ARRAY AND MATH – II

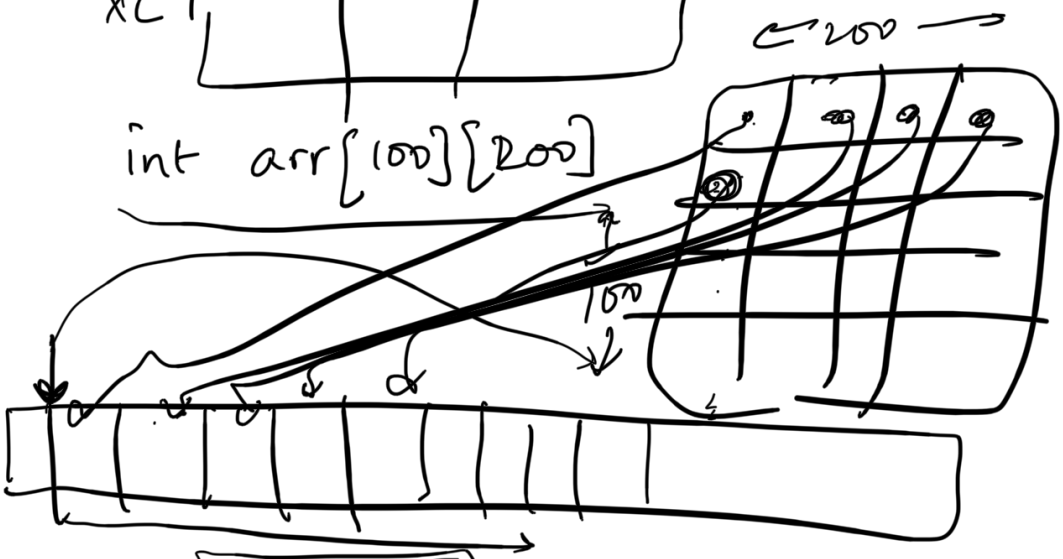
- 2-D arrays (Multi-D)
- Maths

Static

int arr[100]

dynamic

vector
Arraylist



arr [4] [5]

