Todays Content:

1. Max submatrix sum
2. Max expressions

28 Given a mat[N][M] return max submatrix sum.

Eg: mat[5][6]

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | -4 | 3 | 7 | -14 | 3 | 8 |
| 1 | 3 | 7 | 2 | 3 | 2 | 1 |
| 2 | -6 | 4 | -1 | 3 | -3 | 2 |
| 3 | 2 | 7 | 6 | -8 | 4 | -5 |
| 4 | 1 | -4 | 3 | -8 | 2 | 6 |

ans = 42

Idea1: Generate all submatrices:

For each submatrix, iterate & calculate sum & return overall max.

TC: $O(N^2*M^2) * O(N*M) = O(N^3M^3)$ SC: O(1)

↳ #submatrics: Write all explanation.

Idea2: Generate all submatrices:

For each submatrix, calculate sum using Pf mat[][]

return overall max.

TC: $O(N^2*M^2) * O(1) = O(N^2M^2)$ SC: $O(N*M)$

↳ #Sum using Pf mat[][]

Idea3:

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | -4 | 3 | 7 | -14 | 3 | 8 |
| 1 | 3 | 7 | 2 | 3 | 2 | 1 |
| 2 | -6 | 4 | -1 | 3 | -3 | 2 |
| 3 | 2 | 7 | 6 | 8 | 4 | -5 |
| 4 | 1 | -4 | 3 | -8 | 2 | 6 |

#obs: Any rectangle will have
2 boundary rows
2 boundary cols

Hint1: Fix any 2 rows & calculate max submatrix sum among them.

1.

2.

#Note1:

Max SubSum: rows=1, rowe=3

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | -4 | 3 | 7 | -14 | 3 | 8 |
| 1 | 3 | 7 | 2 | 3 | 2 | 1 |
| 2 | -6 | 4 | -1 | 3 | -3 | 2 |
| 3 | 2 | 7 | 6 | 8 | 4 | -5 |
| 4 | 1 | -4 | 3 | -8 | 2 | 6 |

| $S_0$ | $S_1$ | $S_2$ | $S_3$ | $S_4$ | $S_5$ |
|---|---|---|---|---|---|
| -1 | 18 | 7 | 14 | 3 | -2 |

Max SubSum: rows=3, rowe=4

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | -4 | 3 | 7 | -14 | 3 | 8 |
| 1 | 3 | 7 | 2 | 3 | 2 | 1 |
| 2 | -6 | 4 | -1 | 3 | -3 | 2 |
| 3 | 2 | 7 | 6 | 8 | 4 | -7 |
| 4 | 1 | -4 | 3 | -8 | 2 | 6 |

| $S_0$ | $S_1$ | $S_2$ | $S_3$ | $S_4$ | $S_5$ |
|---|---|---|---|---|---|
| 3 | 3 | 9 | 0 | 6 | -1 |

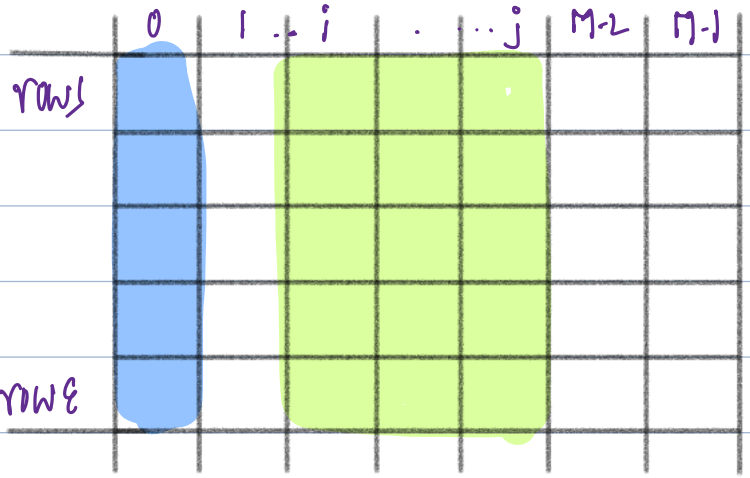
Idea3:

For every rows & rowe:

Calculate maxSubmatrix sum.

# Calculate sum of each column from rows to rowe

Q: How to get these column sum.




cols: $S_0$ $S_1$ $S_i$ $S_{i+1}$ .. $S_j$ .. $S_{M-1}$

4.

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | -4 | 3 | 7 | -14 | 3 | 8 |
| 1 | 3 | 7 | 2 | 3 | 2 | 1 |
| 2 | -6 | 4 | -1 | 3 | -3 | 2 |
| 3 | 2 | 7 | 6 | 8 | 4 | -7 |
| 4 | 1 | -4 | 3 | -8 | 2 | 6 |

| | | | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|
| rows | rowe | S[]: | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | S[]: | 3 | 7 | 2 | 3 | 2 | 1 |
| 1 | 2 | S[]: | -3 | 11 | 1 | 6 | -1 | 3 |
| 1 | 3 | S[]: | -1 | 18 | 7 | 14 | 3 | -4 |

```
int SubSum(int mat[][], int N, int M){ TC: O(N^2*M) = O(N^2M)
                                       SC: O(M)
    long ans = INT_MIN;
    for(int i=0; i<N; i++){ # i : rows
        vector<int> s(M,0);
        for(int j=i; j<N; j++){ # j : row E
            # New row is mat[j], add it in sum;
            for(int k=0; k<M; k++){
                s[k] += mat[j][k]
            }
                                  → Implement this.
            ans = max(ans, kadanes(s));
        }
    }
    return ans;
}
```

#Note:

HB:

