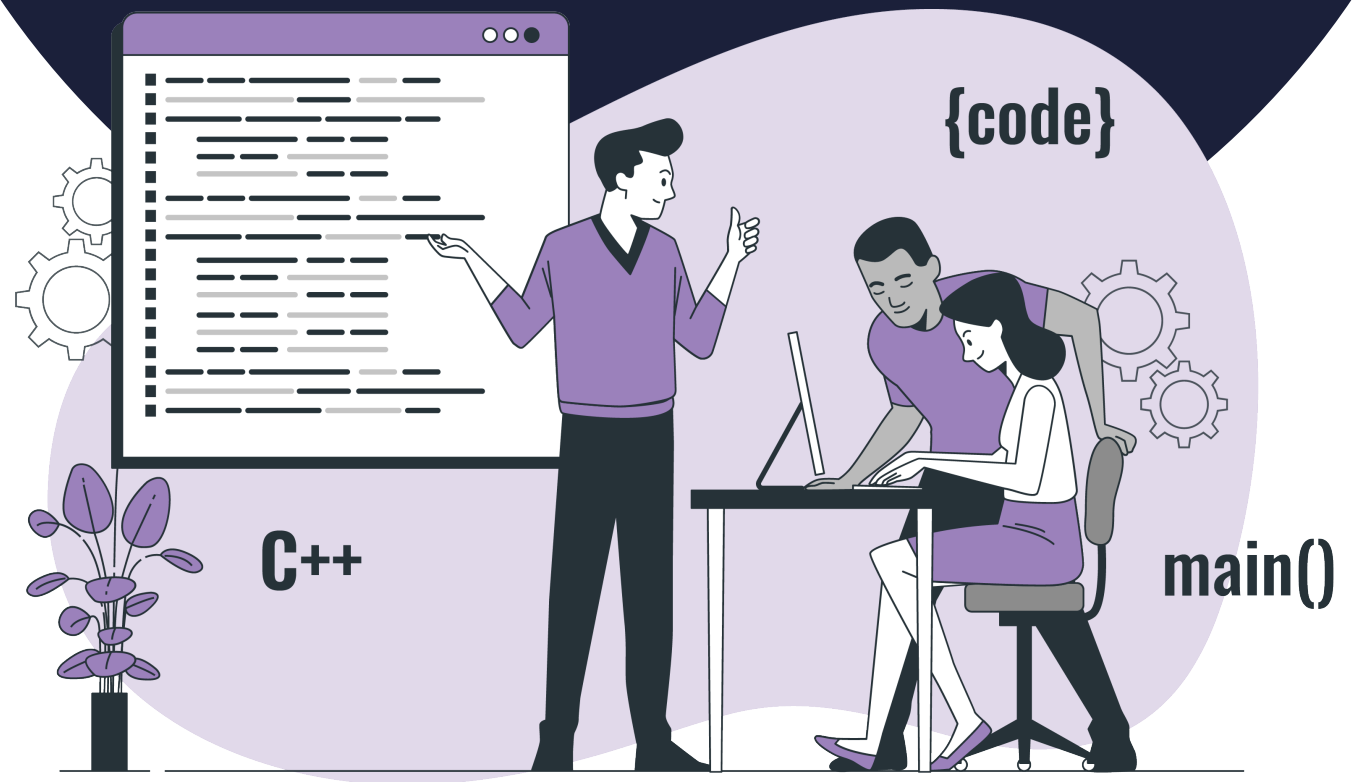
Lesson:

2D Array Problems Set 5



Pre-Requisites

- 2D Arrays

List of Concepts Involved

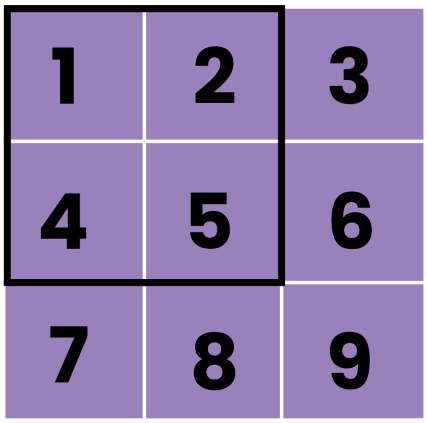
- Prefix Sums in 2D Arrays

Pattern: Prefix Sums in 2D Arrays

Problem 1: Given a matrix 'a' of dimension n x m and 2 coordinates (l1, r1) and (l2, r2). Return the sum of the rectangle from (l1,r1) to (l2, r2).

Input
Enter number of rows : 3
Enter number of columns :3
Enter the edges for which you want the prefix sum (in order l1,l2,r1,r2):
0 1 0 1
Enter elements of the array :
1 2 3
4 5 6
7 8 9

Output
value is : 12



Method 1: Brute Force

Explanation Method 1: We add the value of each element in the given range from l1 to l2 and from r1 to r2. This can be done using a brute force approach where we use two nested for loops and keep adding the value of the elements to the answer.

```
#include<iostream>
#include<vector>
using namespace std;

int solve(vector<vector<int>> a, int l1, int l2, int r1, int r2){
  int ans = 0;
  for(int i = l1; i <=  l2; i++) {
    for(int j = r1; j <= r2; j++) {
      ans += a[i][j];
    }
  }
  return ans;
}


int main(){
     int row , col ,l1, l2 , r1 , r2;
    cout<<"Enter number of rows : ";
    cin>>row;

    cout<<"Enter number of columns :";
    cin>>col;

    cout<<"Enter the edges for which you want the prefix sum (in order l1,l2,r1,r2): ";
    cin>>l1>>l2>>r1>>r2;

    vector<vector<int>>arr(row , vector<int>(col));
cout<<"Enter elements of the array :";
    for(int i=0;i<row;i++){
        for (int j=0;j<col;j++){
            cin>>arr[i][j];
        }
    }

    cout<<"value is : "<<solve(arr,l1,l2,r1,r2)<<endl;
}
```

