Problem - Solving - 5

Question : Maximum Submatrix Sum

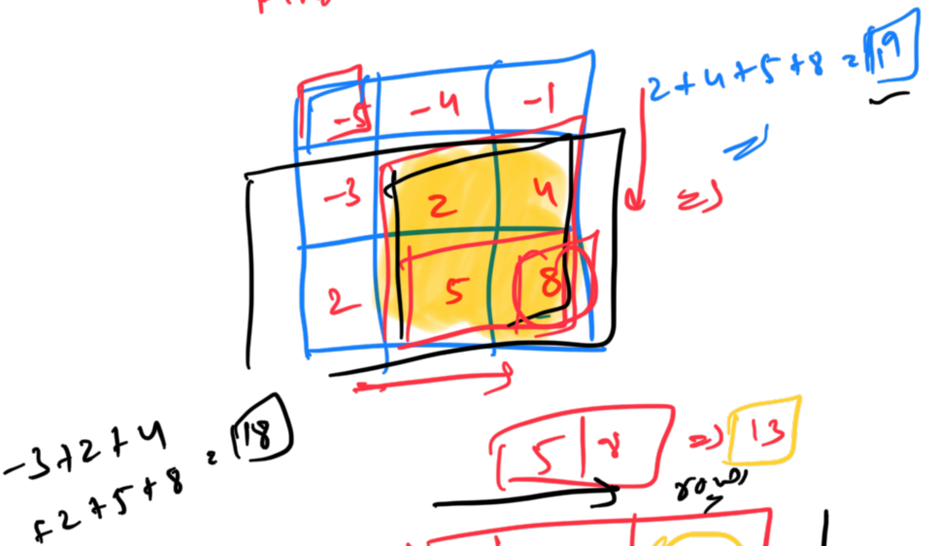
$(N \times M)$

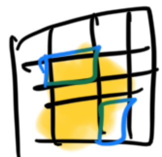
2D Matrix

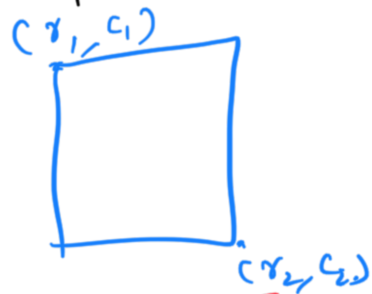
→ All rows are sorted

→ All colums are sorted

Find maximum sum submatrix







Strictly Increasing

Top left (r_1, c_1)

Bottom Right (r_2, c_2)

Brute Force :

Consider all submatrices

(r_1, c_1) (r_2, c_2)

2D Prefix Sum: $O(1)$

$\text{sum}([r_1, c_1] \to [r_2, c_2])$

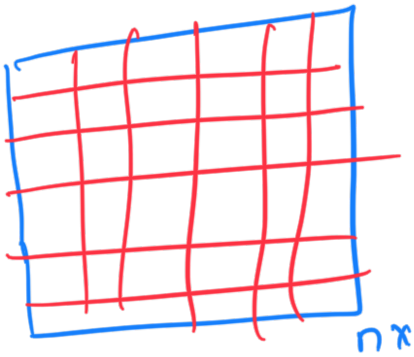
$pre[r_2][c_2] - pre[r_1-1][c_2] - pre[r_2][c_1-1] + pre[r_1-1][c_2-1]$

$O(1)$

N

$O(n^2)$

How many submatrices?



$n \times n$

(r_1, c_1)

(r_2, c_2)

$O(N^4)$

$\to (N^4)$

$O(1)$

$O(N \times N \times N \times N) = O(N^4)$

$r_1 \Rightarrow [0 \ldots N-1]$

$r_2 \Rightarrow [0 \ldots N-1]$

$c_1 \Rightarrow [0 \ldots N-1]$

$c_2 \Rightarrow [0 \ldots N-1]$

T.C: $O(N^4 \times 1) \Rightarrow \boxed{O(N^4)}$

ans = INT_MIN

3	4	5
7	10	11
9	13	15

Approach 2:

Observations

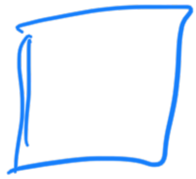
1) Bottom Right is the max element.

2) If all values are positive, sum of Entire Matrix.