0, 1, 2, 3

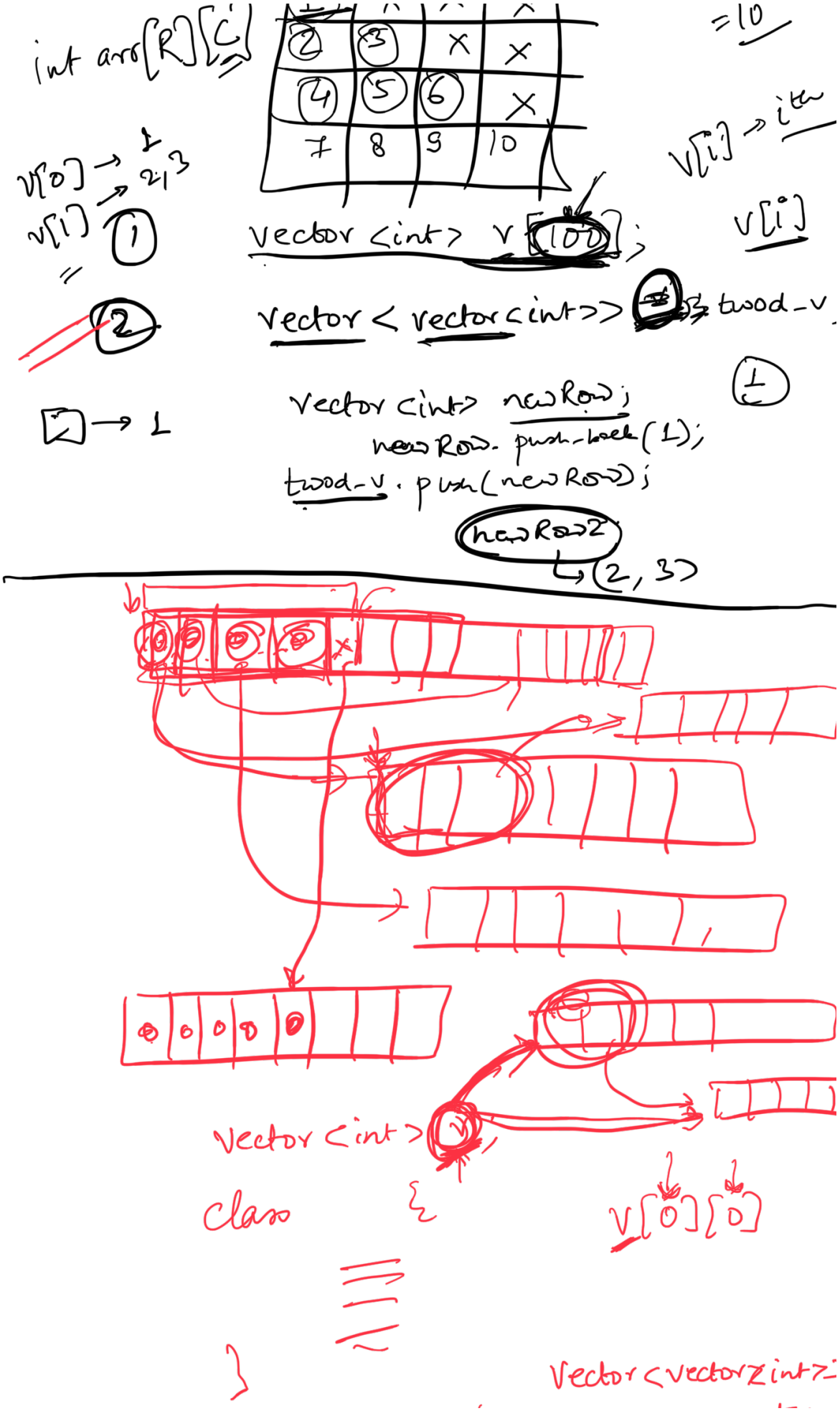
$10 \times 4 + 6$

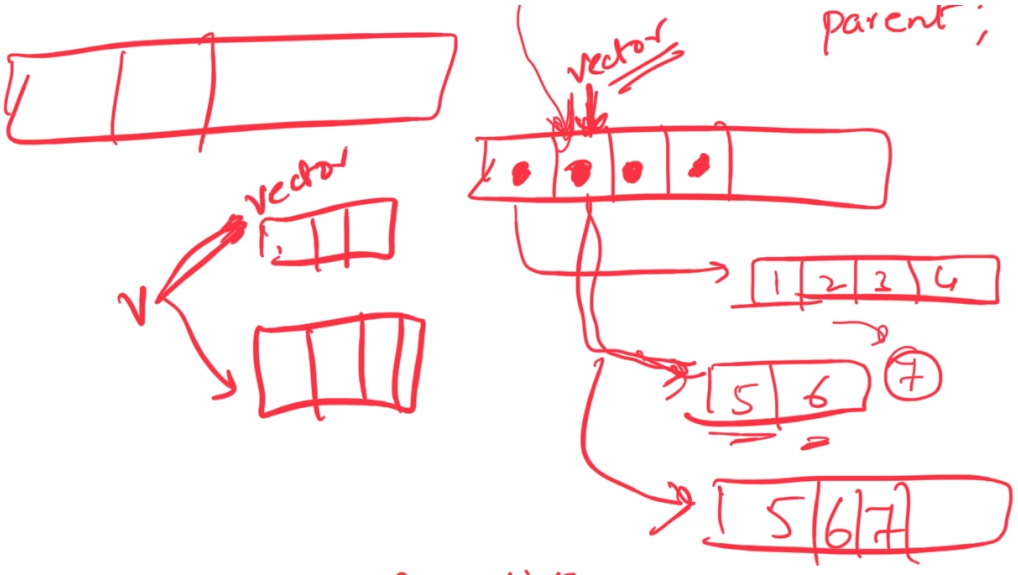
R C

arr[i][j] ← O(1)

(i × R + j + 1)

