CS 010 - Introduction to Computer Science I

Lab 6: More Loops

**Suggested Pre-Lab Work (you should have completed at least some of these items)**

Zyante Chapter 4.1-4.5 and corresponding Codelab exercises

Video tutorials: [Module 6 playlist](http://www.youtube.com/playlist?list=PLTTJbxrH72A1shOqKIKH3S4ainxQRptby)

**Collaboration policy**

Collaboration on these lab exercises is strongly ENCOURAGED.  
Read the full policy at: [Full Collaboration Policy](https://docs.google.com/document/d/1WyzL3qvKLrC1UCRf178b_wYWQmEZlhDObFNFb79U63I/edit?usp=sharing)

Lab Objectives

* understand and use correctly the different forms of loop (while, do-while, for)
* be able to use comments and indentation to visually separate logic code blocks

**Initial Setup**

See the [Lab submission framework](https://docs.google.com/document/d/15vuxeMVluLJC54Sr_nW29g1xQELLWWy9a8y0pDOW64E/edit?usp=sharing)

For all exercises:

As always, all your code must follow the class [style guide](https://docs.google.com/document/d/1DDvMTkQtrsRoQL7E7F4UdJshO-TN06eMy9E_Eq6lxUs/edit?usp=sharing) - correct indentation, spaces around all operators, meaningful variable names, etc.

In particular, now that your programs are starting to get a little more complex, you must pay attention to clearly commenting your code: each “conceptual block” of code should have a brief comment describing its purpose.

**Exercise 0: Survey**

This week's survey is an questionaire about the tools we use in CS 10, we appreciate your feedback. We utilize the feedback to better use the tools within the course, and feedback may be passed on anonymously to the various tool developers, such as Zyante or C9.

<http://goo.gl/Io3QmK>

**Exercise 1: for loops**

Write three for loops to produce the numeric sequences below.  
Your **output should be as displayed,** with numbers separated by a comma and space. There should also be a single blank line between each sequence.

*Hint: To generate all three sequences, you should only have to change the for loop parameters (i.e. the initialization, the conditional, and the update). Once this simple change occurs in the for loop header, a line (at most two) inside the body may need to be changed to accommodate proper output formatting with commas.*

***Before coding, write down in words the algorithm behind each sequence.***

User input has been **bolded and underlined** for emphasis:

Which exercise? **1**  
  
Enter beginning and ending numbers, separated by space: **0 20**  
  
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20  
  
Enter beginning and ending numbers, separated by space: **1 35**  
  
1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35  
  
Enter the beginning and largest positive numbers, separated by space: **2 128**  
  
2, -4, 8, -16, 32, -64, 128, -256

**----------- Another sample run of the same program -----------------**

Which exercise? **1**  
  
Enter beginning and ending numbers, separated by space: **1 5**  
  
1, 2, 3, 4, 5  
  
Enter beginning and ending numbers, separated by space: **2 13**  
  
2, 4, 6, 8, 10, 12  
  
Enter the beginning and largest positive numbers, separated by space: **1 256**  
  
1, -2, 4, -8, 16, -32, 64, -128, 256, -512

As you can see, the for-loop structure is particularly effective at handling iterations where we know in advance (or can easily calculate) where to *start* counting, and where to *stop*; “counter-controlled” loops are a typical use.

**Exercise 2: Sentinel-controlled loops**

Write a program that will read in from user input the scores of an exam (possible scores: 0 to 100). We do not know in advance the number of students, so the user will have to type in a *sentinel* value when they have finished entering scores - i.e a value that could not be mistaken for an actual score. In this case, it’s pretty clear that a *negative number* would be a good choice for our sentinel.   
**We will utilize -1 as our sentinel value.**

Instead of outputting anything within the loop, you will keep track of how many A's, B's, C's, D's and F's there are. Output the letter grades and the count for each *only after you exit the loop* (i.e. after the user has entered the sentinel).

Use the scale: 90 - 100 => A, 80 - 89 => B, 70 - 79 => C, 60 - 69 => D and 0 - 59 => F.

*Hints:   
You will need a separate int variable to keep track of the number receiving each letter grade;   
increment that variable within a branch of an if / else if / else structure;  
make sure the SENTINEL isn't accidentally entered as a grade.*

Since we do not know how many iterations the loop will execute, the while or do-while loop is the best choice for this exercise.

User input has be **bolded and underlined** for emphasis:

Which exercise? **2**

Enter one or more grades, or -1 to stop:

**45**

**78**

**99**

**92**

**45**

**67**

**-1**

The grades breakdown is:

As: 2

Bs: 0

Cs: 1

Ds: 1

Fs: 2