Problems on 2D Array - 2

Assignment Solutions

PW SKILLS

Q1. Given a m*n matrix, Write a function which returns true if the matrix is a perfect matrix. A matrix is called perfect if every diagonal from top-left to bottom-right has the same elements.

Sample Input: arr[]=[[9,8,7,6],[5,9,8,7],[1,5,9,8]]

9	8	7	6
5	9	8	7
1	5	9	8

Sample Output: true

Explanation:We will skip the first row and column and then traverse each row(i) and check if the diagonal elements(i-1,j-1) are equal or not, if not then return false.

Code:

```
#include <iostream>
#include <vector>
using namespace std;
bool check(vector<vector<int>>& mat)
{
   int r=mat.size();
   int c=mat[0].size();
   for(int i=1;i<r;i++)
   {
       for(int j=1;j<c;j++)
       {
           if(mat[i][j]!=mat[i-1][j-1])
           {
               return false;
           }
       }
   }
   return true;
}

int main()
{
    vector<vector<int>> mat={{9,8,7,6},{5,9,8,7},{1,5,9,8}}  ;
    if(check(mat))
    {
       cout<<"true";
    }else{
       cout<<"false";
    }
}
```

```
true

...Program finished with exit code 0
Press ENTER to exit console.
```

Q2. Given an array of intervals where intervals[i] = [start, end], merge all overlapping intervals, and create a function which returns a vector of the non-overlapping intervals that cover all the intervals in the input.

Sample Input: arr[]=[[1,4],[2,3],[5,8],[6,9]]
Sample Output: [[1,4],[5,9]]

Sample Input: arr[]=[[1,5],[3,9]]
Sample Output: [1,9]

Explanation: We need to merge the overlapping intervals i.e. we need to check if the starting point(i) of an interval is less than or equal to the ending point of the previous interval(j-1) then we need to merge it.

Code:

```
#include <iostream>
#include <vector>
using namespace std;
vector<vector<int>> merge(vector<vector<int>>& s) {
       vector<vector<int>> ans;
       int j=0;
       ans.push_back(s[0]);
       for(int i=1;i<s.size();i++)
       {
           if(ans[j][1]>=s[i][0])
           {
               ans[j][1]=max(ans[j][1],s[i][1]);
           }else{
               j++;
               ans.push_back(s[i]);
           }
       }
       return ans;
   }

int main()
{
   vector<vector<int>> mat={{1,4},{2,3},{5,8},{6,9}};
   vector<vector<int>> ans=merge(mat);
   for(int i=0;i<ans.size();i++)
   {
       cout<<ans[i][0]<<" "<<ans[i][1]<<endl;
   }

}
```

```
1 4
5 9


...Program finished with exit code 0
Press ENTER to exit console.
```

Q3. Given an array of intervals where arr[i] = [start, end], return the minimum number of intervals you need to remove to make the rest of the intervals non-overlapping.[

Sample Input: arr[]=[[1,4],[2,3],[4,5],[6,7]]
Sample Output: 1

Explanation: The interval 1,4 and 2,3 are overlapping so removing any one of them will make the intervals non overlapping.

Sample Input: arr[]=[[1,2],[2,3],[3,4],[4,5]]
Sample Output: 0

Explanation: Just like the previous question we need to merge the overlapping intervals and count how many intervals we are merging using the count variable.

Code:

```
#include <iostream>
#include <vector>
using namespace std;
int merge(vector<vector<int>>& s) {
       vector<vector<int>> ans;
       int j=0;
       ans.push_back(s[0]);
       int count=0;
       for(int i=1;i<s.size();i++)
       {
           if(ans[j][1]>s[i][0])
           {
               count++;
               ans[j][1]=max(ans[j][1],s[i][1]);
           }else{
               j++;
               ans.push_back(s[i]);
           }
       }
       return count;
   }

int main()
{
   vector<vector<int>> mat={{1,4},{2,3},{4,5},{6,7}};
   cout<<merge(mat);
}
```

```
1

...Program finished with exit code 0
Press ENTER to exit console.
```