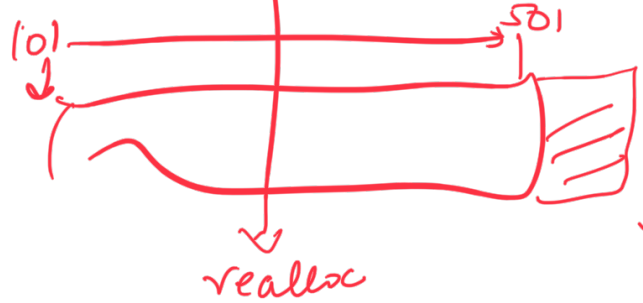
100000
ALLOWED


int arr[R][C]
=10
v[i] → item
v[i]
vector<int> v[100];
vector<vector<int>> twod_v
vector<int> newRow;
newRow.push_back(1);
twod_v.push(newRow);
newRow2
(2,3)
vector<int>
class
{
}
v[0][0]
vector<vector<int>>

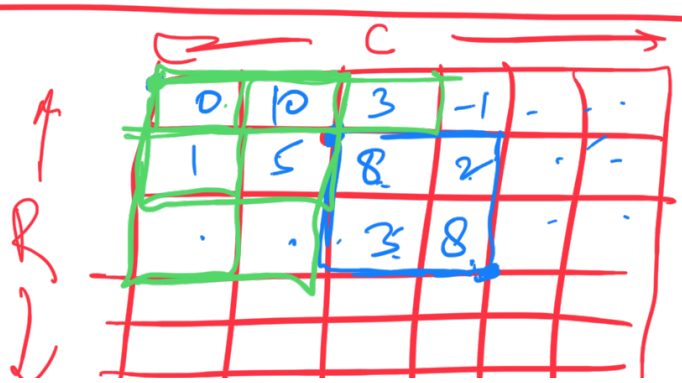


List → Arraylist
List → LinkedList

(int*) malloc(sizeof(int)*100)
400B

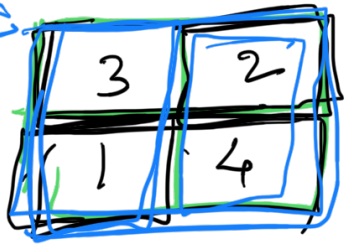


X+4



8+2+13+8
=21

R*C* Sum of matrix



3+2+1+4=

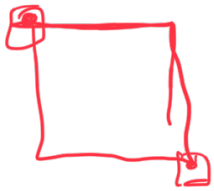
3+2=5
1+4=5
3+1=4
2+4=6
3+2+1+4=1

+0

4+4+4+4

4

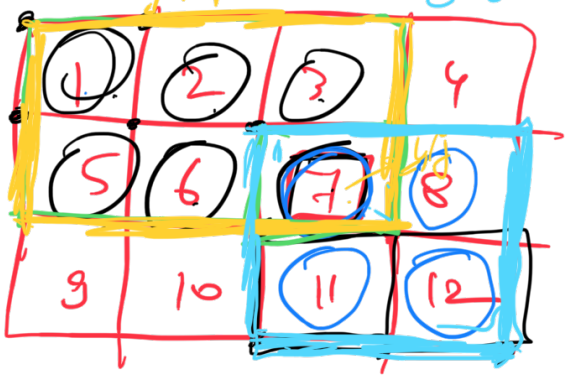
3



3

R

(R-i)



6x4

7x24

(i+1)(j+1) (R-i)(C-j)

O(N²)
(O(R*C))

A[i][j] * [(i+1)(j+1) * (R-i)(C-j)

