CS6140: Advanced Programming Lab
Quiz 2

October 23, 2014

1 Question 1: Finding a Leader

The goal of this problem is to determine the leader of an election if one exists. Assume that we are given votes of n people for a set of contestants. We say that a contestant is a Leader if he/she gets more than $n/2$ votes. You are provided with the following cpp files (these are common for Question 1 and Question 2)

- main.cpp [do not edit]
- Header.cpp [do not edit]
- functions.cpp
- VotesFile1 VotesFile2 (input files)

In addition, you are provided a working makefile. First ensure that you are able to compile the files. In particular, executing "make main" from the command line should compile all the files and create an executable file called "main".

For Question 1, you can execute the following command
./main -p1 n *Inputfile*
Here "-p1" is for problem 1, the value n should be replaced with an integer denoting the number of votes present in the InputFile. Make sure that n is exactly equal to the number of lines in the file. The inputfile contains on each line a vote for each person. As of now, each vote is a string – however, you should not make any assumption on what each vote is. For your convenience, you are given a class called Vote (find this in Header.cpp). The main program reads the Inputfile and creates an array of size n where each entry is a pointer to an object of class Vote. Consider the n objects as votes of n people.

The class Vote which contains a private member which records the vote and only two public functions. The first public function is print which prints the vote value. The second public function takes as a parameter a pointer to another Vote object and returns 1 if the two votes are equal, 0 otherwise. Use this function to check whether two votes are equal or not. Note that there is no ordering between two votes. You are supposed to implement the function findLeader which is defined in the file functions.cpp and is described below. functions.cpp is the only file you need to edit. Do not change the class definition for the class Vote, in particular, you are not supposed to implement any public functions for the class Vote.

findLeader This function takes as input an array having pointers to objects of Vote class. The function finds a leader if one exists and returns the index of any Vote representing the leader. The function returns -1 otherwise. Recall the definition of the leader. Implement a divide and conquer algorithm that executes in $O(n \log n)$ time for the question. Implementing this correctly will fetch you 10 marks. An implementation which is asymptotically worse than $O(n \log n)$ will get no more than 4 marks.

You are free to implement auxiliary functions required in functions.cpp – but DO NOT edit class definitions.

2 Question 2: Finding a sub-matrix with largest possible sum

The input to this problem is a $m \times n$ matrix of integers such that that the entries of the matrix are given in descending order along each row and column. The $(1, 1)$ entry is guaranteed to be non-negative. Our goal is to compute the largest weight sub matrix, that is, the sub matrix such that the sum of its entries is maximised.

For example:

```
3 4
29 20 14 0
25 5 -4 -10
-23 -45 -50 -63
```

Notice that you are required to return the sum of the weights of the entries in the largest weight sub matrix. In the above input example, the largest weight sub matrix is:

```
29 20 14 0
25 5 -4 -10
```

and you only have to return the sum of the weights of the entries in the largest weight sub matrix which is 89. (Can you see why?)
You are provided with the following cpp files (these are common for Question 1 and Question 2)

- main.cpp [do not edit]
- Header.cpp [do not edit]
- functions.cpp
- MatrixFile1 MatrixFile2 (input files)

The input files are integer matrices formatted in the following way. The first line has two positive integers giving the number of rows and the number of columns. Each subsequent row in the input file corresponds to a row in the input matrix, with entries separated by a space.

For Question 2, you can execute the following command
./main -p2 *Inputfile*
Here "-p2" is for problem 2 and the inputfile contains the description of the matrix as given above. The Header.cpp file has a class Matrix implemented which that has basic facility to populate a 2D array with the entries of the matrix read from an inputfile. A `print` function is provided for your convenience. You are required to implement a function `findMaxSubmatrixSum` described below. As earlier, do not change any class definitions and edit ONLY the file functions.cpp

findMaxSubmatrixSum This function takes as input a matrix and returns the sum of the largest weight sub-matrix. Implement a dynamic programming algorithm which runs in $O(n^2)$ time. This will fetch you 10 marks. While you implement this function, also write as comments in the function the definition of the sub-problem definition and the way to compute these sub-problems efficiently. This will help us in evaluation of your code.

3 Hints Penalty

For Question 2 there is a hint available at a penalty of 4 marks. Note that this hint will only provide you the sub-problem definition and the way to compute the sub-problem. You will not be provided with additional code.

4 Submission

You are required to submit your "functions.cpp" file, but it must be renamed "⟨rollnumber⟩.cpp" and tar zipped before you submit it. Please ensure that the file compiles without any error before you submit it.