COMPETITIVE PROGRAMMING

Assignment-07

2303A51463

B-07

Dynamic Programming (DP)

Problem 1: Treasure Island with One Teleport

Algorithm:

1. Read n and `m`.
2. Read treasure grid `a[n][m]`.
3. Create DP table `dp0[n][m]` (no teleport used).
4. Initialize `dp0[0][0] = a[0][0]`.
5. Fill `dp0` using right and down moves.
6. Precompute:
    a. `rowmax[i][j]` = max of `dp0[i][0..j]`
    b. `colmax[i][j]` = max of `dp0[0..i][j]`
7. Create DP table `dp1[n][m]` (teleport used).
8. For each cell `(i,j)`:
    a. Teleport from best cell in same row/column.
    b. Add treasure `a[i][j]`.
    c. Continue DP using right/down.
9. Print `dp1[n-1][m-1]`.

Code:

ass7,1.py > ...

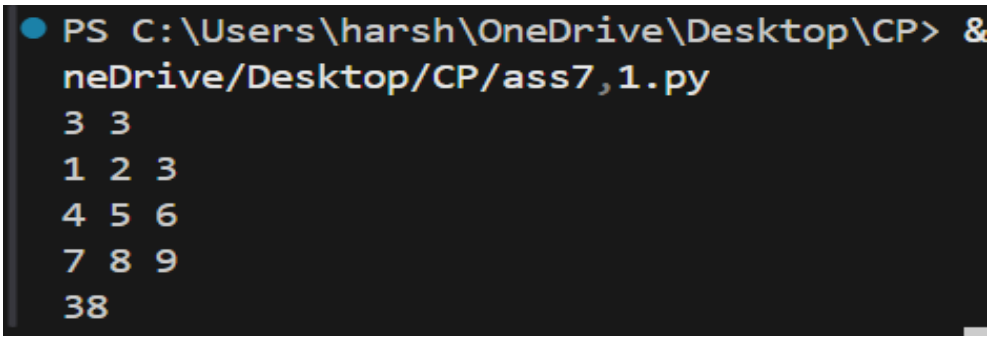
```python
n,m=map(int,input().split())
a=[list(map(int,input().split())) for _ in range(n)]
NEG=-10**18
dp0=[[NEG]*m for _ in range(n)]
dp1=[[NEG]*m for _ in range(n)]
dp0[0][0]=a[0][0]

for i in range(n):
    for j in range(m):
        if i>0:
            dp0[i][j]=max(dp0[i][j],dp0[i-1][j]+a[i][j])
        if j>0:
            dp0[i][j]=max(dp0[i][j],dp0[i][j-1]+a[i][j])
rowmax=[[NEG]*m for _ in range(n)]
colmax=[[NEG]*m for _ in range(n)]

for i in range(n):
    rowmax[i][0]=dp0[i][0]
    for j in range(1,m):
        rowmax[i][j]=max(rowmax[i][j-1],dp0[i][j])
for j in range(m):
    colmax[0][j]=dp0[0][j]
    for i in range(1,n):
        colmax[i][j]=max(colmax[i-1][j],dp0[i][j])
for i in range(n):
    for j in range(m):
        dp1[i][j]=max(rowmax[i][j],colmax[i][j])+a[i][j]
        if i>0:
            dp1[i][j]=max(dp1[i][j],dp1[i-1][j]+a[i][j])
        if j>0:
            dp1[i][j]=max(dp1[i][j],dp1[i][j-1]+a[i][j])

print(dp1[n-1][m-1])
```

Output:



Problem 2: Palindromic Subsequence with Weighted Indices

Algorithm:

1. Read integer n and string `s`.
2. Create DP table `dp[l][r]` = maximum index sum palindrome in substring `l..r`.
3. Initialize `dp[i][i] = i+1` (1-based index value).
4. If `s[l] == s[r]`, add `(l+1)+(r+1)` to `dp[l+1][r-1]`.
5. Else, take maximum of excluding left or right character.
6. Fill DP table bottom-up.
7. Output `dp[0][n-1]`.

ass7.2.py > ...

```python
n=int(input())
s=input()

dp=[[0]*n for _ in range(n)]

for i in range(n):
    dp[i][i]=i+1

for length in range(2,n+1):
    for l in range(n-length+1):
        r=l+length-1
        if s[l]==s[r]:
            if l+1<=r-1:
                dp[l][r]=dp[l+1][r-1]+(l+1)+(r+1)
            else:
                dp[l][r]=(l+1)+(r+1)
        else:
            dp[l][r]=max(dp[l+1][r],dp[l][r-1])

print(dp[0][n-1])

```

Output:

```
PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/
neDrive/Desktop/CP/ass7.2.py
4
abca
8
```