Agenda

- Max Subsequence Sum
- Unique Paths in a Grid I
- Unique Paths in a Grid II
- Dungeons and Princess

subsequence / subset $\rightarrow 2^N$

Given an arr[], find max subsequence sum.

	0	1	2	3	4	5	ans
arr[] →	2	-4	5	3	-8	1	11

pick all the positive elements.

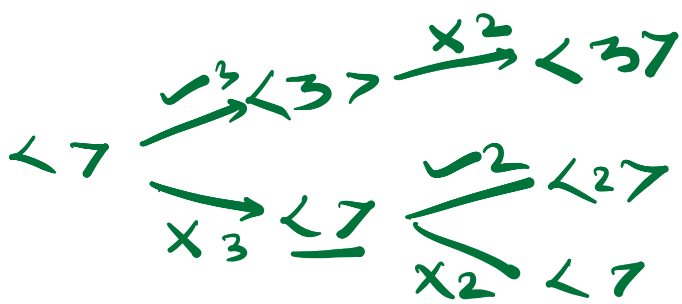
1. Find max subsequence sum from a given array, where selecting adjacent elements is not allowed. (+ve integers)

[9 14 3] ans 14

[13 4 2] ans 15

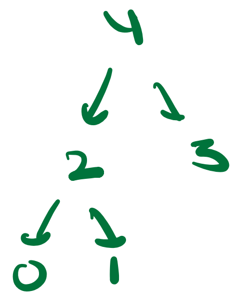
[9 4 13 24] 33

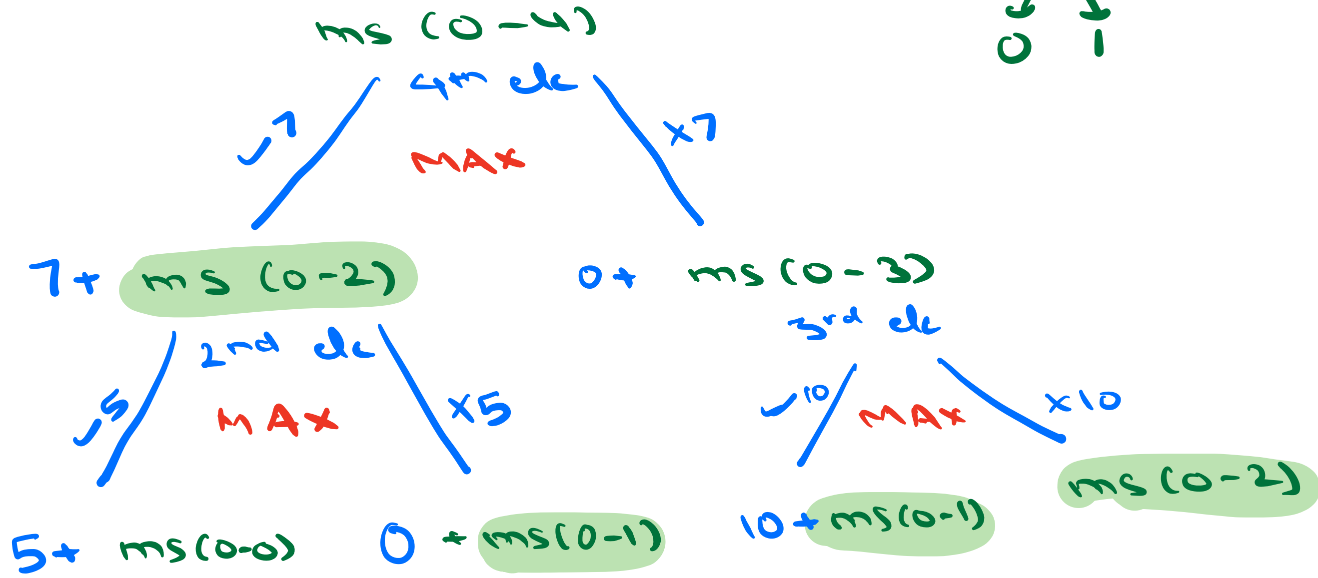
	0	1	2	3	4
ar →	3	2	5	10	7



0 1 2 3 4

ar → 3 2 5 10 7






```
int dp [N] = {-1}

int maxsum (int [] ar, int e){
    if (e==0) return ar[0]
    if (e < 0) return 0
    if (dp [e] != -1)
        return dp [e]

    include = ar[e] + maxsum (ar, e-2)
    exclude = 0 + maxsum (ar, e-1)

    dp[e] = max (include, exclude)
    return dp [e]
}
```

