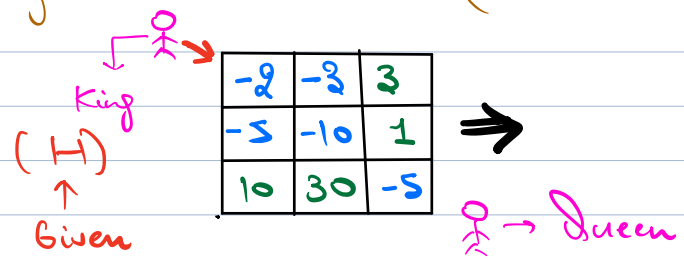
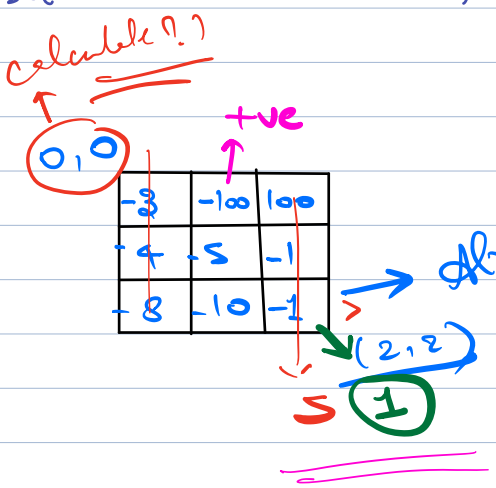
Dungeon Princess (2D Grid Problem)

Google Amazon



If at any cell $H <= 0 \Rightarrow$ Dead

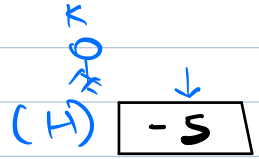
Return the min amount of health with which the king should start so that he is able to reach the queen w/o dying.



