CSE2421 Lab 3 (3 points)
Pointer Basics

In this lab, you will conduct a series of small experiments to get familiar with pointers. As a result, your Makefile should generate a number of executables, each corresponding to an experiment. You also need to submit a document to answer some questions.
Notes:

1) For written questions, you are encouraged to try them on your programs, but you don't need to submit the code.
2) For output, you are free to choose a format you like, as long as the grader can understand your output.
3) Some questions require you to print the addresses of variables. You can print them as unsigned long values ("%lu" in printf)
4) We will use valgrind to check memory leak for every executable, so you'd better try it before you submit.
5) Some of the experiments require your program to take inputs from command line arguments. We have not learned this yet, but you can re-use code from Lab 2.
6) You can assume the inputs to the program are correct. In other words, you can ignore error checking.

Lab 3a: Swap numbers
The executable for this experiment should be lab3a.
Your program should take two integer numbers from the command line arguments and swap them. For example, "lab3a 1 2" should output "2 1".
You should refer to page 16-17 of 11_C_Pointers_Part_1 to write a swap function.
Written question: what happens if the arguments of swap are not pointers (i.e., page 16)?
Test cases: "lab3a 1 2"; "lab3a 100 30"; "lab3a -1 -2"

Lab 3b: Static array
The executable for this experiment should be lab3b.
Your program should create a static array with 20 elements, let array[i] = i, and print all elements in the array.
Written question: what happens if you create another static array called array2, and then do array = array2?
Test cases: no specific test case. "lab3b" should give the right output.

Lab 3c: Pointer arithmetic with static array
The executable for this experiment should be lab3c.
Your program should create a static array with 20 elements, use pointer arithmetic to assign i to array[i], and print all elements in the array.
Your program should print the address of each element in the following two ways: &array[i] and array + i. Are they always the same?
Written question: no
Test cases: no specific test case. "lab3c" should give the right output.

Lab 3d: Dynamic array
The executable for this experiment should be lab3d.
Your program should create a dynamic array whose size is from command line argument, let array[i] = i, and print all elements in the array.
Written question: what happens if you create another dynamic array called array2, and then do array = array2?
Test cases: "lab3d 5"; "lab3d 10"

Lab 3e: Pointer arithmetic with dynamic array
The executable for this experiment should be lab3e.
Your program should create a dynamic array whose size is from command line argument, use pointer arithmetic to assign i to array[i], and print all elements in the array.
Your program should print the address of each element in the following two ways: &array[i] and array + i. Are they always the same?
Written question: no
Test cases: "lab3e 5"; "lab3e 10"

Lab 3f: Static two-dimensional array
The executable for this experiment should be lab3f.
Your program should create a static two-dimensional array with 10 rows and 15 columns, let array[i][j] = i*j; print all elements
Your program should print the address of each element &array[i][j]. Are they continuous in memory?
Your program should print array[0], array[1], …, array[9]. Are they expected?
Written question: explain how your program get the values of array[0], array[1], …, array[9]
Test cases: no specific test case. Just run "lab3f"

Lab 3g: Dynamic two-dimensional array
The executable for this experiment should be lab3g.
Your program should create a dynamic two-dimensional array with m rows and n columns (m and n are from command line arguments), let array[i][j] = i*j; print all elements
Your program should print the address of each element &array[i][j]. Are they continuous in memory?
Your program should print array[0], array[1], …, array[9]. Are they expected?
Written question: explain how your program get the values of array[0], array[1], …, array[9]
Test cases: "lab3g 5 6" "lab3g 7 8"

Grading criteria:

1. We will unzip your zip file, and run "make". If it reports any errors, you will lose all points
2. We will then run your programs (1.5 points)
3. We will check your written answers (1 points)
4. Your program should have no memory leak or any other memory errors (0.5 points)