CSC 215
Project 1
(Courtesy of Megan Olsen, Ph.D. of Loyola University, with modifications to use C++.)

Due: Sunday, 03/02/15 at 11:59 PM

Problem: The FIS Nordic World Ski Championships in Falun is coming! What type of speed does a ski jumper need to achieve on the ramp to make a good distance on their jump? Let's make a program to calculate it and determine how many points they'd receive if they went that distance.

Purpose: This project gives you practice with:

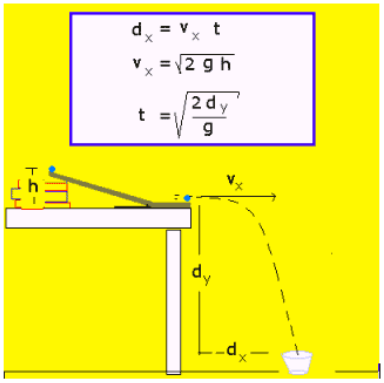
- Using cin for input
- Using the cmath library (Dale and Weems pg. 109)
- Comparing strings (Dale and Weems pg. 191)
- Using decision making in your code (if/else-if/else)
- Drawing a flowchart & finding control paths
- Working in pairs

Details:
Given the type of ski jump (normal vs. large) and the jumper's speed at the end of the ramp, predict how far they will jump using the simplification shown below. After determining their distance, you can also calculate how many points they would get on that hill. For a normal hill, 90m is par (i.e. the average expected result) and on a large hill 120m is par. If they are below par, they lose points for each meter; if they are above par, they gain the same amount of points per meter.

The algorithm below basically follows these steps: 1. Declare variables, 2. Ask user for necessary information, 3. Set variables based on user input, 4. Calculate distance and points, 5. Output results.

Write your code to follow this algorithm:

0. Declare height (short), points_per_meter (double), par (int), velocity (double), hill (String)
1. Ask the user for the jumper's take-off velocity, as a real number in m/s
2. Ask the user if the jump is on the normal hill or the large hill (enter "normal" or "large")
3. If the jump is on the normal hill, then
    a. set height to 46
    b. set points_per_meter to 2
    c. set par to 90
4. Otherwise, if the jump is on the large hill, then
    a. set the height to 70
    b. set points_per_meter to 1.8
    c. set par to 120
5. Otherwise:
    a. output "error, not a type of hill"
    b. end program using System.exit(1);
6. Calculate the time in the air as $\sqrt{(2.0 * height)/9.8}$
7. Calculate the distance as velocity * time
8. Calculate the points earned as 60.0 + (distance - par)*points_per_meter
9. Output the predicted time in the air and distance traveled
10. If the points earned is greater than 60, output "Great job going so far!"
11. Otherwise, output "Better luck next time!"



Steps:

1. Create a new cpp file and name your file SkiJumper.cpp .
2. Write comments in your file to make it clear what you plan for it to do.
3. Draw a flowchart of the above algorithm.
4. Label the control paths in your flowchart
5. Write your C++ code following the above algorithm. **Use cin for all input and cout for all output.**
6. Compile your code. (Run g++ on your file, check for errors, if there are errors fix errors and repeat, else, test your Program.)
7. Create test cases based on your control paths. Use the test cases to make sure your program works correctly, and fix it if it doesn't. Don't just assume you did it right, there are many things that could have gone wrong.
8. Update your comments, and make sure to include an updated version of the header comments. Many lines should change! *Please include first and last name of all programmers.*

Make your output easy to understand.

Submit:

To Canvas a jar file with the following:

- 2 short (250 words or less) individual narratives about what you did for the project, what it was like working with your partner, and what you had the most trouble with. (1 per person; .doc or .pdf)
- A pdf, word doc, or Image of your flowchart. (If you draw the flowchart by hand, you should scan it or take a clear photo of it to digitize it.)
- A jar of your project1 directory that includes:
    i. Your cpp file
    ii. Your compiled binary file
    iii. A Readme.txt file describing how to compile and run your program.