3) If all values are negative,

3) Bottom Right Eleme…

$\boxed{\text{Last Element}}$ is always part of my answer



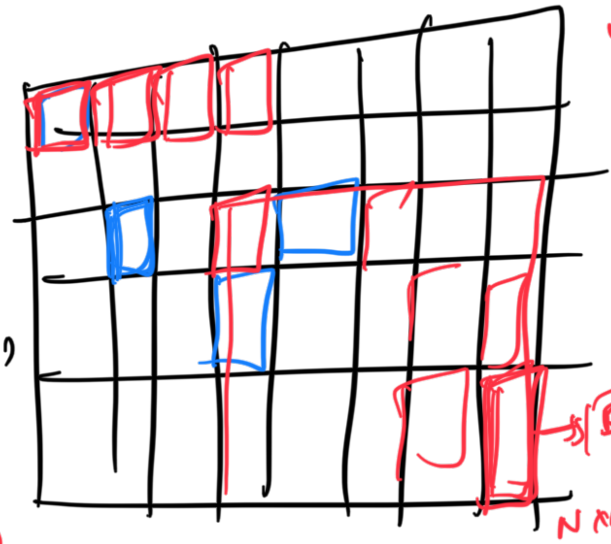
(r_1, c_1) : TL

$\boxed{(r_2, c_2) : BR}$

T2) BR

↓

$(n-1, m-1)$



$\boxed{\text{Brute Force}}$

B.R: $(n-1, m-1)$

$O(1)$

BR

$N \times M$

2D

$O(N \cdot M) \times O(1)$

$\Rightarrow O(N \cdot M)$

$\boxed{\text{T.C:}}$

T.C: $(N \cdot M) \times O(1)$

T.C: $\boxed{O(N \cdot M)}$

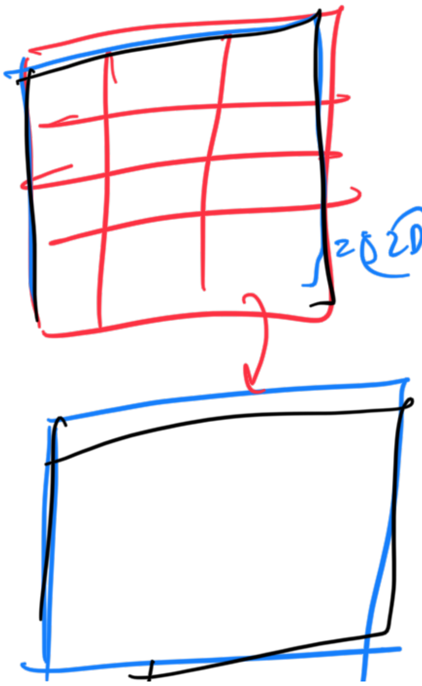
S.C: $O(N \cdot M)$

$O(N \cdot M) + O(N \cdot M) = \boxed{O(N \cdot M)}$

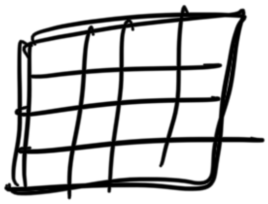
↓

Prefix Sum

mat



2D



(0,0)

$O(n \cdot m)$

$O(1)$

long long

(i, j)

for(i = 0 to n) {
  for(j = 0 to m) {
    TL : $\boxed{(i, j)}$

T.C : O(N·M) + O(N)
O(M)
O(N·M)

prefix

Question: Permutations of string A in string B

$\boxed{A}$ = 'abc'

B = 'abcbacabc'

count = ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5

$\boxed{5}$

abc : bac, cab, cba, acb, bca

6 Permutation

A = abcd

B = baccdbadcacbd

$\boxed{\text{count} = 3}$

n unique char

a b c d

4 3 2 1

4.3.2.1 = $\boxed{N!}$

Observations

1) Every permutation of a string will have same no of characters

2) For all chars, the frequency is exactly

→ (a b c a)

a b c d e

2	1	1	0	0	0	0	0	0	1	1

(26)

→ (b c a a)

a b c d e . .

