CSE 1321L: Programming and Problem Solving I Lab

Lab 5

Flow Control (Part 2)

What students will learn:

* Using WHILE loops
* Using FOR loops
* Using nested FOR loops

If there’s one thing computers are good at, it is repeating something over and over. The concept of repetition (which some call “iteration” and others “looping”) is not terribly difficult since we humans repeat things in our daily lives. Any decent programming language is going to support iteration and usually allows for two different kinds of “looping templates”. These templates are exactly what this lab is going to cover.

The two kinds of loops we’ll cover are the for and while loop. You want to memorize the templates for these.

while <condition>:

<body>

for <element> in <iterable>:

<body>

In a WHILE loop, <condition> can be anything that resolves to a Boolean value (True or False). This could mean Boolean variables, as well as comparison or logical expressions.

In a FOR loop, <iterable> can be anything that can be iterated over (we’ll see more about these at a later module. Right now, the only things that you can iterate over are strings and ranges). At each iteration (read: lap) of the loop, <element> will hold one element from the <iterable>. These elements are retrieves one by one, in the order that they appear in the <iterable>.

It’s important to know when to use them. Here’s an overall guideline to help you out:

1. Use a for loop when you want to repeat something a certain number of times. For example, if you want to repeat something 100 times, and a for loop is a good candidate for that. Or, if you wanted to count from 50 to 3000 in increments of 10, you could do that too.
2. Use a while loop is useful when you don’t know how many times something will repeat; the loop could “go on forever”. As an example, if you ask a user to enter a number between 1- 10 and they consistently enter 45, this could go on forever. Eventually (and hopefully), the user would enter a valid number.

**Lab5A: Largest of 10**

For this lab, please use a **for loop**. The goal of this exercise is for you to create a program that will ask the user to input 10 **positive integer** numbers, **one at a time**. While it does this the program should also keep track of the largest number it has seen so far. After it has run 10 times, it should display the largest number you input.

The user input is indicated in **bold.**

Sample output:

Please enter 10 numbers and this program will display the largest.

Please enter number 1: **50**

Please enter number 2: **51**

Please enter number 3: **10**

Please enter number 4: **1**

Please enter number 5: **99**

Please enter number 6: **1000**

Please enter number 7: **1010**

Please enter number 8: **42**

Please enter number 9: **89**

Please enter number 10: **1000**

The largest number was 1010

**Lab5B: The box**

Write a program that asks the user for a value. Based on the value given to the program by the user, use **nested for-loops** to draw a box that has the length and the width of the value specified by the user. Afterwards, use the same value and another pair of **nested for-loops** to draw a right-facing right-triangle. Finally, use a last pair of **nested for-loops** to draw a right-triangle facing the left. See the examples below for reference.

The user input is indicated in **bold.**

Sample Output #1:

Please enter a value for the size: **4**

This is the requested 4x4 box:

\*\*\*\*

\*\*\*\*

\*\*\*\*

\*\*\*\*

This is the requested right-facing 4x4 right-triangle:

\*

\*\*

\*\*\*

\*\*\*\*

This is the requested left-facing 4x4 right-triangle:

\*

\*\*

\*\*\*

\*\*\*\*

Sample Output #2:

Please enter a value for the size: **5**

This is the requested 5x5 box:

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

This is the requested right-facing 5x5 right-triangle:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

This is the requested left-facing 5x5 right-triangle:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

Sample Output #3:

Please enter a value for the size: **8**

This is the requested 8x8 box:

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

This is the requested right-facing 8x8 right-triangle:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

This is the left-facing requested 8x8 right-triangle:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

**Lab5C: Say “please”**

Write a program that prints the following message:

If you would like to stop this program, say “please”.

Afterwards, prompt the user for input. If the user enters “please”, terminate the program. Otherwise, print the message above and prompt the user to input again. The program should only terminate if the user enters “please”.

User input in **bold**.

Sample Output #1:

If you would like to stop this program, say "please": **please**Program complete

Sample Output #2:

If you would like to stop this program, say "please": **pLeAsE**  
If you would like to stop this program, say "please": **Please**  
If you would like to stop this program, say "please": **please**Program complete

Sample Output #2:

If you would like to stop this program, say "please": **no**  
If you would like to stop this program, say "please": **nah**  
If you would like to stop this program, say "please": **nuh-uh**  
If you would like to stop this program, say "please": **ok fine**  
If you would like to stop this program, say "please": **please**Program complete