CS 161B: Programming and Problem Solving II

Assignment 7: Pointers are Fun!

Academic Integrity

You may NOT, under any circumstances, begin a programming assignment by looking for completed code on StackOverflow or Chegg or any such website, which you can claim as your own. Please check out the Student Code of Conduct at PCC.

The only way to learn to code is to do it yourself. The assignments will be built from examples during the lectures, so ask for clarification during class if something seems confusing. If you start with code from another source and just change the variable names or other content to make it look original, you will receive a zero on the assignment.

I may ask you to explain your assignment verbally. If you cannot satisfactorily explain what your code does, and answer questions about why you wrote it in a particular way, then you should also expect a zero.

When you learn C++/C, you must learn how to use pointers. As you know every variable is a memory location and every memory location has its address defined which can be accessed using the ampersand (&) operator which denotes an address in memory. Let's have some fun with pointers!

Purpose

In this assignment, you will create a single source code file that uses pointer variables and functions.

After completing this assignment you will be able to:

- Write code using pointers.
- Pass addresses of variables to functions.
- Be able to use the address of operator & and the dereferencing operator * appropriately.

Task

- □ There is a Unix Lab component to this assignment that must be completed as well. Instructions to this can be found in the Linux Lab 3 document. Watch this Lab 3 video and follow along.
- □ Open the Algorithmic Design Document, make a copy, and follow the steps to create your algorithm.
- □ You must express your algorithm as **pseudocode.**
- □ You will write a program that reads 2 integers from the user and calls 3 different functions to do the following:
    - □ `void swapArgs(int *, int *)` function that takes 2 pointers to integers and swaps the integers read in main().
    - □ `void divideArgs(int *, int *)` function that takes 2 pointers to integers and stores the integer quotient in the first pointer parameter and the remainder in the second pointer parameter.
    - □ `void powerArgs(int *, int *)` function that takes 2 pointers to integers, raises the first integer to the power of the second integer, and stores the result in the first integer. **You may NOT use the pow() function to do this, you must use a loop to calculate the result.** Recall, any number raised to the 0 power is 1. If the power is a negative number, do not calculate any result. See sample runs below.
- □ Create two integer variables in main(), prompt the user for two integers, and pass their addresses to the pointer parameters.
- □ If the user enters two zero's, print a message "No operations performed!" and end the program.
- □ Be careful to notice where pointers and variable addresses are used - the idea of this assignment is to practice writing code that uses pointers.
- □ Output the values of the two integers entered from the user, and their contents after the calls to the three functions. See sample runs below.

Criteria for Success

❏ Your program must use pointers.

❏ Test your program using the following sample runs, making sure you get the same output when using the given inputs (in **`blue`**):

```
Enter integer 1: 3
Enter integer 2: 17

Before call to swapArgs a: 3 b: 17
After call to swapArgs a: 17 b: 3
After call to divideArgs a: 5 b: 2
After call to powerArgs a: 25 b: 2

Goodbye!
```

```
Enter integer 1: 2
Enter integer 2: 10

Before call to swapArgs a: 2 b: 10
After call to swapArgs a: 10 b: 2
After call to divideArgs a: 5 b: 0
After call to powerArgs a: 1 b: 0

Goodbye!
```

```
Enter integer 1: -10
Enter integer 2: 3

Before call to swapArgs a: -10 b: 3
After call to swapArgs a: 3 b: -10
After call to divideArgs a: 0 b: 3
After call to powerArgs a: 0 b: 3

Goodbye!
```

```
Enter integer 1: 0
Enter integer 2: 0

No operations performed!
```

```
Enter integer 1: -2
Enter integer 2: -9

Before call to swapArgs a: -2 b: -9
```

```
After call to swapArgs a: -9 b: -2
After call to divideArgs a: 4 b: -1
After call to powerArgs a: 4 b: -1

Goodbye!
```

- ❑ Complete zyBooks section **15 CS 161B: Pointers** activities.
- ❑ You may use any IDE to write your code. If your Transfer University is PSU, check out the Special Note For PSU Transfer section in the Syllabus.
- ❑ Please bookmark the PCC Linux and Vim Manual - this will become a frequently used reference!
- ❑ Complete the Linux Lab 3 document. Watch this Lab 3 video and follow along.
- ❑ Complete all sections of your Algorithmic Design Document.
- ❑ Include **pseudocode** or a **flowchart** in part d of the design document.
- ❑ Please open and compare your work with the grading rubric before submitting.
- ❑ Remember to follow all style guidelines.
- ❑ Download your Algorithmic Design Document as a PDF (File -> Download -> PDF), rename it to `a07.pdf`, and upload it to the D2L assignment by **Wednesday**.
- ❑ Upload your `a07.cpp` C++ source file and `lab3.txt` to the D2L assignment by **Sunday**.
- ❑ Do your own work. Consult the syllabus for more information about academic integrity.

Additional Support

- ❑ Post a question for the instructor in the Ask Questions! area of the Course Lobby.