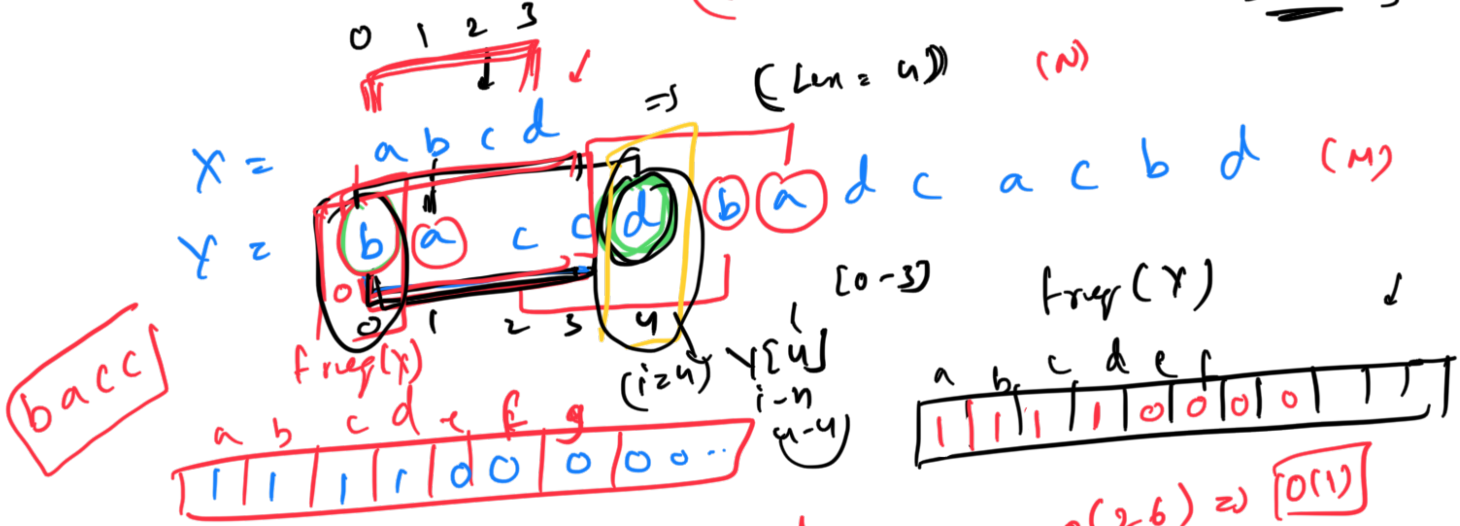
2	1	1	0	0	0	.

abcd
dbca
bcad

N = 4

(a - z) ⇒ 26 characters

Substring



o(26) ⇒ $\boxed{O(1)}$

a b c d

freq.

Count ++;
N = X.length()
M = Y.length

int freq1 [26] = {0};
int freq2 [26] = {0};

for(i = 0; i < n; i++) {
    freq1 [x[i]]++;
}

{(x[i] - 'a')}

```
freq2 [Y[i]]++; )
}
count=0;
if(freq1 == freq2) count++;        -> First sub.
   [length(X)]   [length(Y)]
for(i=n; i<m; i++) {
    freq2[Y[i]]++;
    freq2[Y[i-n]]--;

    if(freq1 == freq2)   count++;
                                   for(i=0;i<26;i++)
    }
}
return count;
```

freq1 (26)   freq2 (26)

for(i=0;i<26;i++){

}

'a' : 97 → 0

'b' : 98 → 1

'c' : 99 → '2'

d = 100

(100 − 97)

'a' → 0

'y' → 1

a+b+c+d = 97 98 99 100 ⇒ 394

b b b d → 394

97+98+98+100

String

abcd bbbd

Y = _ _ _ _ _ _ _ _ _ _ _ _ _ _

M iterations

T.C: O(M * 26)

T.C = $\boxed{O(M)}$

S.C: O(26) => O(1)

Question : Minimum Difference.

minimum (max - min)

A =	2	6	3	8	9	=>	(9 - 2) = 7
	3	6	3	8	9	=>	(6)
B = 1	3	6	3	8	8	=>	(5)
B = 2	3	6	3	8	8	=>	5
B = 3	4	6	3	8	8	=>	4
B = 4	4	6	4	8	8	(8 - 4) =>	4
B = 5	4	6	4	7	8 (7)	=>	$\boxed{3}$
$\boxed{B = 6}$ =	4	6	4	7	7		

(max - min)

B B = 6

B = 4	4	6	4	8	8
		7	4	8	8

→ B = 3 4 ⇒ 4

B = 6 4 8 4 8 8

$\boxed{B = 4}$

A = 3 3 3 3 3 ⇒ (0)

$\boxed{B}$ steps

$\boxed{\text{max} - \text{min}}$

B = 5: 3 3 3 3 3 ⇒ (1)

Observations

1) Answer depends on current max and min of the array

2) a_1 a_2 a_3 a_4 a_5 a_6 a_7 ..

