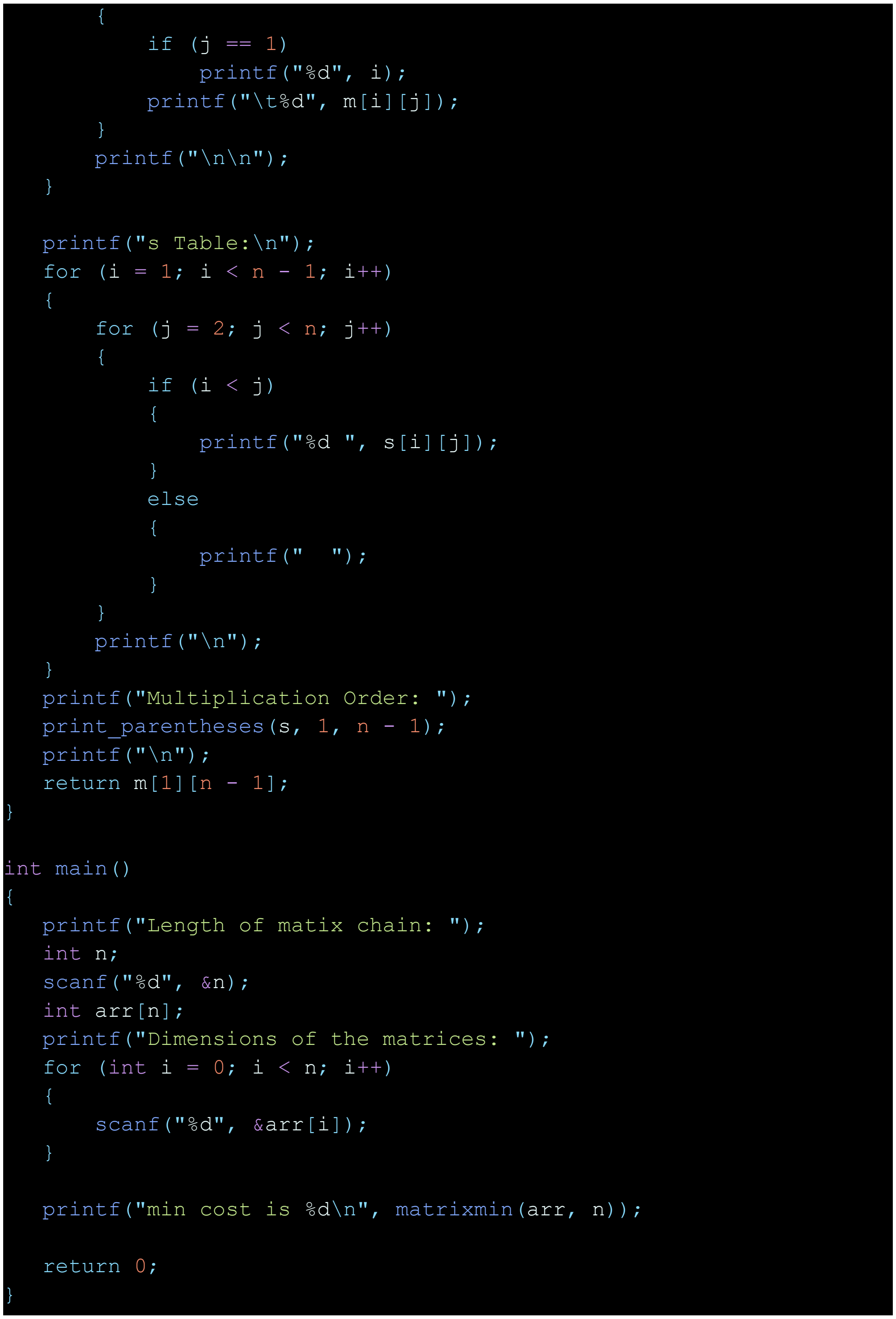
Name	Stephen David Vaz
UID no.	2021700070
Experiment No.	5