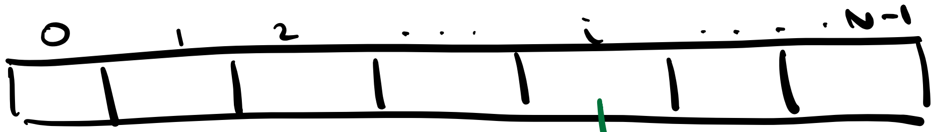
TC : O(N)

SC : O(N)

end index

dp [i] → Max sub sum from 0 → i

dp[N-1] = max sub seq from 0 → N-1



ms (0-i)

✓ith → ar[i] + ms(0, i-2)

✗ ith → 0 + ms(0, i-1)

ms (0,1)

✓1 → ar[1] + ms(0,-1)

✗1 → ms(0,0)

---

dp[i]   ms (0-i)

dp[i-1]   dp[i-2]

ar[] →

	0	1	2	3	4
ar[] →	3	2	5	10	7

dp [5]

	0-0	0-1			
idx	0	1	2	3	4
val	~~-1~~ 3	~~-1~~ 3	~~-1~~ 8	~~-1~~ 13	~~-1~~ 15

ar[0]   max(ar[0], ar[1])

ms (0-2)

✓ 5 + ms(0-0) dp[0] 8

x ms(0-1) dp[1] 3

ms (0-3)

✓ 10 + ms(0,1) dp[1] 13

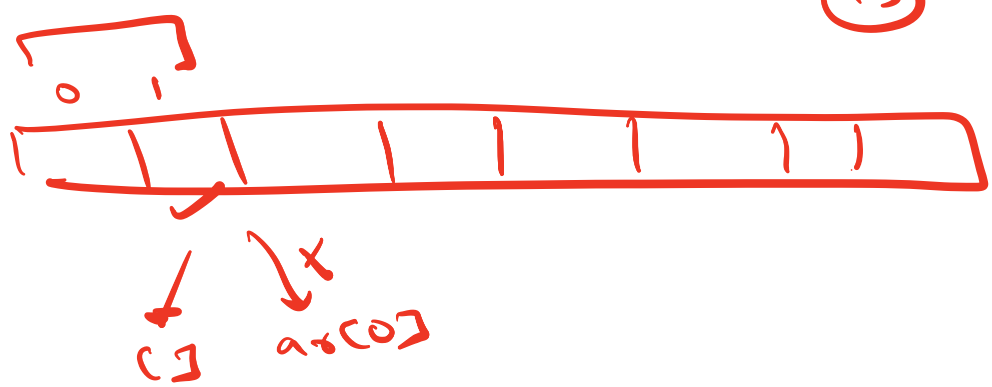
x ms[0,2) dp[2] 8

ms (0,4)

✓ 7 + ms(0,2) 8 15

x ms(0,3) 13

ar :




```
int dp[N] = <-1>
    dp[0] = ar[0]
    dp[1] = max (ar[0], ar[1])
for (i = 2 ; i < N ; i++){
    dp[i] = max(ar[i] + dp[i-2], dp[i-1])
}
return dp[N-1]
```

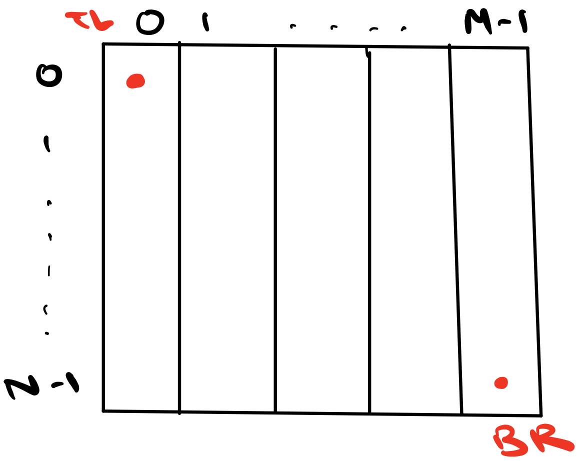
TC: O(N)

SC: O(N)