for (int topr1 = 0; topr2 < R; topr2++

for(int topc1=0; topc1<c; top

for(bottomr1 = topr1;
bottom < R; —

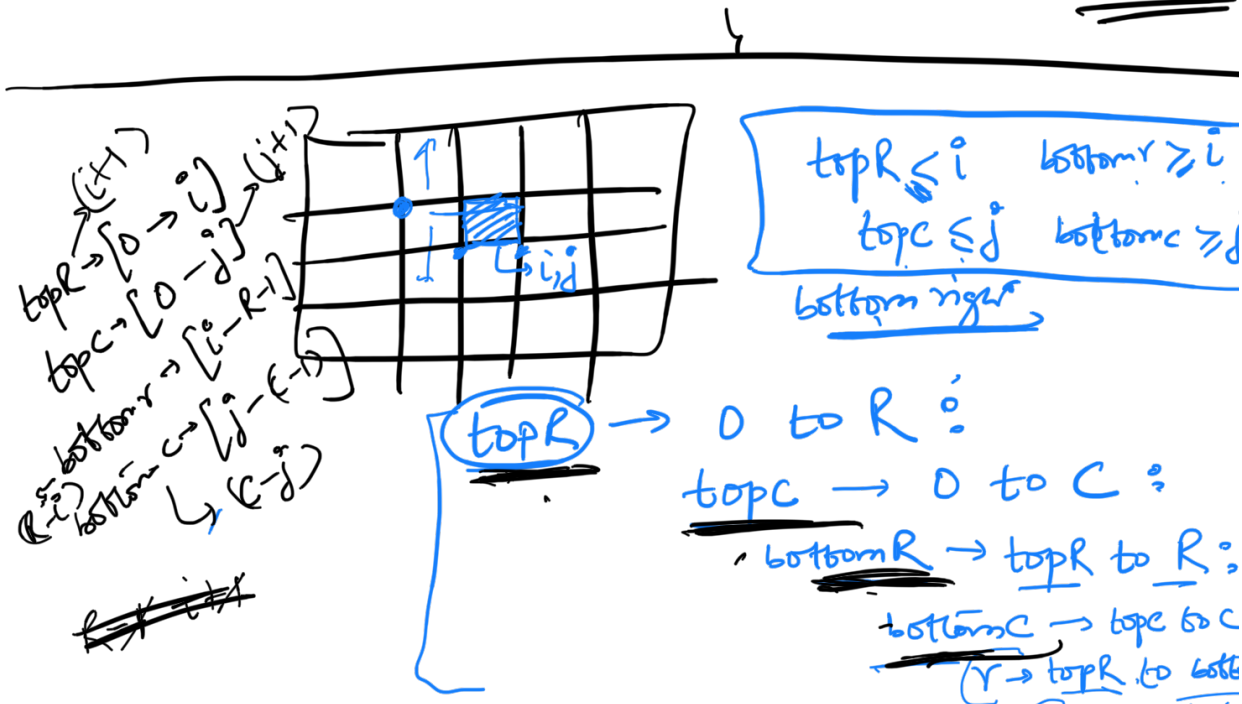
topr1, topc1, bottomr1, bottomc

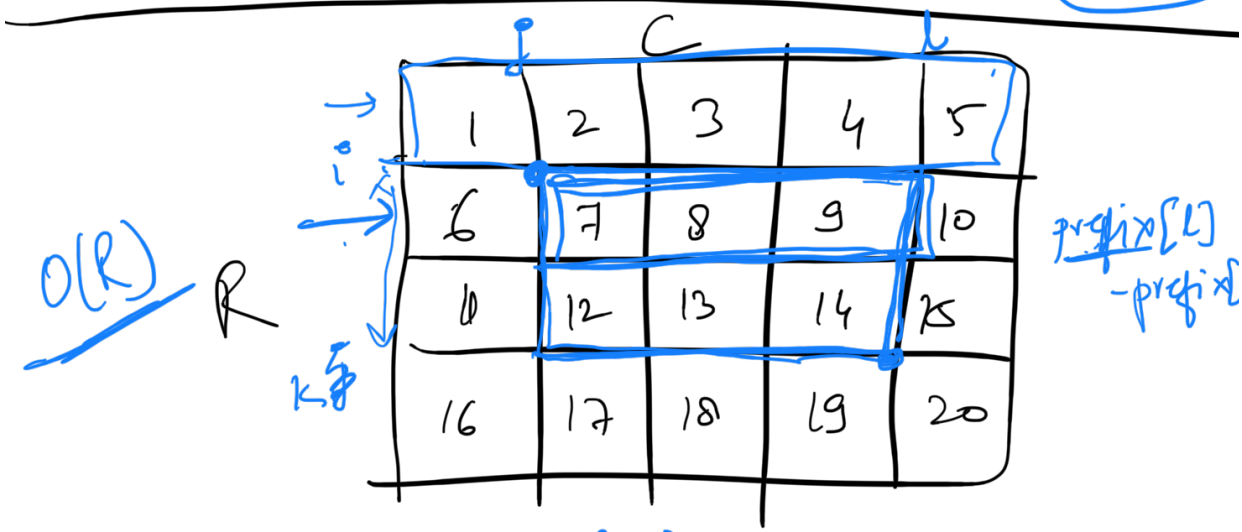
for(i=topr1; i< bottomr1; i++)

O(R³C³)