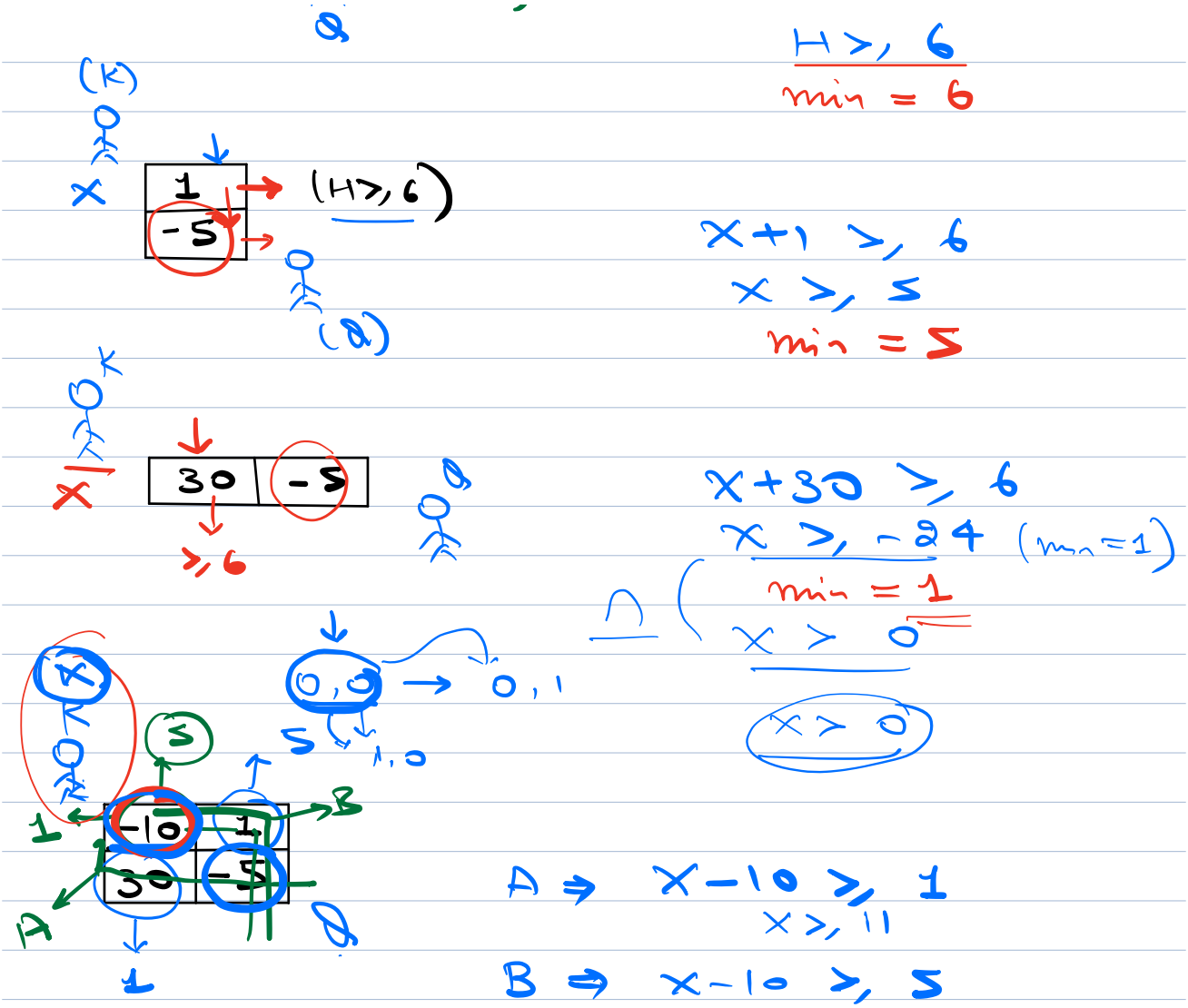
Sol^n Recursion

1) EOC → Down or Right

2) What does the state represent.



(H >= 1) H - 5 >= 1



DP[i][j] = min amount of health required before starting my journey for (i, j)

3) Recurrance Relation

$X + M[i][j] = \min(DP[i+1][j], DP[i][j+1])$

$X = DP[i][j] = \min(DP[i+1][j], DP[i][j+1]) - M[i][j]$

$X > 0$; $X \leq 0 \Rightarrow 1$

H.W: Code $\Rightarrow$ T.C. ??

DP[N-1][M-1] $\Rightarrow$

LCS (Longest Common Subsequence)

Google
Meta
Amazon
Walmart

Given 2 string A & B

find the length of LCS b/w A & B.

A: AGGTAB — AT, AGGT, AAB

B: GXTNAYB → GT, GXTA...

AB, GTA, GT
TB, GTAB .....
↳ (4) ~~Ans~~

1) Brute force

1) Generate all subsequences of A & B and compare them.

$A^{(N)}$ & $B^{(M)}$

$(2^N) \times (2^M) = 2^{N+M}$

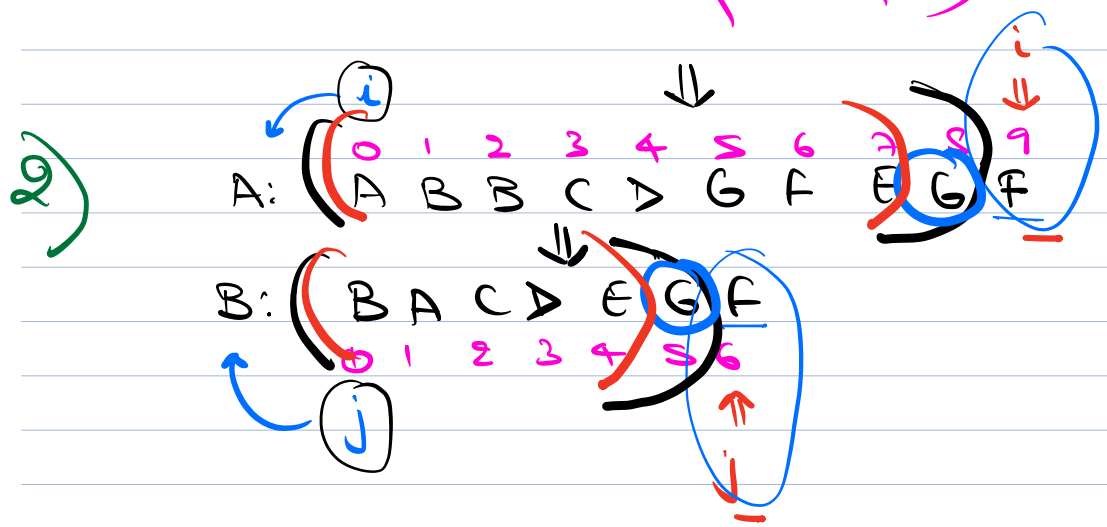
2) Generate all subsequence of A & check if it is also a subsequence of B.

