RAJALAKSHMI ENGINEERING COLLEGE
(Autonomous)

RAJALAKSHMI NAGAR, THANDALAM, CHENNAI-602105

RAJALAKSHMI ENGINEERING COLLEGE

CS23331 – DESIGN AND ANALYSIS OF ALGORITHMS

LABORATORY RECORD NOTEBOOK

Register Number	:	241801197
Name of the Student	:	C. Parvendhan
Year / Semester	:	II / III
Branch	:	Artificial Intelligence and Data Science
Academic Year	:	2025 – 2026 (ODD)

RAJALAKSHMI ENGINEERING COLLEGE

[AUTONOMOUS]

RAJALAKSHMI NAGAR, THANDALAM – 602 105
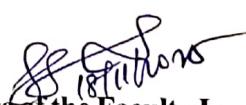
BONAFIDE CERTIFICATE

Name: ...... C. PARVENDHAN ....

Academic Year: 2025-2026 (ODD)   Semester: III   Branch: AIDS

Register Number: | 2116 241801197 |

Certified that this is the bonafide record of work done by the above student in the **CS23331 – DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY** during the year 2025 - 2026.

Signature of the Faculty In- charge

Submitted for the Practical Examination held on ....

Internal Examiner

RAJALAKSHMI ENGINEERING COLLEGE
(An Autonomous Institution affiliated to Anna University)
DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Subject: CS23331 – DESIGN AND ANALYSIS OF ALGORITHMS

Reg No: 241801197

Name: PARVENDHAN C

Department: AIDS

INDEX

S.NO	Date	Name of the Experiment	GitHub QR Code	Marks Awarded	Signature
1.	26/7/2025	Basic C Programming		9	
2.	21/08/25	Finding Time Complexity of Algorithms		8	
3.	19/9/25	Greedy Algorithms		8	
4.	26/9/25	Divide and Conquer		9	

S.NO	Date	Name of the Experiment	GitHub QR Code	Marks Awarded	Signature
5.	24|10|25	Dynamic Programming			
6.	31|10|25	Competitive Programming			

NAME: PARVENDHAN C
REG NO: 241801197
DYNAMIC PROGRAMMING PROGRAM 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 %)

```c
#include <stdio.h>

long long countWays(int n) {
    if (n < 0) return 0;
    if (n == 0) return 1;

    long long dp[n + 1];
    dp[0] = 1;

    for (int i = 1; i <= n; i++) {
        dp[i] = 0;
        if (i - 1 >= 0)
            dp[i] += dp[i - 1];
        if (i - 3 >= 0)
            dp[i] += dp[i - 3];
    }

    return dp[n];
}

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

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

Passed all tests! ✔

PROGRAM 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 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

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int max(int a, int b)
{
    return (a > b) ? a : b;
}
int main()
{
    int n;
    scanf("%d", &n);
    int board[n][n];
    int dp[n][n];
    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++)
    {
        for (int j = 1; j < n; j++)
        {
            dp[i][j] = board[i][j] + max(dp[i-1][j], dp[i][j-1]);
        }
    }
    printf("%d\n", dp[n-1][n-1]);
    return 0;
}
```

	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.

PROGRAM 3:

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	2
azb	

Answer: (penalty regime: 0 %)

```c
1  #include <stdio.h>
2  #include <string.h>
3  int max(int a, int b)
4  {
5      return (a > b) ? a : b;
6  }
7  int main()
8  {
9      char s1[1000], s2[1000];
10     scanf("%s %s", s1, s2);
11     int m = strlen(s1);
12     int n = strlen(s2);
13     int dp[m+1][n+1];
14     for (int i = 0; i <= m; i++)
15     {
16         for (int j = 0; j <= n; j++)
17         {
18             if (i == 0 || j == 0)
19                 dp[i][j] = 0;
20             else if (s1[i-1] == s2[j-1])
21                 dp[i][j] = dp[i-1][j-1] + 1;
22             else
23                 dp[i][j] = max(dp[i-1][j], dp[i][j-1]);
24         }
25     }
26     printf("%d\n", dp[m][n]);
27     return 0;
28 }
29
```

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

Passed all tests! ✔

PROGRAM 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 %)

```c
#include <stdio.h>
int max(int a, int b)
{
    return (a > b) ? a : b;
}
int main()
{
    int n;
    scanf("%d", &n);
    int seq[n], dp[n];
    for (int i = 0; i < n; i++)
        scanf("%d", &seq[i]);
    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 (seq[i] >= seq[j])
            {
                dp[i] = max(dp[i], dp[j] + 1);
            }
        }
    }
    int ans = 0;
    for (int i = 0; i < n; i++)
        if (dp[i] > ans)
            ans = dp[i];
    printf("%d\n", ans);
    return 0;
}
```

	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.