O(N⁶)

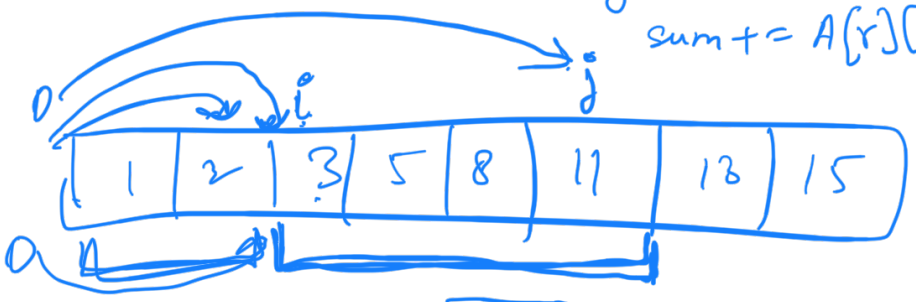
for (i = topC1 ; j < bott
sum += A[





c → j → 1 :

sum += A[r][c]



prefix sum

1 + 2 + 3 + 5 + 8 + 11

prefix[i] → 0 to i

i to j → prefix[j] − prefix[i−1]

prefix[0] = A[0];

prefix[1] = A[0] + A[1]

prefix[2] = A[0] + A[1] + A[2]

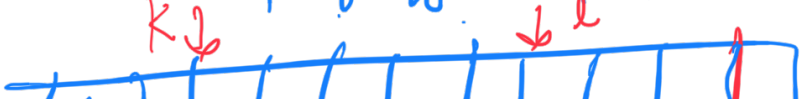
prefix[j] = prefix[j−1] + A[j];

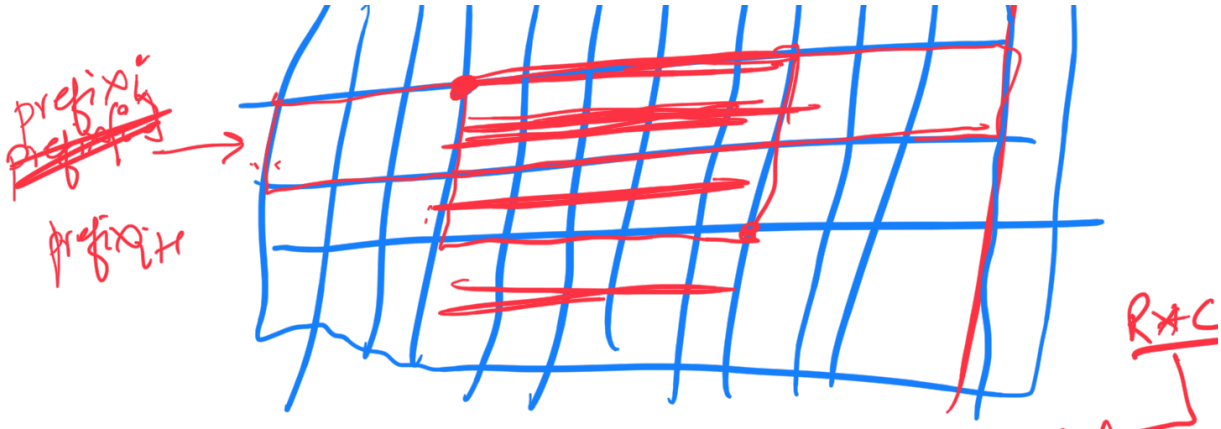
1-D

prefix[j] = A[0] + A[1] + A[2] + ... A[i] .... A[j]