AIM:	Dynamic Programming - Matrix Chain Multiplication
PROBLEM STATEMENT:	Apply the concept of dynamic programming to solve the problem of finding the minimum cost i.e. multiplications required to perform Matrix Chain Multiplications
ALGORITHM/ THEORY:	Matrix Chain Multiplication can be solved using dynamic programming. We can define the minimum number of scalar multiplications needed to iteratively compute the product of a chain of matrices. We start with subchains of length 1 and then compute the minimum cost for subchains of increasing length until we have the minimum cost for the entire chain. The time complexity of this algorithm is O(n^3), where n is the number of matrices in the chain. **Algorithm:** 1. Define the subproblem: Find the minimum number of scalar multiplications needed to compute the product of a chain of matrices. 2. Find the recurrence relation: Let $M[i,j]$ be the minimum number of scalar multiplications needed to compute the product of the chain of matrices from matrix i to matrix j. We can define $M[i,j]$ recursively as follows: $M[i,j] = \min(M[i,k] + M[k+1,j] + a[i-1] \times a[k] \times a[j])$ for $i \leq k < j$ 3. Initialize the base case: $M[i,i] = 0$ for $1 \leq i \leq n$, where n is the number of matrices in the chain. 4. Solve the subproblems: Compute the minimum cost for subchains of increasing length until we have the minimum cost for the entire chain. 5. Return the final answer: The minimum cost for the entire chain is stored in $M[1,n]$, where n is the number of matrices in the chain.

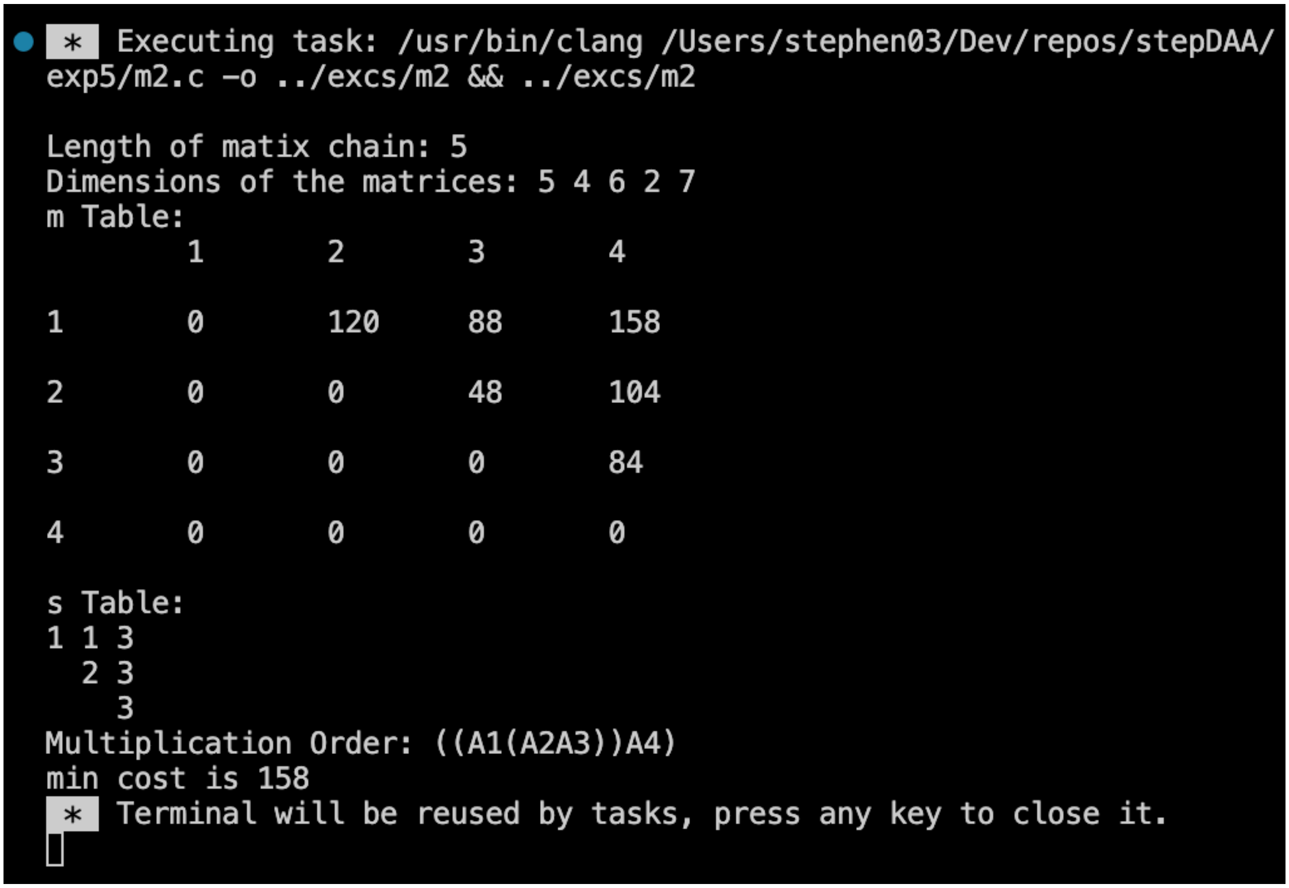
PROGRAM:

