CS 10, Lab 3

Loops

Start a new document in Word or a similar program. Include the names of both members of your team and any additional information or code as instructed below. At the end of the lab, please email your lab report to cs10labtues@gmail.com or cs10labthurs@gmail.com **with the subject line "CS10 Lab3 lastname lastname", with the last names of both members of your group.**

For reference, I've posted a C++ program with examples of different kinds of loops in it:

http://math.scu.edu/~linnell/cs10resources/loops.txt

Part A. An application from physics: Projectile trajectories

Given the variables are defined as follows:

- *g*: the gravitational acceleration—usually taken to be 9.81 m/s^2 near the Earth's surface
- θ: the angle at which the projectile is launched
- *v*: the speed at which the projectile is launched (in m/s)
- y_0: the initial height of the projectile (in m)
- The height *y* (in meters) of the projectile at horizontal distance *x* (in meters) is given by

$$y = y_0 + x\tan\theta - \frac{gx^2}{2(v\cos\theta)^2}$$

1. Write pseudocode that generates the x, y coordinates for a projectile at all values of x from 0 to 9. Your pseudocode should take all necessary variables as input from the user. Use a **`while`** loop. Check your answer with one of the lab teachers before moving on, and get your checklist signed.
2. Translate your pseudocode from 1 into C++ code. Be sure to format your x, y pairs as: (0, 10) (for example) with one pair per line. Copy-paste your pairs here (being sure to delete any extra text) and graph the points: http://www.shodor.org/interactivate/activities/SimplePlot/
   Copy-paste your code into your lab report, and include a screenshot of your plotted points. Run your code and plot the points to be sure it's right before moving on.
3. Change your pseudocode from 1 to use a **`for`** loop. You don't have to re-copy it if you don't want to, feel free to cross stuff out. Check your answer with one of the lab teachers before moving on, and get your checklist signed.
4. Translate your pseudocode from 3 into C++ code. Copy-paste your code into your lab report. Run your code and plot the points to be sure it's right before moving on. Compare your code from 4 and 2. Which do you like better? Write your answer in your lab report.

5. Change your pseudocode from 1 so that it still increments x by one every time, but it continues until the y value falls below the x axis. Why would this be a reasonable thing to do? (Hint: in your first attempt, you will probably print out one such (x, y) pair. How can you change your code so that it prints out NO x, y pairs with the y value below the x axis?)Check your answer with one of the lab teachers before moving on, and get your lab report signed.
6. Translate your pseudocode from 5 into C++ code. Copy-paste your code into your lab report. Run your code with several outputs to be sure it's right before moving on.
7. Change your pseudocode from 5 to use a **`for`** loop. Check your answer with one of the lab teachers before moving on, and get your lab report signed.
8. Translate your pseudocode from 7 into C++ code. Copy-paste your code into your lab report. Run your code with several outputs to be sure it's right before moving on. Get your checklist signed before moving on. Compare your code from 6 and 6. Which do you like better? Write your answer in your lab report.

----------MINIMUM STOPPING POINT----------

Part B. Loops for error handling.

You can find a code snippet that solves last week's lab here:

http://math.scu.edu/~linnell/cs10resources/genotype.txt

1. Treat this code as a block called B (DO NOT CHANGE block B), and write pseudocode using this block so that if the user enters an incorrect input, they are prompted to enter new input as many times as it takes until they enter proper input. If you're stuck, think about what the output of this code should look like. Check your answer with one of the lab teachers before moving on, and get your lab report signed.
2. Copy-paste the code below into your editor, and translate your pseudocode from 1 into C++ code. Copy-paste your code into your lab report. Run your code with several outputs to be sure it's right before moving on.

Part C. An application from math: Prime numbers.

A prime number is a number whose only divisors are 1 and itself. So we know that 7 is prime because none of 2, 3, 4, 5, or 6 divide evenly into 7. Write a program that prompts the user for a number, and then tests whether that number is prime. Your program should print out the result. You must use a for loop.

1. Write pseudocode to do this. If you're feeling stuck, try to think of how you would check if a number is prime using paper and pencil, then try to think of how you would automate that process. Check your answer with one of the lab teachers before moving on, and get your lab report signed.
2. Translate your pseudocode from 1 into C++ code. Copy-paste your code into your lab report. Run your code with several outputs to be sure it's right before moving on.