NAME: ___________________________________

Learning Objectives

- Learn how to use vectors and arrays.
- Improve computational problem solving skills.

Instructions

Work out the answers to these problems manually without the use of a computer. This will help you develop the ability to read and analyze code.

Another reason to practice solving these problems manually is that problems of this type will appear on Quiz 4 and subsequent quizzes and exams. During the quiz/exam, you will not have access to a computer to solve these problems, so you need to develop the ability to read code, analyze it and think critically.

After coming up with answers manually, you can write and run test code to check whether you have solved the problems correctly.

Submit a single document with your answers either by email or through your remote git repository. If submitting through git, your document should be located in a folder named *a4*.

Problems

```
void appendVector(vector<int> & v, const vector<int> & w);
```

Figure 1

1) Provide an implementation of the *appendVector* function whose declaration is shown in Figure 1. Both arguments of the function are vectors of int. The function should modify vector *v* by appending to it all the elements of *w*. For example, if v = (4, 2, 5) and w = (11, 3), then after calling the function v will become (4, 2, 5, 11, 3). Hint: the vector class has function called push_back that appends values. (10 points)

```
int countOccurrences(const vector<int> & v, int k);
```

Figure 2

2) Implement the function *countOccurrences* whose declaration appears in Figure 2. The first argument of the function is a vector *v* of integers and the second argument is an integer *k*. The function returns the number of times *k* occurs in *v*. (10 points)

```
bool isStrictlyIncreasing(const vector<int> & v);
```

Figure 3

3) Write a predicate function called *isStrictlyIncreasing* that checks whether a vector of integers contains values that are in strictly increasing order. A sequence is strictly increasing if each item in the sequence is larger than the item that precedes it. A declaration of the function is shown in Figure 3. The function returns true if the elements are in strictly increasing order, otherwise it returns false. For example, it will return true for v = (3, 4, 7, 68, 81) and it will return false for (3, 4, 4, 5). Be careful not to attempt to access an item past the end of the vector.
(10 points)

```
bool isAllZeros(int a[ROWS][COLS]);
```

Figure 4

4) Suppose that a is a 2-dimensional array of int. Furthermore, suppose that a has number of rows equal to *ROWS* and number of columns equal to *COLS*, where *ROWS* and *COLS* are declared as const int in your program.

Implement a function that uses a loop nested inside another loop to determine whether a contains only zeros. A declaration of the function is shown above. Define *ROWS* and *COLS* as constants that specify the number of rows and columns of the array that is passed into *isAllZeros*. (10 points)

The following is an example declaration of *ROWS* and *COLS*.

```
const int ROWS = 3;
const int COLS = 2;
```