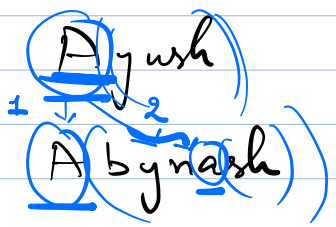
A → $(2^N) \times O(M)$

ADG

B: BACDEGF

T.C. = $O(M \times 2^N)$





1 → Aysh

2 → Ash

LLCS = 1 + 1 +

A → 0, i

B → 0, j

⇒ Variables

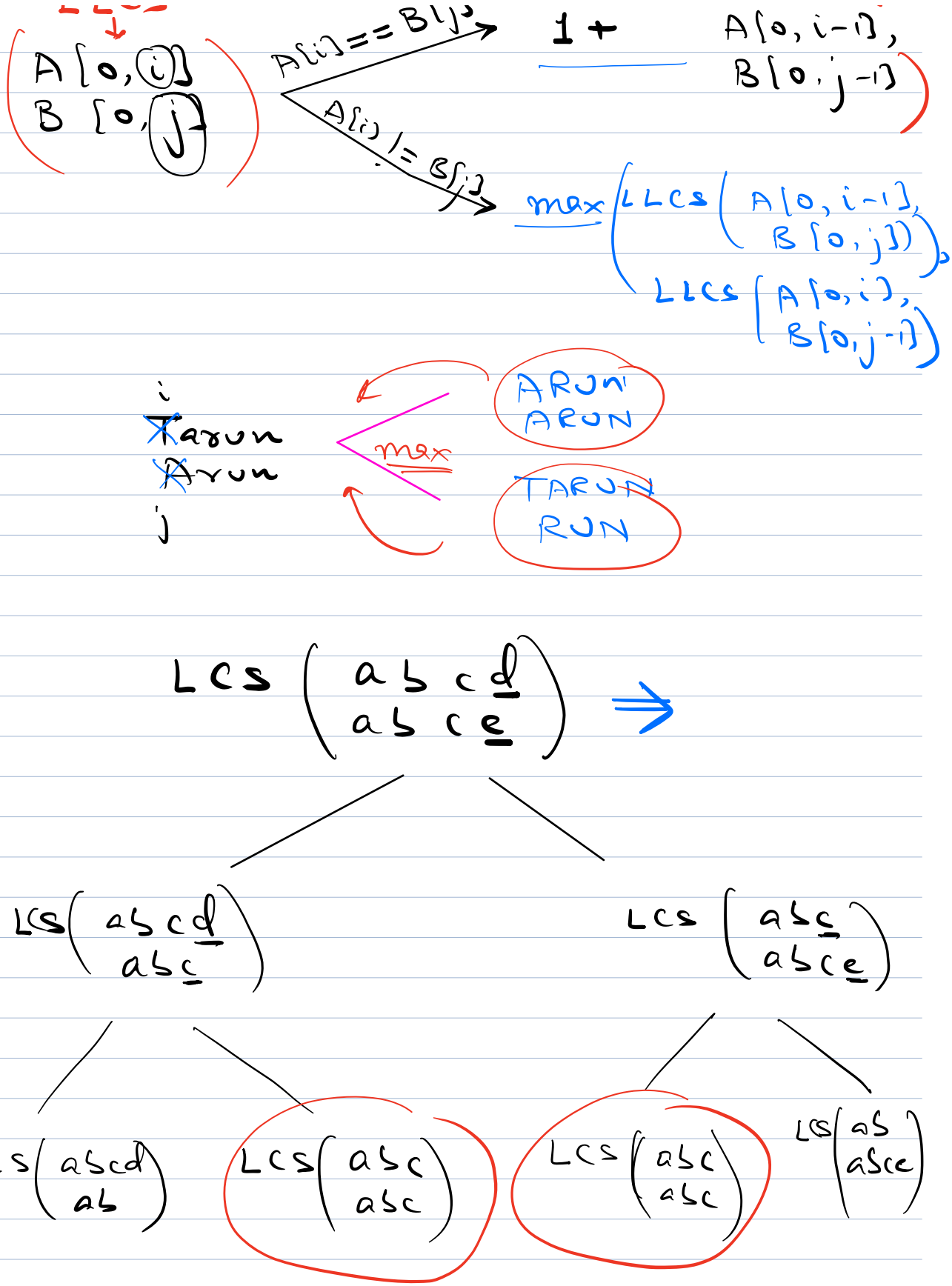
Constants

(i, j) ⇒ Length of LCS b/w

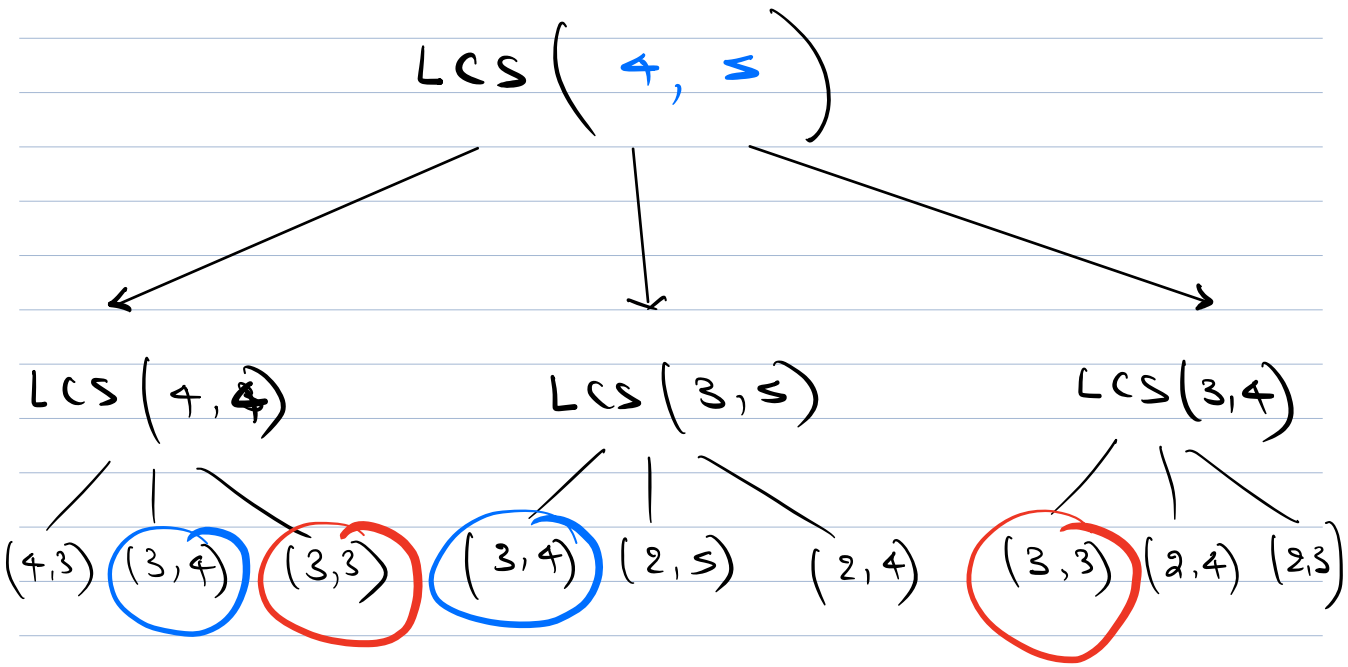
A[0, i]

B[0, j]

LLCS (

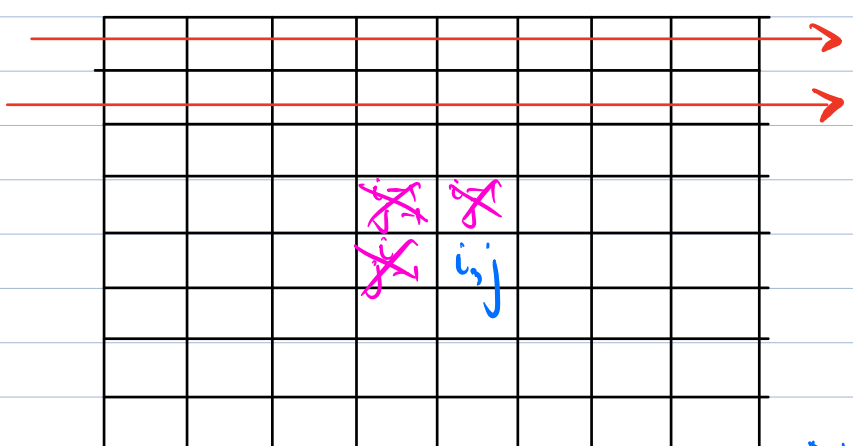

A[0,i]
B[0,j]
A[i]==B[j]
1 +
A[0,i-1],
B[0,j-1]
A[i] != B[j]
max
LLCS(A[0,i-1], B[0,j])
LLCS(A[0,i], B[0,j-1])
i
Tarun
Arun
j
max
ARUN
ARUN
TARUN
RUN
LCS (a b c d / a b c e)
LCS (a b c d / a b c)
LCS (a b c / a b c e)
LCS (a b c d / a b)
LCS (a b c / a b c)
LCS (a b c / a b c)
LCS (a b / a b c e)

$N = 5, \quad M = 6$



DP[i][j] = Length of LCS of
A[0, i]
B[0, j]

