1). Write a function that takes two vectors of int as inputs and returns a vector which is the combination of given two vectors with duplicates removed. (i.e: {1,2,4} & {2,7,10} are given so output is {1,2,4,7,10}).

2). We'll say that an element in an array is alone if there are values before and after it, and those values are different from it. Write a function notAlone() that return a version of the given array where every instance of the given value which is alone is replaced by whichever value to its left or right is larger. • notAlone({1, 2, 3}, 2) returns {1, 3, 3}

• notAlone({1, 2, 3, 2, 5, 2}, 2) returns {1, 3, 3, 5, 5, 2}

• notAlone({3, 4}, 3) returns {3, 4}

3).Write a function scoresAverage() that given a vector of scores, compute the integer average of the first half and the second half, and return whichever is larger. We'll say that the second half begins at index length/2. The vector length will be at least 2. To practice decomposing a program into simple functions, write a separate helper method

int average(const vector& a, int start, int end)

which computes the average of the elements between indexes start..end. Call your helper method twice to implement scoresAverage(). Normally you would compute averages with doubles, but here we use ints so the expected results are exact.

• scoresAverage({2, 2, 4, 4}) returns 4.

• scoresAverage({4, 4, 4, 2, 2, 2}) returns 4.

• scoresAverage({3, 4, 5, 1, 2, 3}) returns 4.

4). Write a program that randomly fills in 0s and 1s into a 4-by-4 matrix, prints the matrix, and finds the first row and column with the most 1s. Here is a sample run of the program:

0 0 1 1

0 0 1 1

1 1 0 1

1 0 1 0

The largest row index: 2 The largest column index: 2

Implement and use following functions in your program:

// generate a random matrix of size m x n

vector<vector> genRandomMatrix(int m, int n)

// print a matrix

void printMatrix(const vector<vector>& a)

// find the row index with the most 1s

int findMaxOnesRow(const vector<vector>& a)

// find the column index with the most 1s

int findMaxOnesCol(const vector<vector>& a)

Note that the matrix is a vector of vectors. It can be created by

vector<vector> a(m,vector(n))

The number of rows of a is a.size() and the number of columns is a[0].size().

5) Write a C++ function removeDuplicates that takes a vector of strings as input and removes any duplicate elements, keeping only the first occurrence of each element. For example, given {1, 2, 3, 2, 4, 1}, it should modify the vector to {1, 2, 3, 4}.

6) Define a C++ function removeElement that takes a std::vector<int> and an integer element as input, and removes all occurrences of element from the vector. For example, if the input vector is {1, 2, 3, 2, 4, 1} and element is 2, the function should modify the vector to {1, 3, 4, 1}.

7) Create a C++ function medianString that takes a vector of strings as input, sorts them in lexographic order and returns their median.

8) Create a C++ function getSubset that takes a std::vector<int> and two integers start and end representing indices. The function should return a new vector containing elements from start to end (inclusive). If end exceeds the vector size, it should include elements up to the last element. For example, given {1, 2, 3, 4, 5, 6} with start = 2 and end = 4, it should return {3, 4, 5}.