3)

$\boxed{B = 6}$

A = 2 2 3 4 5 6 6 7

5   2 2 3 4 5 6 6 6

4   B = 5   2 2 3 4 5 6 6 6

B = 3   3 3 3 4 5 5 5 5

$\boxed{2}$ B = 0   3 3 3 4

(> B) break

O(n) max min

B = 1

freq(max) => f1
freq(min) => f2

larg = 7
small = 2
ans = 5

B = 7

A = 2 2 3 5 7 6

Freq[7] = 1
freq[2] = 2

B = 6

'3' : '3'

freq = {
'2' : 0
'3' : 3
'5' : 3
' ' : 0
'6' : 0
}

larg = ~~6~~ 5
small = 3

f(6) = 2
f(2) = 2

~~B = 4~~

B = 2

Freq(6) = 2
Freq(3) = 3

max = max - 1

freqfast

larg = 12
small = 1

1 1 4 7 10 13 12

(13-1)

$0 \le A[i] \le 10^6$

$A[i] \le 10^6$
Array

freq(1) = 2
Freq(13) = 1
Freq(12) = ∅ 1

freq(large)
freq(small)
larg
small

$10^8, 10^8+6$

12

largest = 12

Freq Array

map <int, int> freq;

larg = INT_MIN, small = INT_MAX;

```
for (i = 0; i < n; i++) {
    larg = max(larg, a[i]);
    small = min(small, a[i]);
    freq[a[i]]++;
}
```

O(n)

B

(B--)

```
while (B > 0 && small < largest) {
    if (freq[small] < freq[largest]) {
        if (B < freq[small]) break;
        freq[small+1] += freq[small];
        B = B - freq[small];
        small++;
    }
    else {
        if (B < freq[larg]) break;
        freq[larg-1] += freq[larg];
        B = B - freq[larg];
        largest--;
    }
}
return max-min;
```

ff max B times & min

max (larg-small) & min

T.C: $O(n) + O(\min(B, \text{largest}-\text{smallest}))$

$B = 10^5$

$a[i] = 10^6$

10^6

1 10^6

freq

1 :	1
10^6 :	1

$\Rightarrow \boxed{O(2)}$

Question: Count total set bits

N {1 to N}

N = 3

1 : 001 => 1

2 : 010 => 1

3 : 011 => 2

1 + 1 + 2 = $\boxed{4}$

N = 5 =>

1 : 001

2 : 010

3 : 011

4 : 100

5 : 101

1 + 1 + 2 + 1 + 2 = 7

Brute Force:

For every number in (1, N),

count set bits.

T.C : N × (No. of bits of that num)

(log n) no.b

7 : 111

1 1 1

3

7 % 2 = 1

N = 3

$(\log_2 n + 1)$

(N) (16)

$\log_2 16 = 4 + 1$

N = 1

N = 0

$\log^n$ bits

10000

$\log_2^{18} = \lfloor 4.x \rfloor + 1$

(18)

10010

T.C: O(N logN)

Approach 2:

Observations

1) We have 2 numbers x, y

$\boxed{y = \frac{x}{2}}$

2) (No of set bits in x) − (No of set bits in y) ≤ 1

x = 7 111 ⇒ 3 Δ = 1

y = 3 011 ⇒ 2

x = 12 1100 ⇒ 2 Δ = 0

y = 6 0110 ⇒ 2

$(y = \frac{x}{2})$

1 0 1

0 1 0

Case 1: x is odd

10011 → 01001

Last bit is always set.

$y = \frac{x}{2}$. y = x >> 1

Difference = 1

x

Right shift once = $\frac{}{2}$

$x = 11$ $\quad$ 1011 $\quad$ 0 1 0 1

$\frac{x}{2} = 5$ $\quad$ 0101

$$\boxed{\begin{array}{l} x = \boxed{x >> 1} \\ x = \frac{x}{2} \end{array}} \Big\} \; y \qquad \frac{x}{2^2}$$

<u>Case 2:</u> x is even

Last bit is always 0

$x = 110010$

$y = x >> 1 =$

Difference = 0

$\boxed{\begin{array}{l} n, y \\ y = \frac{x}{2} \end{array}}$

(2) $\quad$ (4) $\quad$ (5)

10 $\quad$ bits(2) $\quad$ bits(2) + 1

No. of set bits = 1

```
if( N is even ){
    bits(N) =        bits(N/2)
}
else {               bits(N/2) + 1
    bits(N) =
}
```

F(25) $\quad$ F(24)

F(12)

(24) $\quad$ 25

$F(12)$ $\qquad$ $F(12)+1$

Answering bigger problems using smaller subproblems.

