# CS 2275 Exercise 4 – Arrays

Module 4 objectives:

* 1D arrays on the stack
* Simple pointers and 1D arrays on the heap
* Pass by reference, const ref
* Returning and array pointer from a function
* 2D arrays

Readings for Module 4: Lippman 3.5-3.6,8

Using Dev-C++, create a .cpp source file titled ex4-<your last name>.cpp that contains solutions to the following problems. You also will need a corresponding .doc file. As usual, include a great function name and a block comments with a single sentence indicating the purpose of the function.

1. Write a C++ method that accepts a 1 dimensional array of integers and an integer indicating it length, and reverses it. Note that in C++ you will also have to pass in the array size as an integer. Since you are actually passing in the pointer to the array, you can simply reverse it in place. You do not need to return anything and you should not create an unnecessary temp array. Ideally, write a recursive version of this function.
2. Write a C++ function that accepts a 1 dimensional array of integers and an integer indicating its length and returns the index of the minimal value. Do not use any built in functions. Do not sort the array first. If there are multiple values with the same minimal value, return the index of the first minimal value encountered.
3. Write a C++ function that accepts a 1 dimensional array of integers and an integer indicating its length and returns the index of the first occurrence of the maximal value.
4. Write a C++ function that accepts a two dimensional array of integers a, (any size), an integer indicating the number of columns, and an integer b, and returns true IFF (if and only if) b is in each row of a. NOTE: when I specify “write a function” you may always write several functions to implement the required function. HINT: This problem is much easier if you write more than one function.
5. Create a function frequencyCount that given an array of integers, and integer indicating that array’s length, and an integer target, returns the number of times target occurs in that array.
6. Write c++ function mean that accepts an array of integer and an integer indicating it length, and returns the mean of the values in that array. Do not use any built in mean function
   1. Indicate what tests will allow you to determine the correctness of your function. Test with a short array with a few values and also with a large (100 length), randomly filled array of integers between 0 and 999 (rand()). Display the array mean.
7. Write a C++ function unsigned int \*numeralFreqCount(const string &s) that accepts a string and returns a pointer to an array of length 10 containing the frequency count of the numerals ‘0’, ‘1’, …’9’ contained in that string. Thus numeralFreqCount (“0033344AA667”) would return an array with positions 1, 2, 5, 8, and 9 containing 0, position 0 containing 2, position 3 containing a 3, position 4 containing a 2, position 6 containing 2, and position 7 containing a 1. You will have to dynamically allocate the array inside the function and return its pointer. Be sure to delete it in main before returning.
8. Create a function standardDev that accepts and integer array and an integer indicating its length and returns the standard deviation of the values in a. The standard deviation is a statistical measure of the average distance each value in an array is from the mean (<http://en.wikipedia.org/wiki/Standard_deviation> ). To calculate the standard deviation, determine the mean using your mean method. Then sum the square of the difference of each value in the array and that mean. The standard deviation is the square root of that sum divided by the number of elements in the array. (The formal definition of STD is slightly different, but I simplify things here as this is a CS, not Math course ☺ ).

Submit your .cpp file using moodle. each.

Grading rubric: Problems 1-6, 12 points each. Problems 7 and 8, 14 points. Style up to -10: poor variable or poor function names or no block comment with a goal statement for a function. Solutions that have significantly more lines that needed will be docked points for lack of elegance.

**REMINDERS**

* code that does not compile will not be graded
* The grader should not need to modify/uncomment your code to test it. Provide a test mechanism allowing the grader to enter various tests for teach function.

The following code may help with #4:

const int fooRows = 3;

const int fooCols = 4;

int sum2DIntArray(int foo[][fooCols] ) {

int sum=0;

for(int r=0;r<fooRows;r++)

for(int c = 0;c<fooCols;c++)

sum +=foo[r][c];

return sum;

}

int main() {

int foo[fooRows][fooCols];

for(int r=0;r<fooRows;r++)

for(int c = 0;c<fooCols;c++)

foo[r][c] = rand()%100;

cout << sum2DIntArray(foo) << endl;

return 0;

}