↓

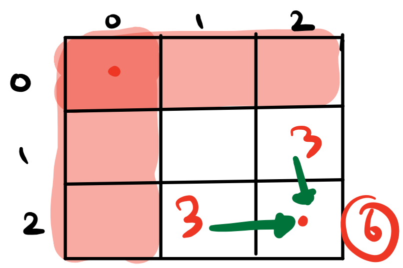
O(1)

2. Given mat[N][M], find total no. of ways from (0,0) to (N-1, M-1). We can take 1 step Down (D) or Right (R) at a time.

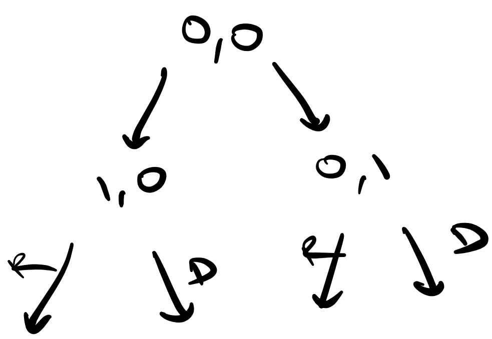


mat[3][3]

ans: 6



RRDD
DDRR
RDRD
DRDR
RDDR
DRRD

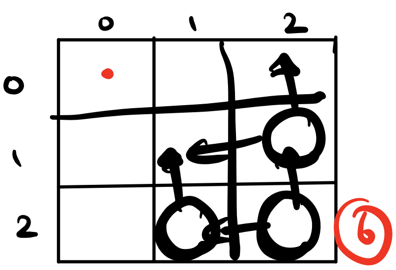


ways(2,2)

ways(N-1, M-1)

ways(2,1) + ways(1,2)

ways(2,1)	ways(1,2)
DDR R	RRD D
RDD R	DRR D
DRD R	RDR D



ways(2,2)

ways(2,1) + ways(1,2)

ways(2,0) + ways(1,1) ways(1,1) + ways(0,2)

```
int dp[N][M] = {-1}
```

2 , 2

N-1 M-1

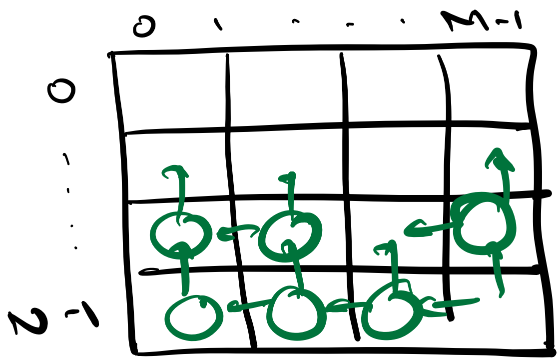
TC: O(N*M)

SC: O(N*M)

```
int ways (int i, int j) {
    if (i==0 || j==0)
        return 1
    if (dp[i][j] != -1)
        return dp[i][j]
    dp[i][j] = ways(i, j-1) + ways(i-1, j)
    return dp[i][j]
}
```

Stack → N+M-1

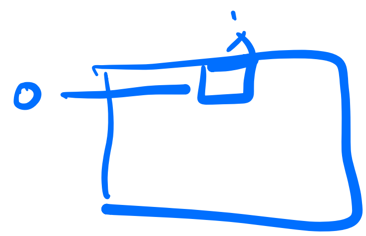
DP[] → N*M



Mat[3][3]

dp

	0	1	2
0	1	1	1
1	1	2	3
2	1	3	6

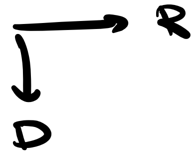


```
int dp[N][M] = {-1}

for (i=0 ; i<N ; i++) {
    for (j=0 ; j<M ; j++) {
        if (i==0 && j==0)
            dp[i][j]=1

        else if (i==0)
            dp[i][j]=1 //
                dp[i][j-1]

        else if (j==0)
            dp[i][j]==1 //
                dp[i-1][j]

        else
            dp[i][j] = dp[i-1][j]+
                       dp[i][j-1]
    }
}
```

TC : O(N*M)

SC : O(N*M)

3. Given mat[N][M], find total no. of ways from (0,0) to (N-1, M-1). Cell with value 1 and 0 represents non-blocked and blocked cell respectively.