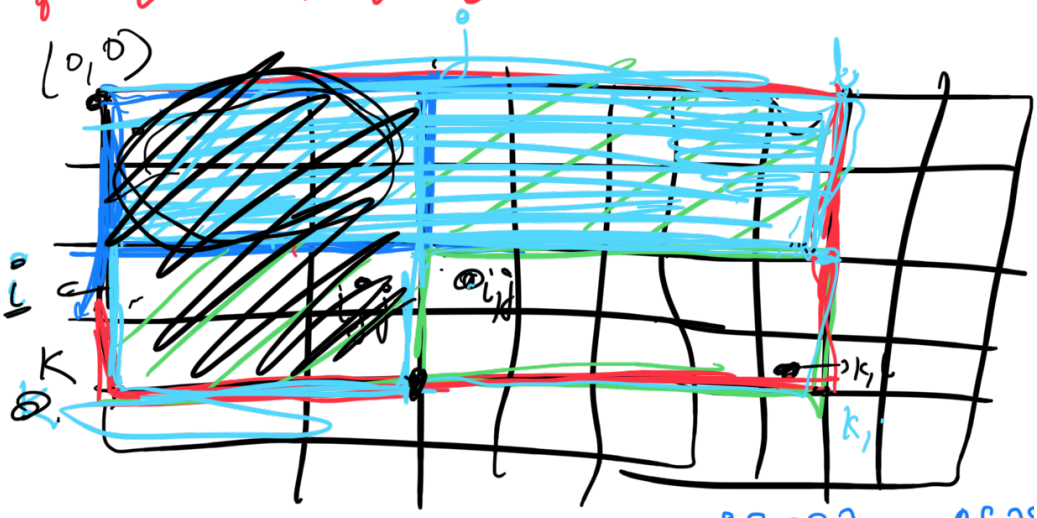
prefix[i−1] = A[0] .... A[i−1]

prefix[j] − prefix[i−1]





prefix_i[l] − prefix_i[k−1] O(R)



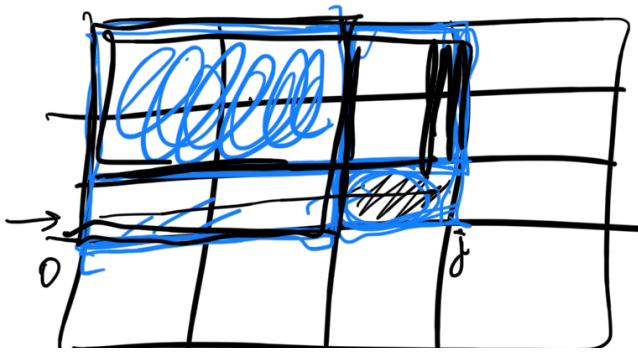
prefixsum[i][j] = A[0][0] + A[0][1] + ... + A[0][j]
+ A[1][0] + A[1][1] + ... + A[1][j]
+ ...
+ A[i][0] + A[i][1] ... A[i][j]

O(c)

(i, j) → (k, l)

prefixsum[i-1][j]

prefixsum[k][l] − prefixsum[i-1][l]
− prefixsum[k][j-1] +
prefixsum[i-1][j-1]



R×C
x(R×C)
O(R²C²)

prefixSum[i][j] = prefixSum[i][j-1]
+ prefixSum[i-1][j]
- prefixSum[i-1][j-1]
+ A[i][j]

1	2	3	4
5	6	7	8
9	10	11	12

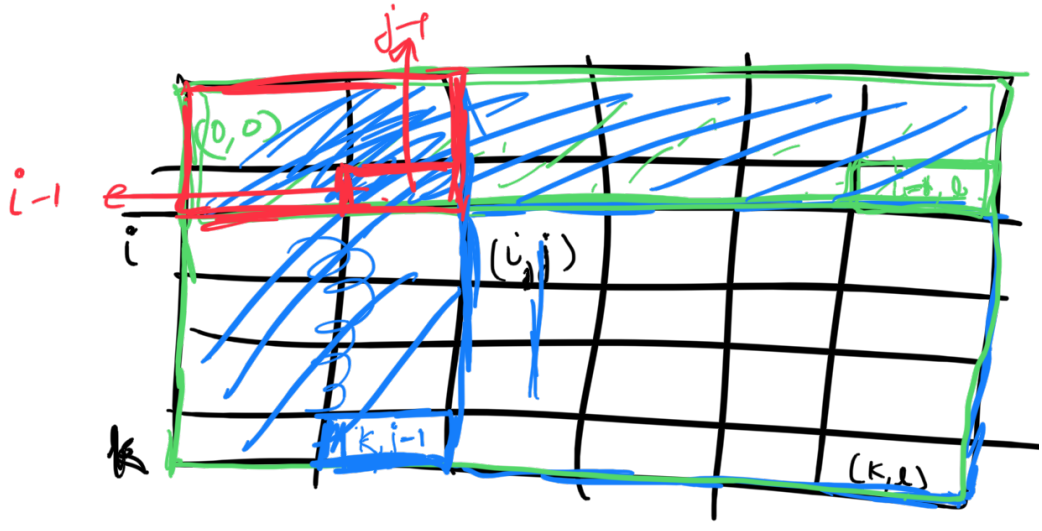
prefixSum(i)(j)

