COL 100 Minor. Make-up Exam

Friday October 26, 2018. 5:30 pm - 6:30 pm

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. Given two vectors $v1$ and $v2$ (both containing numbers), we say that $v1$ is subsumed by $v2$ if all the elements in $v1$ also occur in $v2$. For example, if $v1$={1,2,3,6} and $v2$={1,2,3,4,6} then $v1$ is subsumed by $v2$ since every elemnt of $v1$ also occurs in $v2$. For this problem, you can assume that $v1$ does not contain any duplicate elements. Similarly for $v2$.

   (a) Write a function $isSubsumed(Vector{<}int{>}v1, Vector{<}int{>}v2)$ which takes as argument two vectors $v1$ and $v2$, and returns true if $v1$ is subsumed by $v2$, false otherwise. What is the time complexity of your function in terms of m, n where m and n are sizes of $v1$ and $v2$, respectively. [2 points]

   Solution:

```
bool isSubsumed(vector <int> v1,vector<int> v2)
{
        int m= v1.size();
        int n = v2.size();
        if(m>n)
              return false;
        else
        {
              int i,j;
              for(i=0;i<m;i++)
              {
                    for(j=0;j<n;j++)
                    {
                          if(v1[i]==v2[j])
                                break;
                    }
                    /* If above inner loop was not broken at any
point,
                    then v1 is not subsumed by v2 */
                    if(j==n)
                          return false;
              }
                    /* If we reach here then all elements
```

```
                of v1 are present in v2 */
            return true;
        }
    }
Time complexity- $O(mn)$
```

(b) Now assume that the vectors v1 and v2 are **sorted** in ascending order. Can you write a more efficient function to achieve the above functionality? In particular, write a function to achieve the above task in time $O(m + n)$ by exploiting the fact that two vectors are sorted. As earlier, m and n here are the respective sizes of the two vectors. [4 points]

Solution:

```
bool isSubsumed(vector <int> v1,vector<int> v2)
{
        int m= v1.size();
        int n = v2.size();
        if(m>n)
              return false;
        else
        {
              int i=0,j=0;
             while(i<n && j<n)
             {
                    if (v1[i]==v2[j])
                    {
                          i++;
                          j++;
                    }
                    else if(v1[i]<v2[j])
                    return false;
                    else /*if v1[i] is smaller then
                    checks if it occurs after v2[j]*/
                          j++;
             }
              /* If we reach here then all elements of v1
```

```
                are present in v2[] */
                if(i==n)
                        return true;
                 else
                        return false;
          }
  }
```

2. Assume that you are given a file where the lines are sorted in lexicographic order. The file may contain duplicates (repeated lines). We would like to print the contents of this file to another file with duplicates removed. Write a program achieve this functionality. Your program should inputs a filename from the user, read the contents of this file, and then output the contents to a new file with duplicates removed. You can assume that the contents (lines) of the input file are sorted lexicographically. For example, if the input file is:

   ```
   abhijeet kundu
   nilesh gautam
   nilesh gautam
   surya jawar
   zohar bedi
   zohar bedi
   ```

   then, the output file should be:

   ```
   abhijeet kundu
   nilesh gautam
   surya jawar
   zohar bedi
   ```

   Your program should run in $O(n)$ time where n is the number of lines in the input file. [5 points]

Solution:

```
#include<iostream>
#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 current_line="", prev_line="";
    bool first = true;    //check if it is first line of the file
    // read the next line in current_line
    while (getline(inputfile, current_line))
    {
        // only if current_line is different from the previous line,
        // write current_line to the output
        if (first || current_line != prev_line) {
            first = false;
            outputfile << current_line << endl;
            // update prev_line to current_line
            prev_line = current_line;
        }
    }
    inputfile.close();
    outputfile.close();
    return 0;
}
```

3. (a) Write a function *power*(*double x*, *int m*) which takes two arguments x and m and computes (and returns) x raised to the power m. E.g. if $x = 3$, and $m = 2$, then your function should return 9. You can assume x is real number and m is any positive integer. How much time does your program take to run in terms of the complexity notation? [1 point]

Solution:

```
double power(double x,int m)
{
    double ans=1;
    //Multiply x m times
    for(int i=1;i<=m;i++)
        ans=ans*x;

    return ans;
}
```

Time complexity-O(m)

(b) Now, let us assume additionally that m is a power of 2, i.e., $m = 2^k$ for some integer $k > 0$. Write an improved program (which is much more efficient in terms of order complexity) by exploiting this fact. Hint: you can use the fact that $x^m = x^{\frac{m}{2}} * x^{\frac{m}{2}}$. How much time does your program take to run in terms of k in this case? [4 points]

Solution:

```
double power(double x,int m)
{
        double ans=x; //Initialize answer
        while(m>1)
        {
            ans = ans * ans; //square the current answer
                                  and reduce m to m/2
            m = m/2;
        }
        return ans;
}
```

Time complexity - O(k)

4. Sakshi likes to play tennis based on the day of the week and weather condition. She makes her decision depending on the following four factors.

   - Day of the week (which can be mon, tue, wed, thu, fri, sat or sun)
   - Outlook (which can be sunny, rainy or overcast)
   - Wind (which can be strong or weak)
   - Humidity (which can be high or normal)

   Sakshi plays tennis only on weekends (sat/sun). Further, even on a weekend, she plays tennis only if i) outlook is overcast OR ii) outlook is rainy and wind is weak OR iii) outlook is sunny and humidity is normal. We would like to automate her decision about playing tennis by creating a program which would take as input the value of (a) Day of the week (b) Outlook (b) Wind and (c) Humidity variables from the user, and decide whether Sakshi is going to play tennis or not on the given day. Write a program to achieve this functionality. [4 points]

   Solution:

```
#include< iostream >
using namespace std;
int main()
{
        string day,outlook,wind,humidity;
        cin >> day >> outlook >> wind >> humidity;
       if(day == "sat"||day == "sun")
        {
        //return yes based on the value of the variables and conditions
specified
              if(outlook == "overcast"||(outlook == "rainy"&&wind ==
"weak")||(outlook == "sunny"&&humidity == "normal"))
                    cout <<"Yes she will go out to play";
              else
                         cout <<"No she will not go out to play";
        }
       else //if it's any weekday then return No
               cout <<"No she will not go out to play";
        return 0;
}
```

Rough page 1 Rough page 2