	0	1	2	3
0	1	1	1	1
1	1	0	$\boxed{1}$	0
2	0	1	1	1
3	1	0	1	1
4	1	1	1	1

ways (1,2)

ways (1,1) + ways (0,2)

0

mat[1][1] = 0

→ wall

if (mat[i][j] = = 0)

ways (i, j) = 0

else

ways (i, j) = ways (i, j-1) + ways (i-1, j)

11:00 PM

① If mat[0][0] = 0

ways (0,0) = 0

Not possible to reach any cell

②

dp[0][j] → dp[0][j-1]

dp[i][0] → dp[i-1][0]

4.

M

	0	1	2	3
0	-3	2	4	-5
1	-6	5	-4	6
2	-15	-7	-5	1 -2
3	2	10	8 -3	5 -4 Q

+ve → Red bull

-ve → Dragon

Energy <= 0 => Die

$\rightarrow$ R

$\downarrow$ D

Min energy to start with ?

5

$\boxed{-4}$

$x + (-4) = \underline{1}$

$x = 1 + 4 = 5$

x

$\boxed{-3} \rightarrow \boxed{5}$

$x + (-3) = 5$

$x = 5 + 3 = 8$

$\boxed{-2}$

$\boxed{5}$

$x + (-2) = 5$

$x = 5 + 2 = 7$

x

-5	7
8	5

$\boxed{+9}$ 7

8

$x + (-5) = \min(7, 8)$

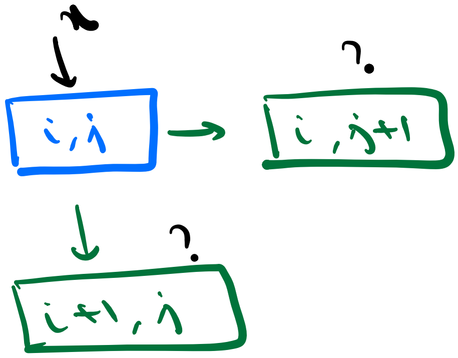
$x = 7 - (-5) = 12$

M	0	1	2	3
0	-3	2	4	-5
1	-6	5	-4	6
2	-15	-7	-5	7 -2
3	2	10	8 -3	5 -4 0

+ve → Red bull

-ve → Dragon

Energy <= 0 => Die

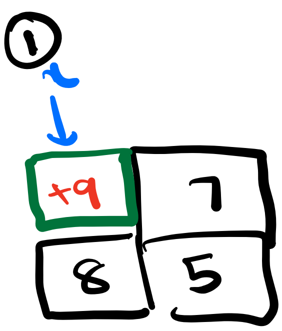


Minenergy (i, j)

x + mat[i][j] = min(Minenergy(i, j+1),
Min energy(i+1, j))

x = max(1, min(Minenergy(i, j+1),
Min energy(i+1, j)) - mat[i][j])

Minenergy (i, j)



$x + 9 = \min(7, 8) = 7$

$x = 7 - 9 = -2$

$x = \max(1, -2) = 1$

$\boxed{x} + a[N-1][M-1] = 1$

$x = \max(1, \ 1 - a[N-1][M-1])$

M	0	1	2	3
0	-3	2	4	-5
1	-6	5	-4	6
2	-15	-7	-5	7 -2
3	2	10	8 -3	5 -4 0 x

+ve → Red Bull

-ve → Dragon

Energy <= 0 => Die

TC: O(N*M)

SC: O(N*M)

```
int dp[N][M] = <-1>

for (i = N-1 ; i >= 0 ; i--) {
    for (j = M-1 ; j >= 0 ; j-- {
        if (i == N-1 && j == M-1)
        dp[i][j] = max(1, 1 - a[i][j])

        else if (i == N-1)
        dp[i][j] = max(1, dp[i][j+1]
                        - mat[i][j])
```

```
else if (j == m-1)
    dp[i][j] = max(1, dp[i+1][j]
                      - mat[i][j])
else
    dp[i][j] =
    max(1, min(dp[i][j+1], dp[i+1][j])
                      - mat[i][j])
}

return dp[0][0]
```

M

	0	1	2	3
0	4 → -3	1 2	1 4	6 -5
1	7 -6	1 5	5 -4	1 6
2	16 -15	8 -7	12 -5	7 -2
3	1 2	1 10	8 -3	5 -4

+ve → Red Bull

-ve → Dragon

Energy <= 0 => Die

$x - 3 = \min(1, 7) = 1$

$x = 1 + 3 = 4$

$x + 10 = 8$

$x = -2$