DAA MOODLE PROGRAMS
DYNAMIC PROGRAMS

230701221
M.Nithyashree
CSE-D

1.

AIM-

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.

CODE-

```c
#include <stdio.h>
long long NoOfWays(int n);

int main()
{
    int n;
    scanf("%d", &n);
    printf("%lld\n", NoOfWays(n));
    return 0;
}

long long NoOfWays(int n)
{
    if (n == 0)
    {
        return 1;
    }

    long long arr[n + 1];
    arr[0] = 1;
    arr[1] = 1;
    arr[2] = 1;
    arr[3] = 2;


    for (int i = 4; i <= n; i++)
    {
        arr[i] = arr[i - 1] + arr[i - 3];
    }

    return arr[n];
}

```

INPUT-

First Line contains the number n

OUTPUT-

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

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

Passed all tests! ✔

Correct

Marks for this submission: 10.00/10.00.

2.

AIM-

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.

CODE-

```c
#include <stdio.h>
int main() {
    int n;
    scanf("%d", &n);

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

    for (int j = 1; j < n; j++) {
        dp[0][j] = dp[0][j - 1] + arr[0][j];
    }

    for (int i = 1; i < n; i++) {
        dp[i][0] = dp[i - 1][0] + arr[i][0];
    }

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

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


}
```

INPUT-

First Line contains the integer n
The next n lines contain the n*n chessboard values

OUTPUT-

Print Maximum monetary value of the path

	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! ✔

Correct

Marks for this submission: 10.00/10.00.

3.

AIM-

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

Example:

s1: ggtabe

s2: tgatasb

CODE-

```c
#include <stdio.h>
#include <string.h>

int longestcommonsubsequence(char s1[], char s2[]) {
    int m = strlen(s1);
    int n = strlen(s2);
    int dp[m + 1][n + 1];
    for(int i = 0; i <= m; i++)
    {
        for(int j = 0; j <= n; j++)
        {
            if (i == 0 || j == 0)
            {
                dp[i][j] = 0;
            }
            else if(s1[i - 1] == s2[j - 1])
            {
                dp[i][j] = dp[i - 1][j - 1] + 1;
            }
            else
            {
                dp[i][j] = (dp[i - 1][j] > dp[i][j - 1]) ? dp[i - 1][j] : dp[i][j - 1];
            }
        }
    }
    return dp[m][n];
}
int main()
{
    char s1[100], s2[100];
    scanf("%s", s1);
    scanf("%s", s2);
    int result = longestcommonsubsequence(s1, s2);
    printf("%d\n", result);
}
```

OUTPUT-

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

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

4.

AIM-

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

CODE-

```c
#include <stdio.h>

int longestnondecsubsequence(int arr[], int n)
{
    int dp[n];
    int maxLength = 1;
    for (int i = 0; i < n; i++)
    {
        dp[i] = 1;
    }
    for (int i = 1; i < n; i++)
    {
        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[n];
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }

    int result = longestnondecsubsequence(arr, n);
    printf("%d\n", result);

    return 0;
}
```

INPUT-

9

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

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

OUTPUT-

6

	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! ✔

Correct

Marks for this submission: 1.00/1.00.