Bharatiya Vidya Bhavan’s

Sardar Patel Institute of Technology

Bhavan’s Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

SE – COMP (SE-A/08) **Sub- DAA Lab**

Name	**Naman Badlani**
UID No.	**2021300008**
Subject	**Design And Analysis Of Algorithm**
Class	**Comps A**
Experiment No.	**5**
AIM	To implement Matrix Chain Multiplication

Theory –

Matrix Chain Multiplication is an optimization problem that involves finding the most efficient way to multiply a sequence of matrices. The problem is to determine the order in which the matrices should be multiplied to minimize the number of scalar multiplications.

The idea is to consider all possible ways of splitting the matrix sequence into two sub-sequences, compute the minimum cost of multiplying each sub-sequence recursively, and add the cost of multiplying the resulting matrices. We then take the minimum of all possible ways of splitting the matrix sequence.

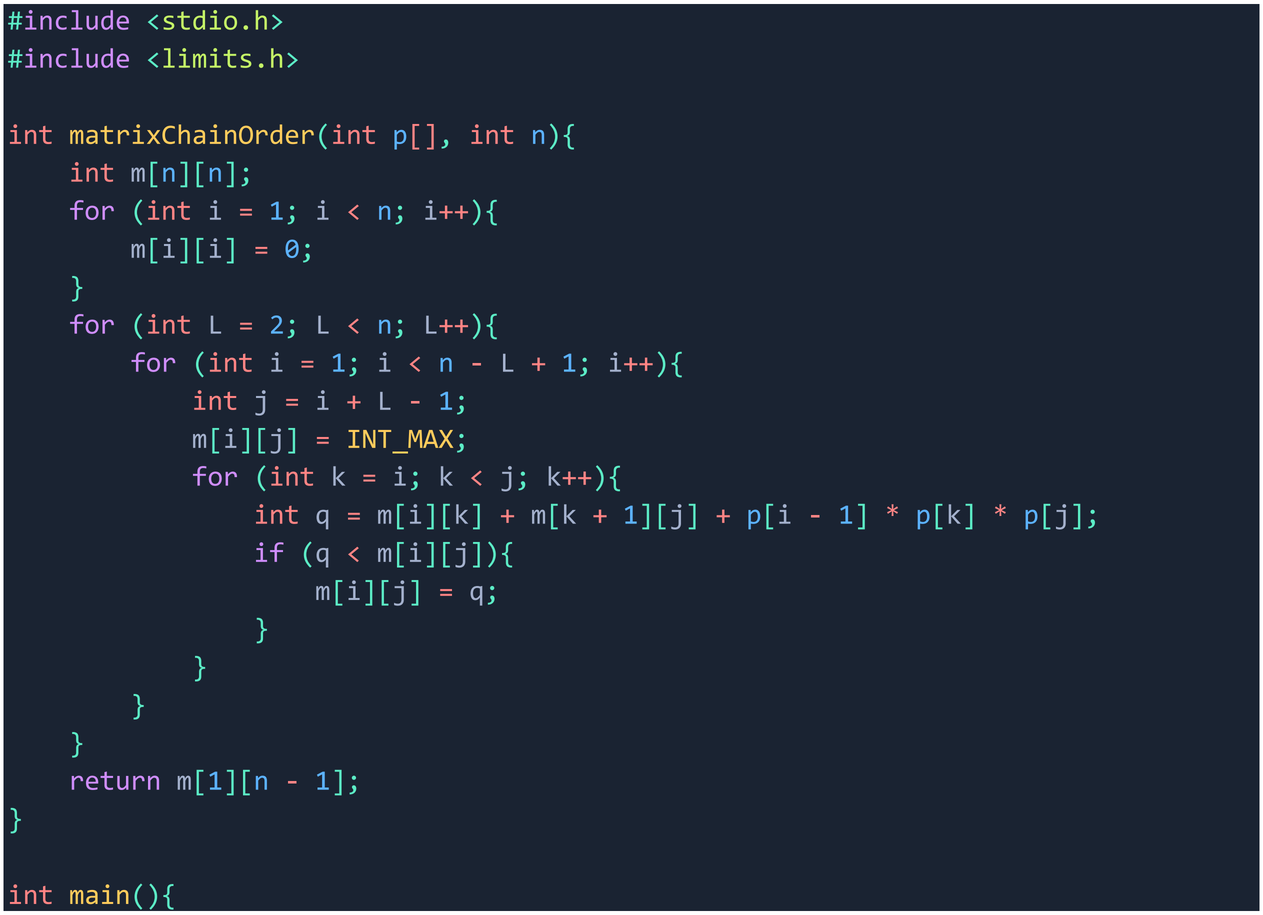
The overall time complexity of this algorithm is O(n^3), where n is the number of matrices in the sequence. This makes it much more efficient than simply trying all possible orders of matrix multiplication, which would take O(n!) time.