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

void print_parentheses(int s[][5], int i, int j)
{
    if (i == j)
    {
        printf("A%d", i);
        return;
    }
    printf("(");
    print_parentheses(s, i, s[i][j]);
    print_parentheses(s, s[i][j] + 1, j);
    printf(")");
}

int matrixmin(int p[], int n)
{
    int m[n][n];
    int s[n][n];
    memset(m, 0, sizeof(m[0][0]) * n * n);

    int i, j, k, L, q;

    for (L = 2; L < n; L++)
    {
        for (i = 1; i < n - L + 1; i++)
        {
            j = i + L - 1;
            m[i][j] = INT_MAX;
            for (k = i; k <= j - 1; k++)
            {
                q = m[i][k] + m[k + 1][j] + p[i - 1] * p[k] * p[j];
                if (q < m[i][j])
                {
                    m[i][j] = q;
                    s[i][j] = k;
                }
            }
        }
    }

    printf("m Table:\n");
    for (int i = 1; i < n; i++)
    {
        printf("\t%d", i);
    }

    printf("\n\n");
    for (i = 1; i < n; i++)
    {
        for (j = 1; j < n; j++)
```



```
        {
            if (j == 1)
                printf("%d", i);
            printf("\t%d", m[i][j]);
        }
        printf("\n\n");
    }

    printf("s Table:\n");
    for (i = 1; i < n - 1; i++)
    {
        for (j = 2; j < n; j++)
        {
            if (i < j)
            {
                printf("%d ", s[i][j]);
            }
            else
            {
                printf("  ");
            }
        }
        printf("\n");
    }
    printf("Multiplication Order: ");
    print_parentheses(s, 1, n - 1);
    printf("\n");
    return m[1][n - 1];
}

int main()
{
    printf("Length of matix chain: ");
    int n;
    scanf("%d", &n);
    int arr[n];
    printf("Dimensions of the matrices: ");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }

    printf("min cost is %d\n", matrixmin(arr, n));

    return 0;
}
```

RESULT:



```
● * Executing task: /usr/bin/clang /Users/stephen03/Dev/repos/stepDAA/
  exp5/m2.c -o ../excs/m2 && ../excs/m2

  Length of matix chain: 5
  Dimensions of the matrices: 5 4 6 2 7
  m Table:
          1       2       3       4

  1       0       120     88      158

  2       0       0       48      104

  3       0       0       0       84

  4       0       0       0       0

  s Table:
  1 1 3
    2 3
      3
  Multiplication Order: ((A1(A2A3))A4)
  min cost is 158
  * Terminal will be reused by tasks, press any key to close it.
```

For Matrix Chain [5,4,6,2,7]

CONCLUSION:	Successfully understood the application of dynamic programming in matrix chain multiplication. Also, understood how to find the minimum cost of matrix multiplications.