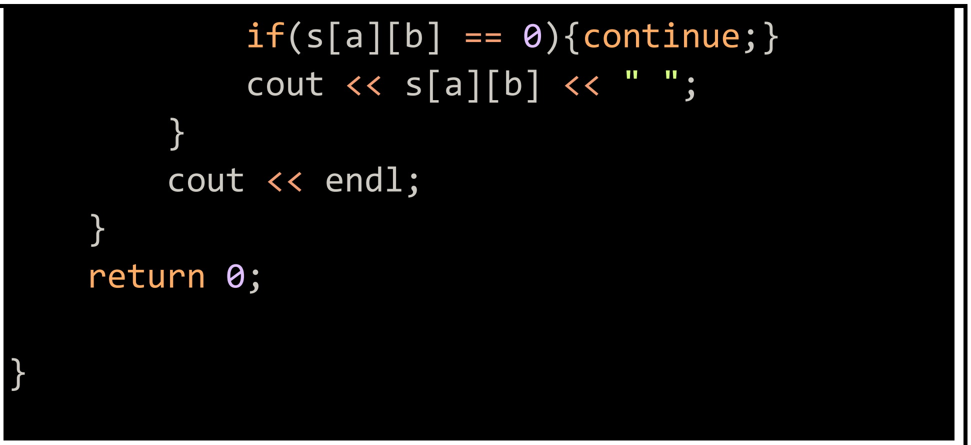
NAME:	Shubham Solanki
UID:	2022301015
SUBJECT	Design and Analysis of Algorithms
EXPERIMENT NO:	4
AIM:	To implement Matrix Chain Multiplication
Algorithm:	**MATRIX-CHAIN-ORDER (p)** 1. n length[p]-1 2. for i ← 1 to n 3. do m [i, i] ← 0 4. for l ← 2 to n // l is the chain length 5. do for i ← 1 to n-l + 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 13.return m and s. **PRINT-OPTIMAL-PARENS (s, i, j)** 1. if i=j 2. then print "A" 3. else print "(" 4. PRINT-OPTIMAL-PARENS (s, i, s [i, j]) 5. PRINT-OPTIMAL-PARENS (s, s [i, j] + 1, j)

6. print ")"

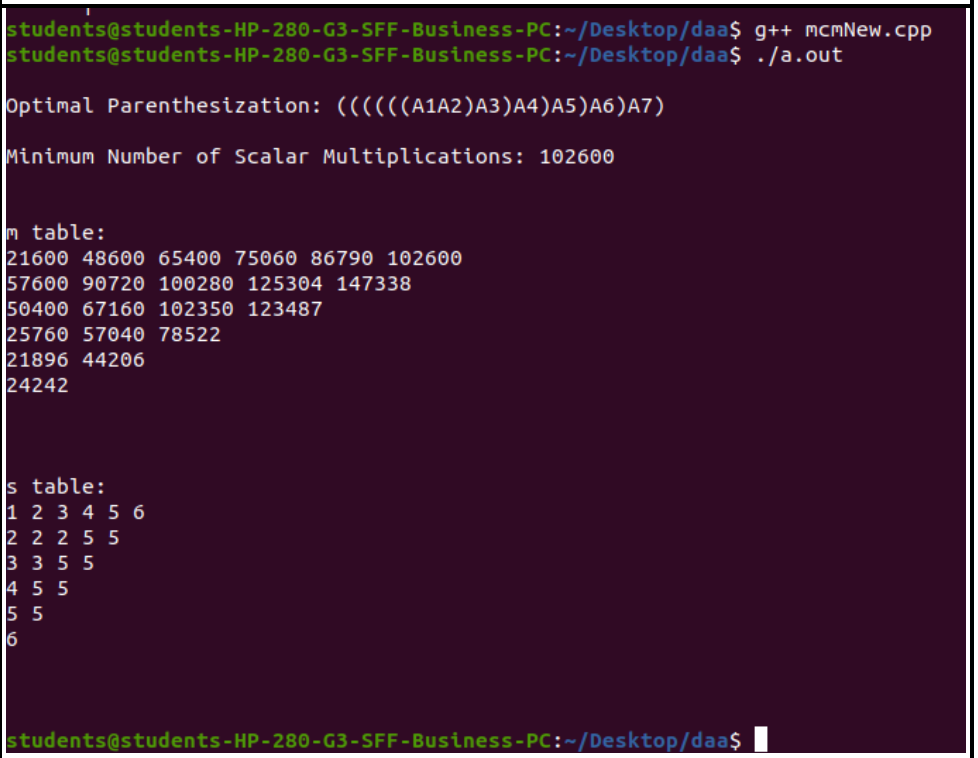
Code:

```
#include <iostream>
#include <climits>
#include <random>
#include <ctime>
using namespace std;

void matrixChainOrder(int p[], int n, int
m[][100], int s[][100])
{
    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-1; 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;

                    s[i][j] = k;

                }
            }

        }

    }

}



void printOptimalParenthesis(int s[][100], int i, int j)
{
    if(i == j)
    cout << "A" << i;
    else
    {
        cout << "("; printOptimalParenthesis(s, i, s[i][j]); printOptimalParenthesis(s, s[i][j]+1, j); cout << ")";
    }

}

int main()
{
    int p[8];
    srand ( time(NULL) );
    random_device rd;
    mt19937 gen(rd());
    uniform_int_distribution<> distr(15, 46);
    for(int i=0; i<10; ++i)
    p[i] = distr(gen);

    int n = sizeof(p)/sizeof(p[0]) - 1;
    int m[100][100];

    int s[100][100];

    matrixChainOrder(p, n, m, s);

    cout << "\nOptimal Parenthesization: ";
printOptimalParenthesis(s, 1, n);
    cout << endl;

    cout << "\nMinimum Number of Scalar Multiplications: " << m[1][n] << endl;
    cout << "\n\nm table:";
    for(int a = 0; a < 8; a++)

    {

        for(int b = 0; b < 8; b++)

        {

            if(m[a][b] == 0){continue;}
            cout << m[a][b] << " ";
        }

        cout << endl;

    }
    cout << "\n\ns table:";

    for(int a = 0; a < 10; a++)
    {
        for(int b = 0; b < 10; b++)
        {
```



```
            if(s[a][b] == 0){continue;}
            cout << s[a][b] << " ";
        }
        cout << endl;
    }
    return 0;

}
```

Output



```
students@students-HP-280-G3-SFF-Business-PC:~/Desktop/daa$ g++ mcmNew.cpp
students@students-HP-280-G3-SFF-Business-PC:~/Desktop/daa$ ./a.out

Optimal Parenthesization: ((((((A1A2)A3)A4)A5)A6)A7)

Minimum Number of Scalar Multiplications: 102600


m table:
21600 48600 65400 75060 86790 102600
57600 90720 100280 125304 147338
50400 67160 102350 123487
25760 57040 78522
21896 44206
24242



s table:
1 2 3 4 5 6
2 2 2 5 5
3 3 5 5
4 5 5
5 5
6



students@students-HP-280-G3-SFF-Business-PC:~/Desktop/daa$ 
```

Conclusion:

Thus, after performing this experiment I understood how matrix chain multiplication works and how significant it is while multiplying metrices