Name	Aravinthaa S
Register No	230701032
Dept	CSE A

DYNAMIC PROGRAMMING

Ex No – 4.1	PLAYING WITH NUMBERS

Question:

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

Aim:

To find the number of possible ways a given number n can be represented using only 1 and 3, using an efficient algorithm.

Program:

```
#include <stdio.h>


long long countWays(int n) {
    long long dp[n + 1];


    dp[0] = 1;
    dp[1] = 1;
    dp[2] = 1;
    dp[3] = 2;


    for (int i = 4; i <= n; i++) {
```

```
        dp[i] = dp[i - 1] + dp[i - 3];
    }

    return dp[n];
}

int main() {
    int n;
    scanf("%d", &n);

    long long result = countWays(n);
    printf("%lld\n", result);

    return 0;
}
```

Input and Output:

	Input	Expected	Got	
✔	6	6	6	✔
✔	25	8641	8641	✔
✔	100	24382819596721629	24382819596721629	✔

Passed all tests! ✔

Ex No – 4.2	PLAYING WITH CHESSBOARD

Question:

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 positior (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 b providing an efficient DP algorithm.

Example:
Input
3
1 2 4
2 3 4
8 7 1
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

Aim:

To find the maximum monetary path from the top-left to the bottom-right of a chessboard using dynamic programming, where movement is allowed only to the right or down.

Program:

```
#include <stdio.h>


int main() {
   int n;
   scanf("%d", &n);
   int chessboard[n][n], d[n][n];


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


   d[0][0] = chessboard[0][0];
   for (int j = 1; j < n; j++) {
      d[0][j] = d[0][j - 1] + chessboard[0][j];
```

```
    }
    for (int i = 1; i < n; i++) {
        d[i][0] = d[i - 1][0] + chessboard[i][0];
    }
    for (int i = 1; i < n; i++) {
        for (int j = 1; j < n; j++) {
            int max_prev = d[i - 1][j] > d[i][j - 1] ? d[i - 1][j] : d[i][j - 1];
            d[i][j] = max_prev + chessboard[i][j];
        }
    }

    printf("%d\n", d[n - 1][n - 1]);
    return 0;
}
```

Input and Output:

	Input	Expected	Got	
✔	3 1 2 4 2 3 4 8 7 1	19	19	✔
✔	3 1 3 1 1 5 1 4 2 1	12	12	✔
✔	4 1 1 3 4 1 5 7 8 2 3 4 6 1 6 9 0	28	28	✔

Passed all tests! ✔

Ex No – 4.3	LONGEST COMMON SUBSEQUENCE

Question:

Given two strings find the length of the common longest subsequence(need not be contiguous) between the two.

Example:

s1: ggtabe

s2: tgatasb

s1	a	g	g	t	a	b	
s2	g	x	t	x	a	y	b

The length is 4

Solveing it using Dynamic Programming

For example:

Input	Result
aab azb	2

Aim:

To find the length of the longest common subsequence (LCS) between two strings using dynamic programming.

Program:

```
#include <stdio.h>

#include <string.h>


int Length(char s1[], char s2[]) {

   int m = strlen(s1);

   int n = strlen(s2);

   int dp[101][101] = {0};


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

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

```
            if (s1[i - 1] == s2[j - 1]) {
                dp[i][j] = dp[i - 1][j - 1] + 1;
            } else {
                dp[i][j] = dp[i - 1][j];
                if (dp[i][j - 1] > dp[i][j]) {
                    dp[i][j] = dp[i][j - 1];
                }
            }
        }
    }

    return dp[m][n];
}

int main() {
    char s1[101], s2[101];
    scanf("%s", s1);
    scanf("%s", s2);
    printf("%d\n", Length(s1, s2));
    return 0;
}
```

Input and Output:

	Input	Expected	Got	
✔	aab azb	2	2	✔
✔	ABCD ABCD	4	4	✔

Passed all tests! ✔

Ex No – 4.4	LONGEST NON - DECREASING SUBSEQUENCE

Question:

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

Aim:

To find the length of the Longest Non-decreasing Subsequence in a given sequence using an efficient algorithm.

Program:

```
#include <stdio.h>


int lseq(int arr[], int n) {
   int dp[100];
   int maxLength = 1;


   for (int i = 0; i < n; i++) {
      dp[i] = 1;
      for (int j = 0; j < i; j++) {
         if (arr[j] <= arr[i]) {
            dp[i] = dp[i] > dp[j] + 1 ? dp[i] : dp[j] + 1;
         }
      }
      if (dp[i] > maxLength) {
         maxLength = dp[i];
      }
   }
```

```
    return maxLength;
}

int main() {
    int n;
    scanf("%d", &n);
    int arr[100];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("%d\n", lseq(arr, n));
    return 0;
}
```

Input and Output:

	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	✔

Passed all tests! ✔