0	1	2	3	4	5	6	7	8	...	N
0	1	1	2	1	2	2	3			

bits =

Base Cases:

count = 0

```
bits[N+1] = {0}
bits[0] = 0;
bits[1] = 1;
for (x = 2; x ≤ N; x++) {
    bits[x] = bits[x/2] + x % 2;
    count += bits[x];
}
```

odd 1, even 0

bits[2/2]

bits(1)

bits(1)+1

Memoization => Dynamic Programming

$\boxed{N = 10^9}$

$N = 8 = 2^3$

8 numbers

0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0

$N = 2^x$

$(4 + 4 + 4)$

$\left(\frac{2^x}{2}\right) \times x$

$\boxed{2^{x-1} \times x}$ — ①

(0 to $2^x - 1$)

$N = 10^9$

$\log(10^9)$

$9 \log 10$

$= 9 \times 3$

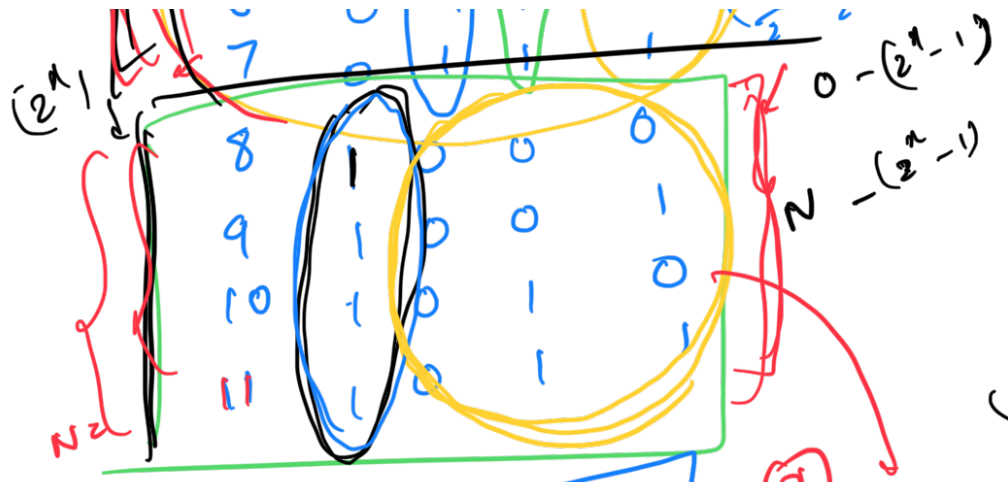
$= \boxed{27}$

$O(1)$

$N = 2^x$

closest power of 2 < ...

8, $\frac{8}{2}$, $\frac{8}{2}$



$0 - (2^x - 1)$

$N - (2^x - 1)$

$\boxed{N = 2^x + t}$

$11 = 2^3 + 3$

$(x \cdot 2^{x-1})$

0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1

setBits(1 to 3)

$N = 11$

$2^x = 8$

$11 - (8-1)$

$\boxed{N - (2^x - 1)}$ — 2

$11 - (8-1)$

$11 - 7 = 4$

$N - (2^x - 1)$

```
int countSetBits(N)
    if(N == 0 || N == 1) return N;

    x = floor(log2(N));
    ans = 0;
    ans += x * 2^(x-1);          — 1
    ans += N - (2^x - 1)         — 2
    ans += countSetBits(N - (2^x - 1))
}
```

$\boxed{\log 31 = 5}$

Ex $N = 31$ 11111

F(31)   1111

↓

F(15)   1111

31 - 16.

15 - 8)

$\log_2 N$

$F(7)$ 111 $\quad 7-4$

$F(3)$ 11 $\quad 3-2$

$N = (2^x)$

$F(1)$ 1 $\quad O(1)$

T.C: $O(\log_2 N)$

$N = 10^9$

$\log_2 10^9$

$= 9 \log_2 10 \approx \boxed{27}$