Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058, India

(Autonomous College Affiliated to University of Mumbai)

Experiment No.	**4**
Aim	**To implement matrix chain multiplication and also to compute its time complexity**
Subject.	**Design and Analysis of Algorithm**
Name	**Pratham Jain**
UID No.	**2021300051**
Class & Division	**Comps-A (A3- C batch)**
Date of Performance.	**20-03-23**
Date of Submission.	**27-03-23**

Theory:

Dynamic Programming is a technique in computer programming that helps to efficiently solve a class of problems that have overlapping sub-problems and optimal substructure property. If any problem can be divided into sub-problems, which in turn are divided into smaller sub-problems, and if there are overlapping among these sub-problems, then the solutions to these sub-problems can be saved for future reference. The approach of solving
problems using dynamic programming algorithm has following steps:

1. Characterize the structure of an optimal solution.
2. Recursively define the value of an optimal solution.
3. Compute the value of an optimal solution, typically in a bottom-up fashion.
4. Construct an optimal solution from computed information.

Algorithm:

MATRIX-CHAIN-ORDER (p):

1. n   length[p]-1
2. for i ← 1 to n
3. do m [i, i] ← 0
4. for l ← 2 to n   // l is the chain length
5. do for i ← 1 to n-l + 1

6. do j ← i+ l -1
7. m[i,j] ← ∞
8. for k ← i to j-1
9. do q ← m [i, k] + m [k + 1, j] + pi-1 pk pj
10. If q < m [i,j]
11. then m [i,j] ← q
12. s [i,j] ← k

Code:

```
#include<stdio.h>
#include<limits.h>

// Matrix Ai has dimension p[i-1] x p[i] for i = 1..n

int MatrixChainMultiplication(int p[], int n)
{
    int m[n][n];
    int i, j, k, L, q;

    for (i=1; i<n; i++)
        m[i][i] = 0;    //number of multiplications are 0(zero) when there is only 
one matrix

    //Here L is chain length. It varies from length 2 to length n.
    for (L=2; L<n; L++)
    {
        for (i=1; i<n-L+1; i++)
        {
            j = i+L-1;
            m[i][j] = INT_MAX;  //assigning to maximum value

            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;    //if number of multiplications found less that 
number will be updated.
                }
            }
        }
    }

    return m[1][n-1];   //returning the final answer which is M[1][n]
```

```
}

int main()
{
    int n,i;
    printf("Enter number of matrices\n");
    scanf("%d",&n);

    n++;

    int arr[n];

    printf("Enter dimensions \n");

    for(i=0;i<n;i++)
    {
        printf("Enter d%d :: ",i);
        scanf("%d",&arr[i]);
    }

    int size = sizeof(arr)/sizeof(arr[0]);

    printf("Minimum number of multiplications is %d ",
MatrixChainMultiplication(arr, size));

    return 0;
}
```

Output:

```
PS C:\COLLEGE\CODING (psipl psoop DS)\SEM 4 DAA\EXP 4\output> & .
Enter number of matrices
3
Enter dimensions
Enter d0 :: 5
Enter d1 :: 3
Enter d2 :: 2
Enter d3 :: 1
Minimum number of multiplications is 21
PS C:\COLLEGE\CODING (psipl psoop DS)\SEM 4 DAA\EXP 4\output>
```

Conclusion: I have learned and successfully implemented matrix chain multiplication program.