CS-212: LAB 3 – Recursive Functions

Learning Objective:

- Writing recursive functions, including a quick sort.

This lab has three parts:

- The functions section will describe the functions you will need to write

Lab 3 Setup:

1. Open a new terminal window to SSH into your account on the [bingweb.binghamton.com] server. When logged in, make sure you are in your home directory by typing the following command if you are not already there. The "~" is just a shortcut symbol that means home directory:
   cd ~

2. Change your current directory to the "CS212" directory by typing:
   cd CS212

3. Confirm that you are in the "CS212" directory by typing:
   pwd

4. Create a directory for Lab3.
   mkdir Lab3

5. Move into the new directory
   cd Lab3

6. Create a new c file for you to write your lab.
   > <Last Name><FirstInitial>_Lab3.c
   > <Last Name><FirstInitial>_Lab3_Functions.c
   > <Last Name><FirstInitial>_Lab3.h

   Example: I would create FoosJ_Lab3.c, FoosJ_lab3_Functions.c and FoosJ_Lab3.h

   Note: <Last Name><First Initial>_Lab3.c will only have your main function.

INTRODUCTION:

In this lab you will write functions recursively. Refer to the sample program, playbook notes and Chapter 3 of the book for examples.

THE FUNCTIONS:

PART A:

FUNCTION 1: PrintHeader, CenterString, PrintDivider

You may copy these functions from a prior lab or a sample program

FUNCTION 2: OpenOutputFile, CloseOutputFile.

Files must be opened and closed in a function.

PART B – Writing recursive functions:

FUNCTION 3: (1st Recursive Function) Write the function StarPattern

Func 3-A: Write a function to print a header to either the screen or an output file.

HINT: If you write this function efficiently, you will be able to re-use it later in this lab. If you don't write it efficiently, then you will have to write multiple print header functions.

Func 3-B: Write a recursive function that will print the pattern below.

This function must be able to print to either an output file or the screen.

This function will take a parameter for the number of lines.

Call this function twice.

- First call goes to the output file and will print 12 lines
- Second call goes to the screen and will print 15 lines.

Sample Output for the first recursive function:

```
--------------------------------------------------------------------------------
                    Star Pattern - Printed to the output file
--------------------------------------------------------------------------------
*
**
***
****
*****
******
*******
********
*********
**********
***********
************
--------------------------------------------------------------------------------
```

FUNCTION 4: 2nd Recursive Function – Sum the elements of an array

Create two arrays of size 20 in the main function.

Use a function call the print the header for each set of output.

Func 4-A: Write a function to initialize an array to random numbers between 0-99. Call this function twice.

HINT: Seed the random number generator in the main, not the function.

HINT: If you write this function efficiently, you may re-use it in part C.

See Appendix A for how to code a random number generator.

Func 4-B: Write a function that will print the array with two integers per line.

HINT: Think modulus operator

HINT: If you write this function efficiently, you may re-use it in part C.

Func 4-C: Write a recursive function that will sum the elements of the array and return the sum from where the function is called. (HINT: Think single return).

Print the results with a label. The first array is to be printed to the output file, the second to the screen.

Sample Output:

```
--------------------------------------------------------------------------------
                    2nd Recursive Function: Sum the Array
--------------------------------------------------------------------------------
      ARRAY
     -------
    40    19
     3    45
    59    24
    73    61
    86    34
     5    92
    94     3
    40    26
    91    73
    75    97

Sum of Array One: 1040

--------------------------------------------------------------------------------
```

FUNCTION 5: 3rd Recursive Function – PrintDigit

Use a function call the print the header for each set of output.

Write a recursive function to print on digit at a time.

For the output that is printed to the screen, pass in integer 1680.

For the output that is printed to the output file, pass in integer 1385.

Sample Output:

```
--------------------------------------------------------------------------------
                    3rd Recursive Function: PrintDigits
--------------------------------------------------------------------------------
Digits printed recursively:
1680
--------------------------------------------------------------------------------
```

PART C – Writing a recursive sort:

Create two arrays of 50 random integers in the main. Integers should be between 0 and 999.

Call a function to initialize all elements of the array. You may reuse a function you wrote earlier.

All output will need a header, you may reuse a function you wrote earlier.

Print your array first array to the screen and your second array to the output file.

Print the array before and after your sort it.

Output two integers per line, so you may use a function you wrote earlier in the lab.

Sort the array using a quick sort algorithm.

Hints:

1. YouTube video: https://www.youtube.com/watch?v=Vtckgz38QHs
2. Textbook: Program 13.14

EXTRA CREDIT A: 10%

If your lab does not run with proper output, no extra credit may be earned.

If PrintHeader, PrintTable, and InitalizeArray are all written efficiently, so they can be reused, then you will earn a 10% extra credit points.

EXTRA CREDIT B: 10%

Write a recursive function that will reverse the digits of a number.

The recursive function must be a single return function

Call the recursive function with the two integer values (25689 and 345621)

You MUST solve this program WITHOUT converting the integer to any other data type

For each value, output the original value and the reversed value to the output file, clearly labeled

HINTS

Explanation of the reversal by a simple example:

- You will call the function with two parameters: the number to reverse, and 0 (this will eventually become the number in reverse).

- The process with the number comes into the function set to 123
- 123%10 =3
- 3 + 10*0 = 3
- 123/10 = 12
- 12%10 = 2
- 2 + 10*3 = 32
- 12/10 = 1
- 1%10 = 1
- 1 + 10 * 32 =321

RUNNING YOUR PROGRAM

Your program must run in gcc on the unix operating system with 0 errors and 0 warnings.

To run your program:

```
gcc *.c -Wall
```

This will create an output file named a.out

Valgrind must run with 0 errors and 0 warnings.

To run Valgrind:

```
valgrind --leak-check=full ./a.out
```

FILES TO SUBMIT TO BRIGHTSPACE

Submit your three code files, your output file, and copy your screen output into a text file.

<Last Name><FirstInitial>_Lab3.c
<Last Name><FirstInitial>_Lab3_Functions.c
<Last Name><FirstInitial>_Lab3.h
Lab3_Output.txt
Screen_Output.txt

APPENDIX A :

Code to generate a random number:

```c
// Include for the srand and rand function
#include <stdlib.h>

// Include for the time function
#include <time.h>

int main (void)
{

   int randomNumber;
   time_t t;

   // Intializes random number generator to the clock
   srand((unsigned int) time(&t));

   // Create a random number between 1 and 20
   randomNumber = rand() % 20 + 1;
```

Note: Don't forget to include the necessary header files.

Note: Your header file includes, and prototype should be in your header file.

ADDITIONAL PARTIAL CREDIT GRADING RUBRIC:

NOTE: The full rubric is on Brightspace under Misc

Infraction	Points Lost
Missing your header from the screen or output file (not both)	-5 points
One of your recursive functions is not recursive	-20 points
A second recursive function is not recursive **NOTE: More than 2 recursive functions are not recursive, the lab is incomplete and falls under Part 4 of the grading rubric 50%).**	-10 more points
Part C - Not printing your array before you sort it	-10 points