Assignment #2
Conditional & Repetitive Statements
Due: Monday, 04/22/13, 11:59pm

(75 pts) Problem Statement: Write three C++ programs to practice conditional and repetitive logic. In addition, you will read input from the user.

1. Write a C++ program, grade_calc.cpp, that calculates the total grade for N classroom exercises as a percentage. The user should input the value for N followed by each of the N scores and totals. Calculate the overall percentage (sum of the total points earned divided by the total points possible), and output it as a percentage. Sample input and output are shown as follows:

   How many exercises to input? 3

   Score received for exercise 1: 10
   Total points possible for exercise 1: 10
   Score received for exercise 2: 7
   Total points possible for exercise 2: 12
   Score received for exercise 3: 5
   Total points possible for exercise 3: 8

   Your total is 22 out of 30, or 73.33%.

   **From Savitch, Programming Projects #7, p. 95

2. Write a C++ program, buoyancy.cpp, to read the weight, (in pounds) and radius, r, (in feet) of a sphere as input from the user using the C++ cin input stream, and output whether the sphere will sink or float in the water. Use $\gamma = 62.4 \text{ lb/ft}^3$ as the specific weight of water, and the volume, $V$, of the sphere is $(4/3)\pi r^3$. The buoyant force can be computed by

   $$F_b = V \times \gamma$$

   where $F_b$ is the buoyant force, $V$ is the volume of the submerged object, and $\gamma$ is the specific weight of the fluid. If $F_b$ is greater than or equal to the weight of the object enter by the user, then your program will output, “This sphere will float”, otherwise your program will output, “This sphere will sink.”

   Then, ask the user if he/she wants to find out whether another sphere will sink or float based on a different weight and/or radius. If the user wants to calculate buoyancy again, then repeat the process above, asking the user for the weight and radius of the sphere, calculating the buoyant force, outputting a message, etc.

   **Revised from Savitch, Programming Projects #6, p. 95
3. Edit your rand_numbers.cpp program from Lab #3, so that
   • it chooses a random int that is always less than 50. I think the mod operator %
     would be very useful here. Anything mod 50 is < 50.

   • it gives the user 5 tries to guess the mystery int. Use the cin stream to read in the
     user’s guesses, and don’t forget to print a prompt for the user, i.e.
     
     ```
     cout << “Guess my number: ”;
     cin >> user_guess;
     ```

   • it tells the user if they are too high, too low, or exactly right. If they don’t get it
     right in 5 guesses the program just ends. The program also ends if they guess
     correctly. (Just return 0;) Use if statements with else clauses to determine low,
     high, just right. INDENT YOUR CODE INSIDE THE IF STATEMENTS!!!!

Here is an example run of my program:
Guess my number. 25
25 is too low
Guess my number. 34
Perfect guess!

Here is another running of my program:
Guess my number. 25
25 is too low
Guess my number. 35
35 is too low
Guess my number. 42
42 is too high
Guess my number. 39
39 is too high
Guess my number. 37
Perfect guess!

• Now, modify your rand_numbers.cpp so that the computer asks YOU to think of a
  random number between 0 – 50. And the computer gets 5 guesses. Write your
  program so that the computer always guesses ½ way inside the possible interval.

Here is my output when my secret number is 40 (user input is in bold):
Enter a positive number that is less than 50: 40
My first guess is 25.
Enter 1 for too low, 2 for perfect, 3 for too high. 1
My 2nd guess is 37.
Enter 1 for too low, 2 for perfect, 3 for too high. 1
My 3rd guess is 43.
Enter 1 for too low, 2 for perfect, 3 for too high. 3
My 4th guess is 40.
Enter 1 for too low, 2 for perfect, 3 for too high. 2
Yeah!
(10 pts) In your implementations, make sure that you include a program header/description in your program, in addition to proper indentation/spacing and other comments! Read the class style guideline for more information: http://classes.engr.oregonstate.edu/eecs/spring2013/cs161-001/161_style_guideline.pdf

You are graded on having a header, proper comments, and readable code with indentation and vertical spacing that is CONSISTENT throughout your program. DO NOT align your entire program on the left side. This will cause you to automatically lose the full 10 points. In addition, do not forget your program header!!!

(15 pts) You are required to turn in a written document (as a pdf) addressing Polya’s steps to solving a problem for both problems above, with step 3 being the C++ programs you write to carry out/implement your plans. With this said, your written document must include these three sections:

**Understanding the Problem**
In your own words, explain what YOU think the problem is asking you to do. In this section, document your uncertainties about the problem and anything else that you feel was unclear or vague. This is to ensure that YOUR understanding matches MY understanding of the problem.

**Devising a Plan/Design**
At a minimum, provide an algorithm/pseudo code you designed to help solve the problem. In addition, include pictures/flow charts you used to help you devise your plan, as well as any other design decisions you made such as how to manage your time, how to decompose the problem, where to start first, etc. You can scan any handwritten work and attach it to the document as needed.

**Looking Back/Self-Reflection**
Report any checking/self-reflection you did while solving the problem. For instance, how did you make sense of the output from the implementation? This includes things such as using a calculator to make sure the output is correct, testing to make sure your code executes correctly and behaves the way you expect under specific circumstances, using external sources of information such as the internet to make sense of the results, etc. Also, include a statement about what you learned from the assignment.

Electronically submit your three C++, buoyancy.cpp, grade_calc.cpp, and rand_numbers.cpp, program files and your design document, .pdf, by the assignment due date, using TEACH.