Given ar[N] find Max of expression $\{ |ar[i] - ar[j]| + |i-j| \}$

Note: (i, j) are indices of pair

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| ar[] = { | -3 | 6 | 8 | 2 | 7 | 10 | -10 | 5 } |

Dry Run:

| (i, j) : | $\|ar[i] - ar[j]\| + \|i-j\|$ |
|---|---|
| 5, 6 | $\|10 - (-10)\| + \|5-6\| = 20 + 1 = 21$ |
| 2, 6 | $\|8 - (-10)\| + \|2-6\| = 18 + 4 = 22$ |

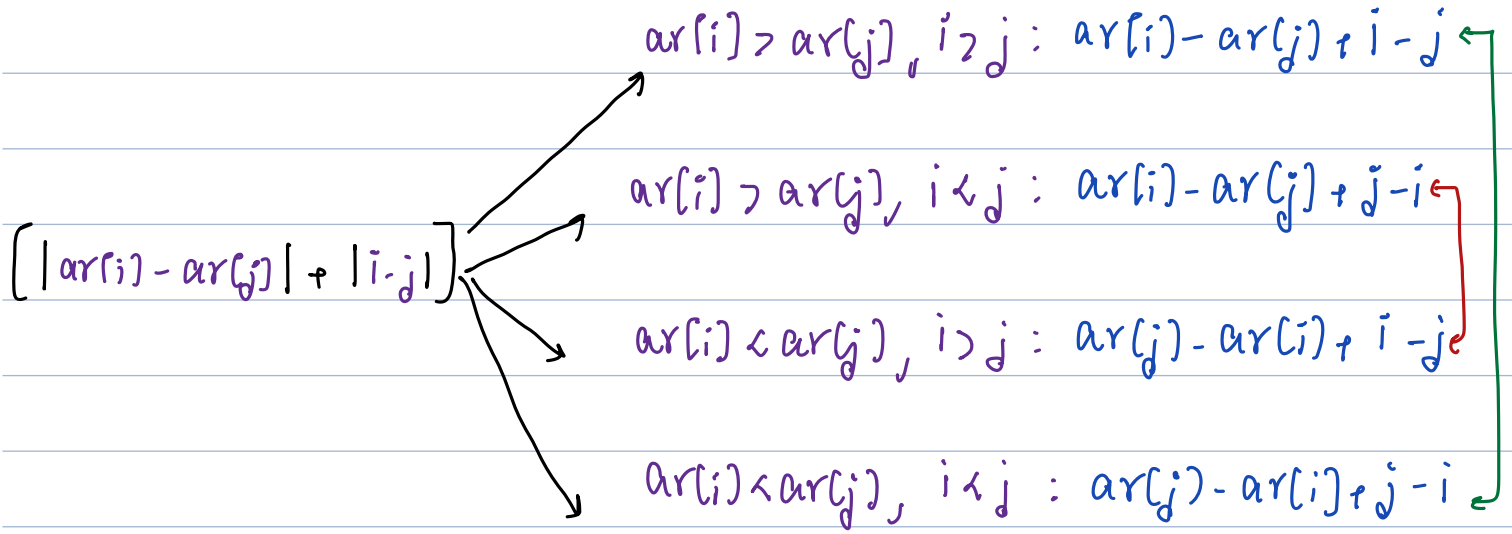
Idea1: Generate all pairs

for all pairs, calculate expression value & get average max.

TC: O(N²) SC: O(1)

Idea2: if $n > 0$ $|n| = n$

$n < 0$ $|n| = -n$




# Claim is final pair should belong to one of the above expressions, since we don't know which calculate max for all expressions & return final ans.

# Calculate max of below expressions

1. Find pair (i, j) such that ar[i] - ar[j] + i - j is max?

= ar[i] - ar[j] + i - j

= ar[i] + i - ar[j] - j

= (ar[i] + i) - (ar[j] + j)

# In above expression for each ar[k] we add index.

Iterate in array calculate max ar[k] + k = Sn

Iterate in array calculate min ar[k] + k = Sy

value = Sn - Sy

2. Find pair (i, j) such that ar[j] - ar[i] + j - i is max?

= ar[j] - ar[i] + j - i

= ar[j] + j - i - ar[i]

= (ar[j] + j) - (i + ar[i])

# In above expression for each ar[k] we add index.

Iterate in array calculate max ar[k] + k = Sn

Iterate in array calculate min ar[k] + k = Sy

value = Sn - Sy

3. Find pair (i, j) such that ar[i] - ar[j] + j - i is max?

= ar[i] - ar[j] + j - i

= ar[i] - i + j - ar[j]

= (ar[i] - i) - (ar[j] - j)

# In above expression for each ar[k] we subtract index

Iterate in array calculate max ar[k] - k = Sx

Iterate in array calculate min ar[k] - k = Sy

Value = Sx - Sy

4. Find pair (i, j) such that ar[j] - ar[i] + i - j is max?

= ar[j] - ar[i] + i - j

= ar[j] - j + i - ar[i]

= ar[j] - j - (ar[i] - i)

# In above expression for each ar[k] we subtract index

Iterate in array calculate max ar[k] - k = Sx

Iterate in array calculate min ar[k] - k = Sy

Value = Sx - Sy

```
int mindiff(vector<int> arr, int N){
    int case1=0, max1=INT_MIN, min1=INT_MAX;
    for(int i=0; i<N; i++){
        max1 = Math.max(max1, arr[i]+i);
        min1 = Math.min(min1, arr[i]+i);
    }
    case1 = max1 - min1;
    int case2=0, max2=INT_MIN, min2=INT_MAX;
    for(int i=0; i<N; i++){
        max2 = Math.max(max2, arr[i]-i)
        min2 = Math.min(min2, arr[i]-i);
    }
    case2 = max2 - min2;
    return Math.max(case1, case2);
}
```