Q

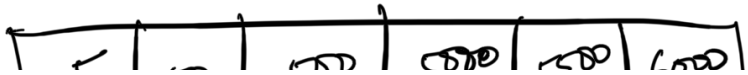
preprocessing time + Q * O(queries)

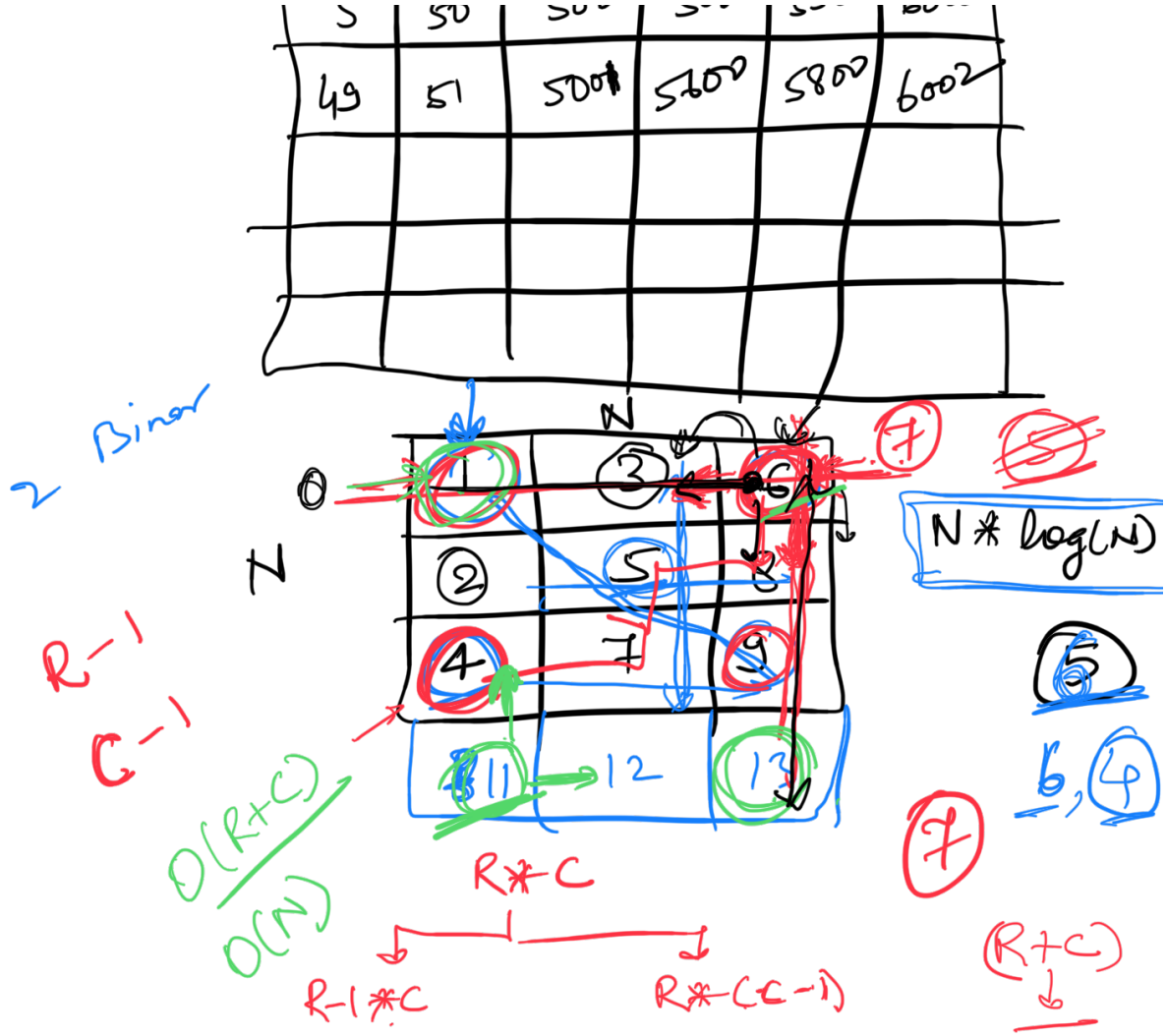
R*C*O(1) + Q * O(1)

