).

Babylonian Challenge for those who find some of the above too simple

If you know your way around Windows & VS already, and have integer & floating point arithmetic down pat, work problem 9 of chapter 2 (p. 96). The problem is implementing the babylonian algorithm for finding the square root of a positive number.

What to submit:

You need to upload 4 (or 5) things to Blackboard. Upload the files as attachments; if you need to explain anything, add it in the text field on the assignment page. ***Please use the specified filenames for your source code.***

(You should be able to upload multiple files. if you need to submit a revised version, a resubmission should overwrite an earlier file of the same name. If it won't allow you to overwrite an earlier submission, please email me. It means I didn't set up the assignment properly in BB)

1. Parts 1 & 2: Your **Visual Studio documentation notes**: a text file, "Visual_Studio_notes.txt" (or .docx if you prefer) with your notes on how to find your VS project files with browser)
2. Part 3: Submit your division.cpp program
3. Part 4: Submit your gravity.cpp program
4. Part 5: Submit your progressive_taxes.cpp program
5. Babylonian Challenge, if you did it.