Algorithm –

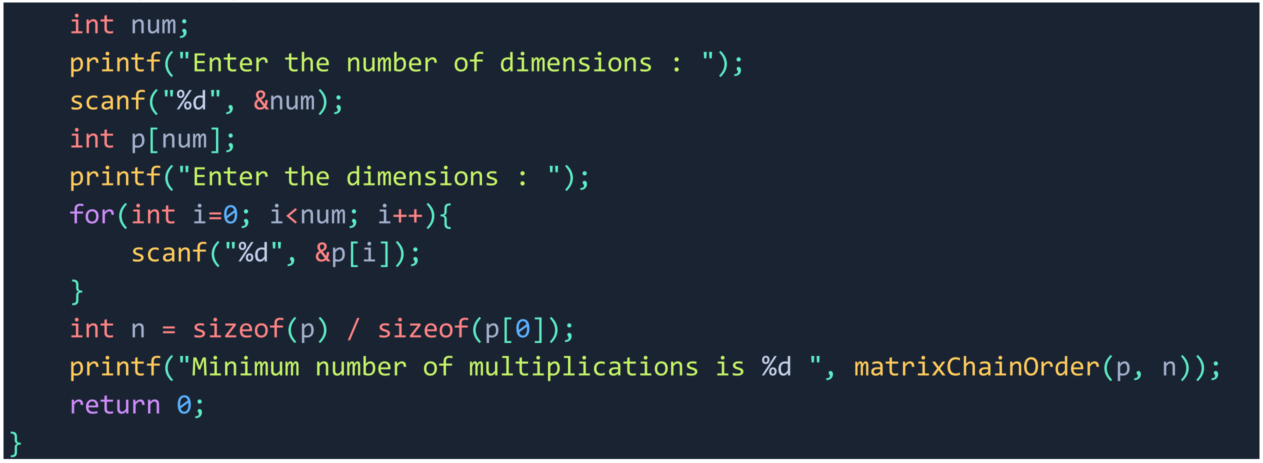
```
1. n = length[p]-1
 2. for i ← 1 to n
 3. do m [i, i] ← 0
```

```
for l ← 2 to n    // l is the chain length
    do for i ← 1 to n-l + 1
        do j ← i+ l -1
            m[i,j] ← ∞
            for k ← i to j-1
                do q ← m [i, k] + m [k + 1, j] + pi-1 pk pj
                    If q < m [i,j]
                        then m [i,j] ← q
                            s [i,j] ← k
return m and s.
```

