COL 100 Minor 2 Exam

October 4, 2018. 8:00 am - 9:00 am

Name:

Entry Number:

Group:

Notes:

- Total number of questions: 4. Max Marks: 20
- All answers should be written on the question paper itself.
- The last two sheets in the question paper are meant for rough work. If you run out of space, you can answer questions in this rough space. However, please clearly mention in the answer space for the appropriate question that we should look at the rough space for its answer.
- We will collect the question paper (including the last two sheets meant for rough work). We will not be collecting back any other rough sheets.

1. Write a function *countDigits*(*string str*) which inputs a string and counts the number of (1) digits (2) lower case letters from English alphabet (3) uppercase letters from English alphabet in the string. For example, if your input string is "COL100-Minor2Exam" then you program should output:

```
number of digits: 4
number of lowercase letters: 7
number of uppercase letters: 5
```

You can make use of the following functions: *isdigit*(*char ch*) returns true if input character *ch* is a digit false othersiwe. *islower*(*char ch*): returns true if the input character *ch* is a lower case letter, false otherwise. *isupper*(*char ch*): returns true if the input character *ch* is upper case letter, false otherwise. [5 marks]

```
bool isdigit(char c)
{
  return ((int)c > 47 && (int)c < 58) ? true : false;
}

bool islower(char c)
{
  return ((int)c > 96 && (int)c < 123) ? true : false;
}

bool isupper(char c)
{
  return ((int)c > 64 && (int)c < 91) ? true : false;
}

void countDigits(string str)
{
  int numDigits = 0;
  int numlettersLow = 0;
```

```
  int numlettersUpp = 0;
  int len = str.size();
  for(int i=0; i<len; i++){
    if(isdigit(str[i])) numDigits++;
    else if(islower(str[i])) numlettersLow++;
    else if(isupper(str[i])) numlettersUpp++;
  }
  cout << "number of digits: " << numDigits << endl;
  cout << "number of lowercase letters: " << numlettersLow << endl;
  cout << "number of uppercase letters: " << numlettersUpp << endl;
}
```

2. Let A be an $m \times n$ matrix of integers. Frobenius norm of the matrix A is the sum of the squares of each of its elements. In other words, $||A||_F = \sum_{i,j=1,1}^{i,j=m,n} A_{ij} * A_{ij}$, where $||A||_F$ denotes the Frobenius norm and A_{ij} denotes the ij^{th} element of A. Write a function $getFrobeniusNorm(Grid<int>\ A)$ which inputs a matrix A represented as a grid, and returns the Frobenius norm of A. [4 marks]

```
int getFrobeniusNorm(Grid<int> A)
{
  int A_row = A.numRows();
  int A_col = A.numCols();
  int fnorm = 0;
  for(int i=0; i<A_row; i++)
    for(int j=0; j<A_col; j++)
      fnorm += A[i][j] * A[i][j];
  return fnorm;
}
```

3. We say that a Vector of integers is sorted in ascending order if all of the element at position i is always less than equal to the element at position $i+1$ for all possisble values of i. For example, Vector $\{1, 4, 5, 7, 7, 11\}$ is Vector sorted in ascending order. Similarly, Vector $\{2, 4, 8, 10\}$ is also sorted. Given two sorted vectors $v1$ and $v2$, we say that $v3$ is corresponding merged vector if (a) $v3$ contains exactly the elements present in both $v1$ and $v2$ (with duplicates repeated as many times as in original vectors) (b) the sorted order of the original vectors is maintained. In other words, each element of $v3$ is either an element of $v1$ or $v2$ such that $v3$ is still sorted. For the example above, the merged vector $v3$ would be $\{1, 2, 4, 4, 5, 7, 7, 8, 10, 11\}$. Write a function $merge(Vector<int>\ v1, Vector<int>\ v2)$ which inputs two sorted Vectors $v1$ and $v2$ and returns the resultant of merging the elements of $v1$ and $v2$. Your program should run in $O(m+n)$ time where m and n are sizes of the two vectors. [6 marks]

```
vector<int> merge(vector<int> v1, vector<int> v2)
{
    int len_v1 = v1.size();
    int len_v2 = v2.size();
    int len_v3 = len_v1 + len_v2;
    vector<int> v3(len_v3);
    int idx_v1 =0;
    int idx_v2 =0;
    for(int i=0; i< len_v3; i++)
    {
      if(idx_v1 == len_v1) v3[i] = v2[idx_v2++];
      else if(idx_v2 == len_v2) v3[i] = v1[idx_v1++];
      else if(v1[idx_v1] < v2[idx_v2]) v3[i] = v1[idx_v1++];
      else v3[i] = v2[idx_v2++];
    }
    return v3;
}
```

4. Suppose you are working with an application which needs to write the contents of a file to another file but only the even numbered lines should be copied. Write a program which inputs a string from the user: $filename$, creates an input stream over file named $filename$, reads the contents of this file, and selectively writes the even numbered lines to another file named "out.txt". Note that you will have to create an output stream over "out.txt". Also, do not forget to close the files at the end of the operation. For example, if the file has the following content:

```
col100 is going well.
some more practice will help.
it is good to know so many students are doing col100.
I hope the Minor 2 goes well for everyone.
```

Then your output file should contain the following lines:

```
some more practice will help.
I hope the Minor 2 goes well for everyone.
```

[5 marks].

```
#include <iostream>
#include <vector>
#include <string>
#include <fstream>

using namespace std;

int main(){
    string filename;
    cin >> filename;
    ifstream inputfile;
    inputfile.open(filename.c_str(), std::ifstream::in);

    if(!inputfile.is_open())
    {
```

```
    cout << "Error: Cannot open " << filename << endl;
    return 0;
  }
  ofstream outputfile;
  outputfile.open("out.txt", std::ifstream::out);
  if(!outputfile.is_open())
  {
    cout << "Error: Cannot open " << "out.txt" << endl;
    return 0;
  }
  string line;
  bool even_line = 0;
  while (getline(inputfile, line)) {
    if(even_line) outputfile << line << endl;
    even_line = !even_line;
  }
  inputfile.close();
  outputfile.close();
}
```