****

**NAME: -** Darji Akshatkumar Hiteshbhai

**RollNo: -** 23MCD001

**Branch: -** M.tech-CSE**(Data Science)**

**Subject: -** Complexity Theory & Algorithms

**Practical-8**

**Aim:** Implement matrix chain multiplication using dynamic programming concepts.

**Code for matrix chain multiplication –**

#include <bits/stdc++.h>

using namespace std;

void PRINT\_OPT\_PARENS(vector<vector<int>>& s, int i, int j) {

    if (i == j) {

        cout << "A" << i;

    } else {

        cout << "(";

        PRINT\_OPT\_PARENS(s, i, s[i][j]);

        PRINT\_OPT\_PARENS(s, s[i][j] + 1, j);

        cout << ")";

    }

}

void PRINT\_DP\_TABLE(vector<vector<int>>& dp, vector<vector<int>>& s, int N) {

    cout << "DP Table:" << endl;

    for (int i = 1; i < N; i++) {

        for (int j = 1; j < N; j++) {

            if (i == j) {

                cout << "0";

            } else {

                cout << "[" << dp[i][j] << " ";

                cout << "k=" << s[i][j] << "]";

            }

            if (j < N - 1) {

                cout << "\t";

            }

        }

        cout << endl;

    }

}

int f(vector<int>& arr, int i, int j, vector<vector<int>>& dp, vector<vector<int>>& s) {

    if (i == j)

        return 0;

    if (dp[i][j] != -1)

        return dp[i][j];

    int mini = INT\_MAX;

    for (int k = i; k <= j - 1; k++) {

        int ans = f(arr, i, k, dp, s) + f(arr, k + 1, j, dp, s) + arr[i - 1] \* arr[k] \* arr[j];

        if (ans < mini) {

            mini = ans;

            s[i][j] = k; // Store the optimal value of k

        }

    }

    return dp[i][j] = mini;

}

int matrixMultiplication(vector<int>& arr, int N) {

    vector<vector<int>> dp(N, vector<int>(N, -1));

    vector<vector<int>> s(N, vector<int>(N, -1));

    int i = 1;

    int j = N - 1;

    int minOperations = f(arr, i, j, dp, s);

    cout << "The minimum number of operations is " << minOperations << endl;

    cout << "Optimal Parentheses Placement: ";

    PRINT\_OPT\_PARENS(s, i, j);

    cout << endl;

    PRINT\_DP\_TABLE(dp, s, N);

    return minOperations;

}

int main() {

    //values of p

    vector<int> arr = {40,200,3,100,50};

    int n = arr.size();

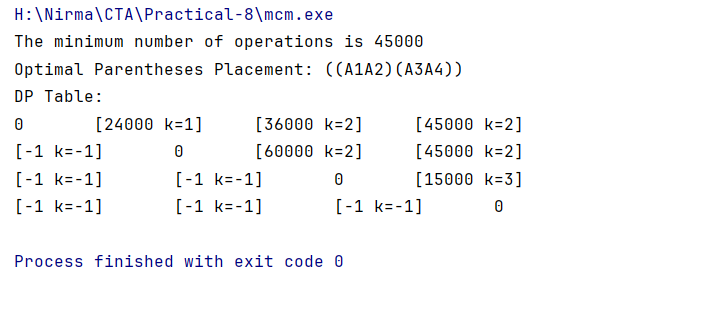
    matrixMultiplication(arr, n);

    return 0;

}

**Output –**

**Test Case – 1**

****