Program –

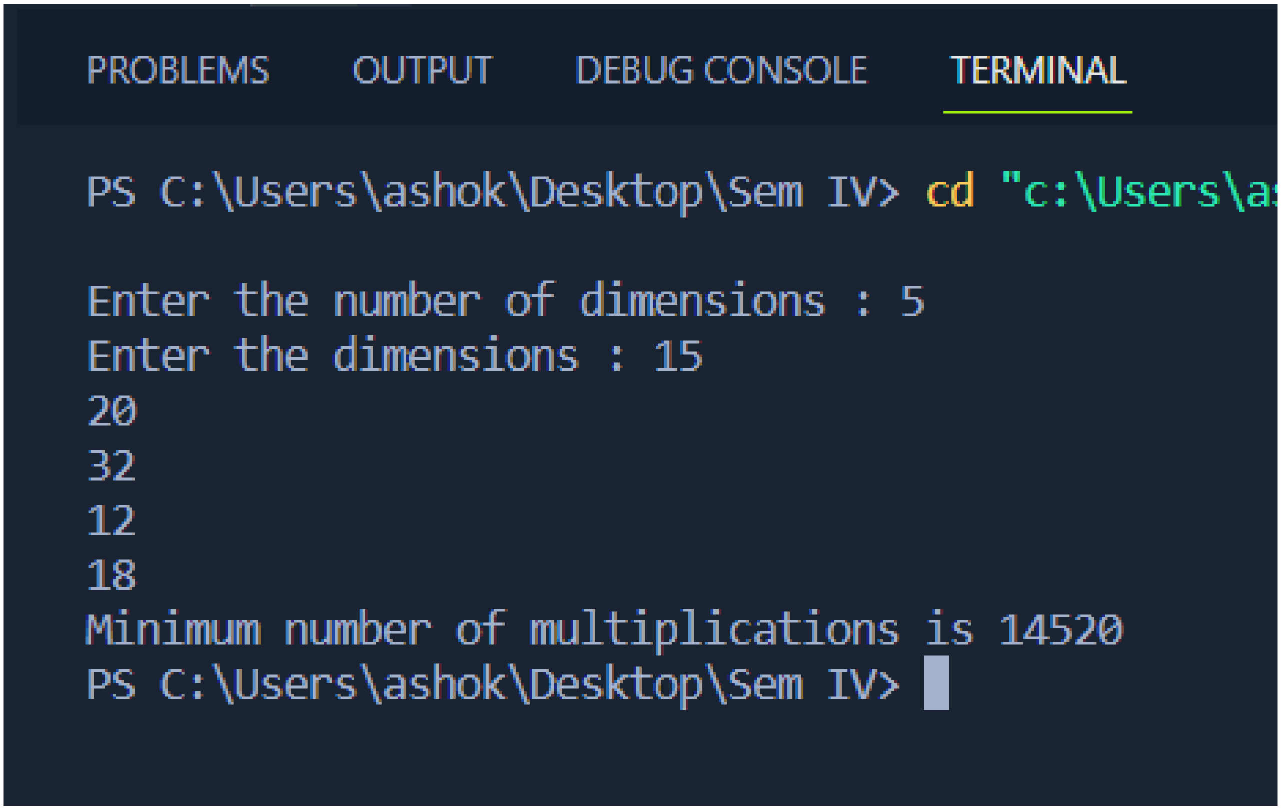


```c
#include <stdio.h>
#include <limits.h>

int matrixChainOrder(int p[], int n){
    int m[n][n];
    for (int i = 1; i < n; i++){
        m[i][i] = 0;
    }
    for (int L = 2; L < n; L++){
        for (int i = 1; i < n - L + 1; i++){
            int j = i + L - 1;
            m[i][j] = INT_MAX;
            for (int k = i; k < j; k++){
                int q = m[i][k] + m[k + 1][j] + p[i - 1] * p[k] * p[j];
                if (q < m[i][j]){
                    m[i][j] = q;
                }
            }
        }
    }
    return m[1][n - 1];
}

int main(){
```



```
    int num;
    printf("Enter the number of dimensions : ");
    scanf("%d", &num);
    int p[num];
    printf("Enter the dimensions : ");
    for(int i=0; i<num; i++){
        scanf("%d", &p[i]);
    }
    int n = sizeof(p) / sizeof(p[0]);
    printf("Minimum number of multiplications is %d ", matrixChainOrder(p, n));
    return 0;
}
```

Result Analysis –



```
PS C:\Users\ashok\Desktop\Sem IV> cd "c:\Users\a

Enter the number of dimensions : 5
Enter the dimensions : 15
20
32
12
18
Minimum number of multiplications is 14520
PS C:\Users\ashok\Desktop\Sem IV>
```