# **DYNAMIC PROGRAMMING:-**

# 1) Playing with Numbers:

Ram and Sita are playing with numbers by giving puzzles to each other. Now it was Ram term, so he gave Sita a positive integer 'n' and two numbers 1 and 3. He asked her to find the possible ways by which the number n can be represented using 1 and 3. Write any efficient algorithm to find the possible ways.

# Example 1:

```
Input: 6
Output:6
```

**Explanation:** There are 6 ways to 6 represent number with 1 and 3

```
1+1+1+1+1+1
3+3
1+1+1+3
1+1+3+1
1+3+1+1
3+1+1+1
```

# **Input Format**

First Line contains the number n

# **Output Format**

Print: The number of possible ways 'n' can be represented using 1 and 3

Sample Input

6

Sample Output

6

```
Answer: (penalty regime: 0 %)
```

	Input	Expected	Got	
~	6	6	6	~
~	25	8641	8641	<b>~</b>
~	100	24382819596721629	24382819596721629	*

Passed all tests! 🗸

# 2) Playing with Chessboard:

Ram is given with an n\*n chessboard with each cell with a monetary value. Ram stands at the (0,0), that the position of the top left white rook. He is been given a task to reach the bottom right black rook position (n-1, n-1) constrained that he needs to reach the position by traveling the maximum monetary path under the condition that he can only travel one step right or one step down the board. Help ram to achieve it by providing an efficient DP algorithm.

#### Example:

#### Input

3

**1**24

**2** 3 4

871

#### **Output:**

19

## **Explanation:**

Totally there will be 6 paths among that the optimal is Optimal path value:1+2+8+7+1=19

#### **Input Format**

First Line contains the integer n

The next n lines contain the n\*n chessboard values

## **Output Format**

Print Maximum monetary value of the path

```
Answer: (penalty regime: 0 %)

#include <stdio.h>
define MAX 100

int main() {

int n;

int board[MAX][MAX];

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

scanf("%d", %board[i][j]);}}

dp[0][0] = board[0][0];

for (int j = 1; j < n; j++) {

dp[0][j] = dp[0][j-1] + board[0][j];}

for (int i = 1; i < n; i++) {

dp[i][0] = dp[i-1][0] + board[i][0];}

for (int i = 1; i < n; i++) {

dp[i][j] = board[i][j] + (dp[i-1][j] > dp[i]-1];}

printf("%d\n", dp[n-1][n-1]);

return 0;

1

#include <stdio.h>
#define MAX 100

int main() {

int main() {

int n;

int board[MAX][MAX];

scanf("%d", %board[i][j]);}

dp[0][0] = board[i][j];

dp[i][j] = board[i][j];

dp[i][j] = dp[i-1][j];

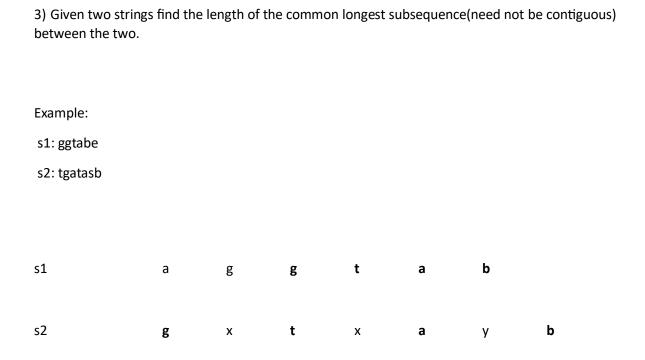
printf("%d\n", dp[n-1][n-1]);

return 0;
```

	Input	Expected	Got	
~	3	19	19	~
	1 2 4			
	2 3 4			
	8 7 1			
~	3	12	12	~
	1 3 1			
	151			
	4 2 1			
~	4	28	28	~
	1 1 3 4			
	1 5 7 8			
	2 3 4 6			
	1690			

Passed all tests! 🗸

Correct



# The length is 4

Solveing it using Dynamic Programming

# For example:

Input	Result
aab	2
azb	

```
Answer: (penalty regime: 0 %)
```

```
printf("%d\n",a[11][12]);
return 0;
```

Input	Expected	Got	
✓ aab azb	2	2	<b>~</b>
✓ ABCD ABCD	4	4	~

# 4) Problem statement:

Find the length of the Longest Non-decreasing Subsequence in a given Sequence.

Eg:

Input:9

Sequence:[-1,3,4,5,2,2,2,2,3]

the subsequence is [-1,2,2,2,2,3]

Output:6

```
Answer: (penalty regime: 0 %)

#include <stdio.h>
#define MAX 100

int main() {

int s[MAX];

int a[MAX];

for (int i=0;i<n;i++) {scanf("%d",&s[i]);}

int m= 1;

for (int i = 0; i < n; i++) {

a[i] = 1;

for (int j = 0; j < i; j++) {

if (s[j] <= s[i]) {a[i] = a[i]>a[j]+1?a[i]:a[j]+1;}

if (a[i] > m) {m = a[i];}

printf("%d\n", m);

return 0;

}
```

Г	Input	Expected	Got	
~	9 -1 3 4 5 2 2 2 2 3	6	6	~
~	7 1 2 2 4 5 7 6	6	6	~
Passe	ed all tests! 🗸			