2
i ⇒ N
j ⇒ M
N × M



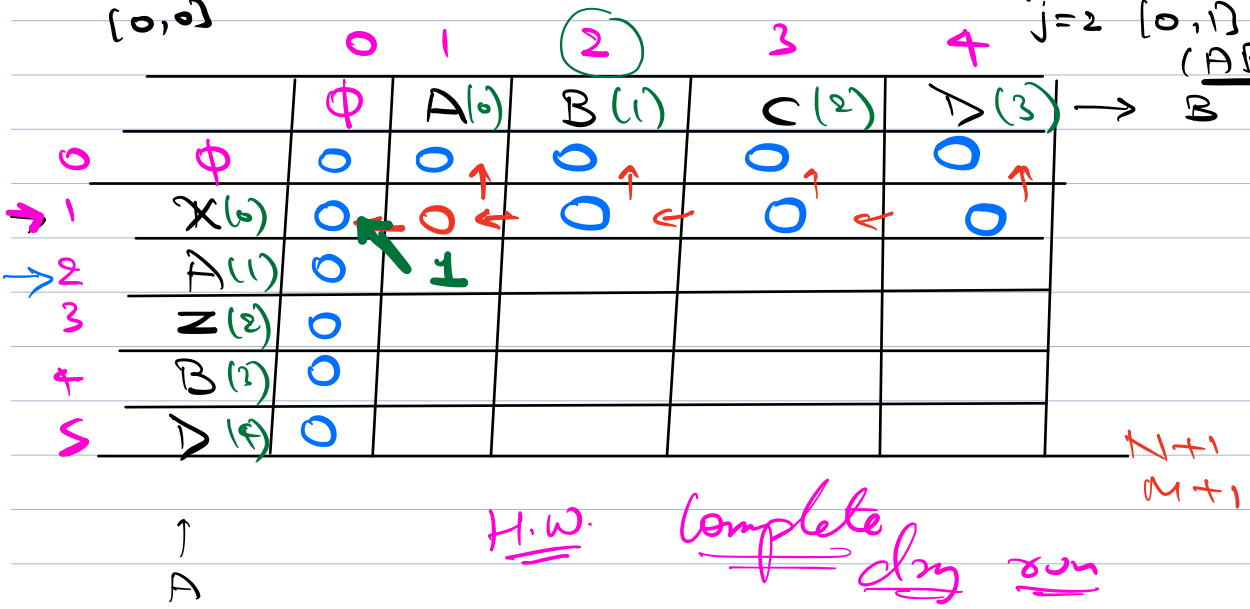
A: XA
B: j : [1, 4]
N × M

i = 2 ...

[0,0]

j=1 [0,0] (A)

j=2 [0,1] (AB)



H.W. Complete dry run

DP[i][j] ⇒ A[0, i-1]
B[0, j-1]

Code

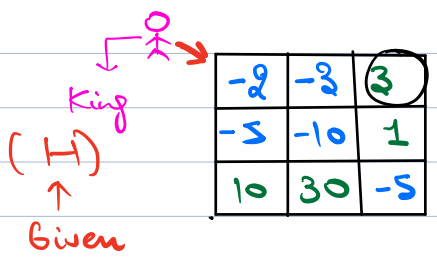
```
DP[N+1][M+1]
for (i=0; i<=N; i++){
    DP[i][0] = 0;
}
for (j=0; j<=M; j++){
    DP[0][j] = 0
}
```

```
for (i = 1; i <= N; i++) {
    for (j = 1; j <= M; j++) {
        if (A[i-1] == B[j-1]) {
            DP[i][j] = 1 + DP[i-1][j-1];
        }
        else {
            DP[i][j] = max(DP[i-1][j], DP[i][j-1])
        }
    }
}
```

T.C. = $O(N \times M)$

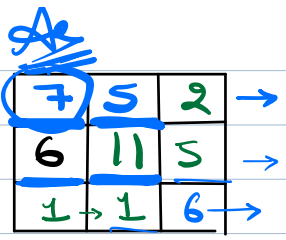
S.C. = $O(N \times M)$ $\Rightarrow$ $O(M)$

Doubt



$x - 2 \geq 5 \Rightarrow x \geq 7$

$x - 5 \geq 1 \Rightarrow x \geq 6$



$x - 10 \geq 1$

$x \geq 11$

$x + 30 \geq 6$

$x \geq -24$

$x + 10 \geq 1$

$x \geq -9$

$x + 3 \geq 5$

$x \geq 2$

$x - 5 \geq 1$

$x \geq 6$