OUTPUT :

```
Enter number of rows : 3
Enter number of columns :3
Enter the edges for which you want the prefix sum (in order l1,l2,r1,r2): 0 1 2 2
Enter elements of the array :
2 4 5
3 9 8
7 6 0
value is : 13
```

Method 2: Pre-Calculating the horizontal sum for each row in the Matrix

Explanation Method 2: We add the value of each element in the given range from l1 to l2 and from r1 to r2. Rather than using the standard brute force approach we try to optimize it and pre-calculate the horizontal sum for each row in the matrix. This can be stored in the same matrix. Now we can traverse only the rows through a single for loop rather than using a nested for loop as we have a precalculated prefix matrix and calculate the answer using prefix sum technique.

```
#include<iostream>
#include<vector>
using namespace std;

int solve(vector<vector<int>> a, int l1, int l2, int r1, int r2)
{
    // convert each row of the matrix into prefix-sum vector
    int n = a.size(), m = a[0].size();
    for (int i = 0; i < n; i++)
    {
        for (int j = 1; j < m; j++)
        {
            a[i][j] += a[i][j - 1];
        }
    }
    int ans = 0;
    for (int i = l1; i <= l2; i++)
    {
        ans += a[i][r2] - (r1 >= 1 ? a[i][r1 - 1] : 0);
    }
    return ans;
}


int main(){
     int row , col ,l1, l2 , r1 , r2;
    cout<<"Enter number of rows : ";
    cin>>row;

    cout<<"Enter number of columns :";
    cin>>col;
```

```
    cout<<"Enter the edges for which you want the prefix sum (in order l1,l2,r1,r2): ";
        cin>>l1>>l2>>r1>>r2;

        vector<vector<int>>arr(row , vector<int>(col));
    cout<<"Enter elements of the array :";
        for(int i=0;i<row;i++){
            for (int j=0;j<col;j++){
                cin>>arr[i][j];
            }
        }

        cout<<"value is : "<<solve(arr,l1,l2,r1,r2)<<endl;
    }
```

OUTPUT :

```
Enter number of rows : 3
Enter number of columns :3
Enter the edges for which you want the prefix sum (in order l1,l2,r1,r2): 0 1 2 2
Enter elements of the :
2 4 5
3 9 8
7 6 0
value is : 13
```

Method 3: Prefix sum Over columns and Rows Both

Explanation Method 3: We add the value of each element in the given range from l1 to l2 and from r1 to r2. Now we even try to optimize the previous method by creating a prefix sum of both row and column in the same matrix. This can be precalculated as horizontal prefix sum for every particular row and vertical prefix sum for columns. Now we can calculate the final answer without using any for loop as we have pre calculated the prefix matrix.

```cpp
#include<iostream>
#include<vector>
using namespace std;

int solve(vector<vector<int>> a, int l1, int l2, int r1, int r2)
{
    int n = a.size(), m = a[0].size();
    // convert the first row into prefix vector
    for (int j = 1; j < m; j++)
    {
        a[0][j] += a[0][j - 1];
    }
    // convert the first col into prefix vector
    for (int i = 1; i < n; i++)
    {
        a[i][0] += a[i - 1][0];
    }
    // convert the matrix into prefix matrix
    for (int i = 1; i < n; i++)
    {
        for (int j = 1; j < m; j++)
        {
            a[i][j] += a[i - 1][j] + a[i][j - 1] - a[i - 1][j - 1];
        }
    }
    int full = a[r2][l2];
    int top = (l1 >= 1 ? a[l1 - 1][r1] : 0);
    int left = (r1 >= 1 ? a[l2][r1 - 1] : 0);
    int double_subtracted = (l1 >= 1 && r1 >= 1 ? a[l1 - 1][r1 - 1] : 0);
    int ans = full - top - left + double_subtracted;
    return ans;
}



int main(){
     int row , col ,l1, l2 , r1 , r2;
    cout<<"Enter number of rows : ";
    cin>>row;

    cout<<"Enter number of columns :";
    cin>>col;

    cout<<"Enter the edges for which you want the prefix sum (in order l1,l2,r1,r2): ";
    cin>>l1>>l2>>r1>>r2;

    vector<vector<int>>arr(row , vector<int>(col));
cout<<"Enter elements of the array :";
    for(int i=0;i<row;i++){
        for (int j=0;j<col;j++){
            cin>>arr[i][j];
        }
    }

    cout<<"value is : "<<solve(arr,l1,l2,r1,r2)<<endl;
}
```

OUTPUT :

```
Enter number of rows : 3
Enter number of columns :3
Enter the edges for which you want the prefix sum (in order l1,l2,r1,r2):
0 1 2 2
Enter elements of the array :
2 4 5
3 9 8
7 6 0
value is : 13
```

Upcoming Class Teasers

- Time and Space Complexity