O(R*C + Q)

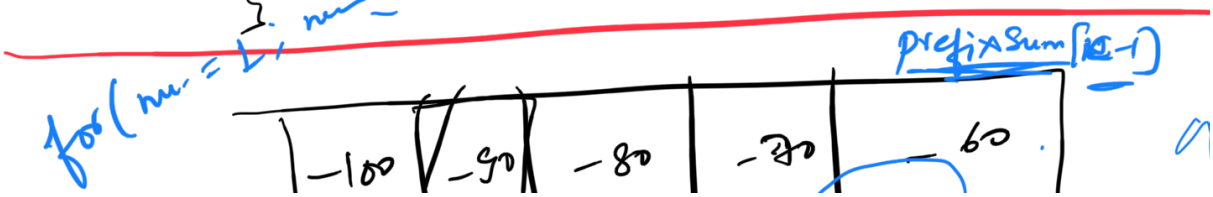


prefixSum(k,l) - prefixSum(i-1,l)
- prefixSum(k,j-1) + prefixSum(i-1,j-1)






```
int r= 0,  c = C-1;
while (r<R && c>=0) {
    if( key == A[r][c])
        return 1;
    if( key < A[r][c])
        c--;
    else
        r++;
}
return true;
```



Max Sum Submatrix

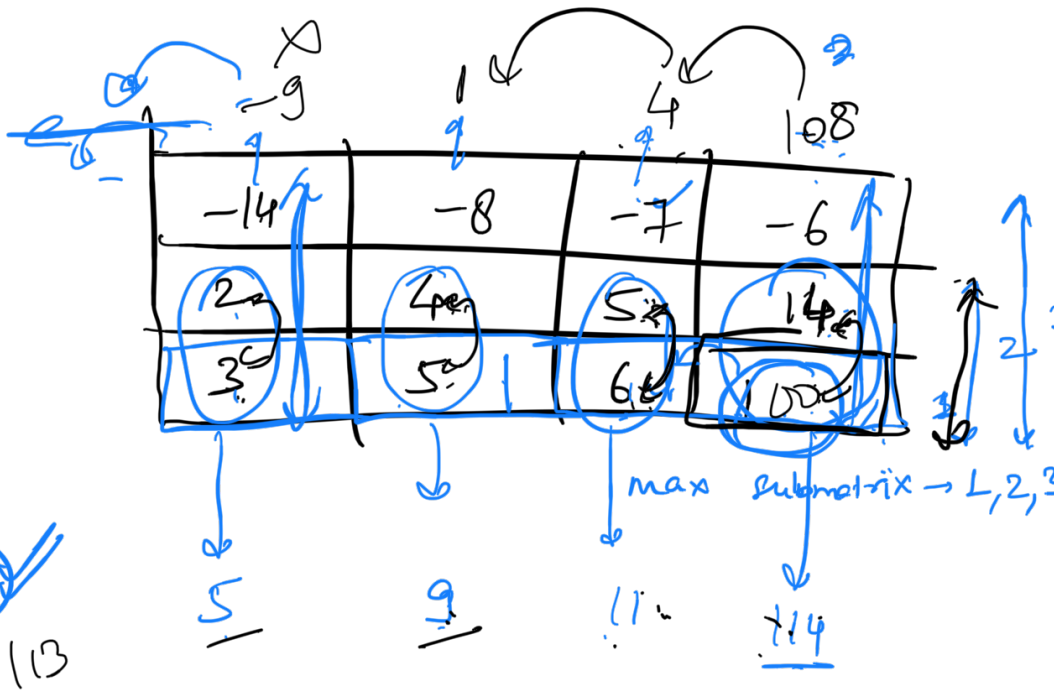
[(i,j) → ~~P[i][j]~~ (R-1, C-1)] → ~~Suffix sum~~

O(R*C)

(0,0) to (i,j)

Time : O(R*C)

space : O(R*C)



114

139

~~100~~ 113

-14	-12
-11	-5

-5