CS 010 - Introduction to Computer Science I

Lab 7 - Functions

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

Zyante Chapter 5 sections 5.1 - 5.4, and corresponding Codelab exercises

Video tutorials: [Module 7 playlist](http://www.youtube.com/playlist?list=PLTTJbxrH72A2CVm0M7-jsAvKaKz5Rycg5)

### 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**

To gain experience with:

|  |  |  |
| --- | --- | --- |
| * random numbers | * invoking functions | * writing functions |

**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**

The survey link will be added. The survey will go live at 8am on Tuesday.

**Exercise 1: genRandInt Function**

As with all programs that utilize random numbers, you must seed the random number generator once and only once. If you are comparing to the results shown in example you should seed with srand(433) but if you are turning into R'Sub, you should use srand(time(0)).

Recall, to use the random number function, rand(), we must include the C Standard Library:

#include <cstdlib>

A note to those not working within C9. The random numbers generator by different systems (Windows, Linux, OS X) may be different as the implementation for generating pseudo random numbers is not exactly the same, so if you are developing outside C9 and are not getting precisely the displayed example, try either running on C9 or submitting to R'Sub.

Your task is to write a function. The function must fill the following requirements:

* name: genRandInt
* return type: integer
* parameters: two integers (order matters)
  + the smallest number in the range of random numbers
  + the largest number in the range of random numbers

Your exercise should:

* Invoke your function multiple times, storing the returned result.
* Use a single variable to store the result of your function invocation. Simply update the variable and output the result prior to the next invocation.
* Use the key phrase "Random number" to preface your output results, as shown below.

**Example Run** (inputs are **bold & underlined** to emphasize the difference between output and typed input)

|  |  |
| --- | --- |
| **Example 1**  $ g++ lab7.cpp  $ run a.out  Which exercise? **1**  Enter the smallest possible number: **23**  Enter the biggest possible number: **75**  Random number: 38  Random number: 34  Random number: 55 | **Example 2**  $ g++ lab7.cpp  $ run a.out  Which exercise? **1**  Enter the smallest possible number: **12**  Enter the biggest possible number: **23**  Random number: 16  Random number: 16  Random number: 19 |

**Exercise 2: Coin Flip**

Your goal is to write a program that simulates a coin toss. You **SHOULD** utilize the function you wrote in exercise 1 to generate the random integers required to simulate the coin toss.

As with all programs that utilize random numbers, you must seed the random number generator once and only once. If you are comparing to the results shown in example you should seed with srand(433) but if you are turning into R'Sub, you should use srand(time(0)).

A note to those not working within C9. The random numbers generator by different systems (Windows, Linux, OS X) may be different as the implementation for generating pseudo random numbers is not exactly the same, so if you are developing outside C9 and are not getting precisely the displayed example, try either running on C9 or submitting to R'Sub.

Inside a loop, continue tossing a coin until the user responds "no" to the question, as shown below. For our purposes we will only allow two possible values to be generated; heads will be the value 1 and tails will be the value 2.

Your exercise should:

* Invoke your function multiple times, storing the returned result.
* Use a single variable to store the result of your function invocation.
* Use the key phrase "Flip" to preface your coin toss result, as shown below.

**Example Run** (inputs are **bold & underlined** to emphasize the difference between output and typed input)

|  |  |
| --- | --- |
| **Example 1**  $ g++ lab7.cpp  $ run a.out  Which exercise? **2**  Flip: heads  Toss the coin again (yes/no)? **yes**  Flip: heads  Toss the coin again (yes/no)? **yes**  Flip: tails  Toss the coin again (yes/no)? **yes**  Flip: heads  Toss the coin again (yes/no)? **no** | **Example 2**  $ g++ lab7.cpp  $ run a.out  Which exercise? **2**  Flip: heads  Toss the coin again (yes/no)? **yes**  Flip: heads  Toss the coin again (yes/no)? **yes**  Flip: tails  Toss the coin again (yes/no)? **yes**  Flip: heads  Toss the coin again (yes/no)? **yes**  Flip: heads  Toss the coin again (yes/no)? **no** |