COMP2130 LAB 2 – Arrays, Pointers and Structures

Instructions: *You should create a zip file with the results of this lab. You should email this file to:* ***comp2130.2015@gmail.com****. Please submit this lab at the end of your class. The subject of the email must be: LAB2*

(1.)

Consider the following variable declarations. Assume each variable stores the values in a matrix. Write code that multiplies the matrix `a` by the matrix `b` and stores the result in the matrix `c` .

```
float a[2][3],b[3][2],c[2][2];
```

(2.)

Write a complete program that prompts the user for an input string, sorts its characters, and prints out the sorted output. Assume the string contains no spaces and is at most 30 characters. Sort the characters according to byte values, irrespective of the symbols those values represent, from smallest to largest. The output should be one contiguous string, printed on one line. Example: “Input: apple” should print `“aelpp”`.

(3.)

Write a program that looks at all the command line arguments and reports if any of the arguments are the same (i.e., they match exactly). The program should print out the matching argument and the positions it occupies in the list of arguments.

(4.)

A phone book typically lists the name, address, and telephone number of everyone living in an area. Write code defining a structure template that could be used to store this data. Assume that a name and address will be no more than 30 characters each, and that a telephone number has exactly seven digits. Write code that will enter records in your structure. Write a function that will accept a pointer to a phone book structure and present all the records. Write a function that accepts a name and locate the record matching that name.

(5.)

Use the following code to describe a rectangle. Assume the rectangle sides are parallel to the x and y axes (no rotation), and that the corners of the rectangle are located properly according to their compass designations in the structure.

```
struct point {
     int x,y;
};
struct rect {
     struct point ne,se,sw,nw;
};
```

Write a function called `RectArea()` that returns an integer value equal to the area of the given rectangle. The rectangle should be passed in as an argument.

THE END