

DAA – LAB 4

Name : Madhuramsinh Solanki

Reg no: 22BRS1327

Q1 Matrix chain multiplication using dp approach

Code:

```
#include <iostream>

#include <vector>

#include <climits>

using namespace std;

int minMultiplications(const vector<int>& arr) {

    int n = arr.size();

    vector<vector<int>> dp(n, vector<int>(n, 0));

    for (int len = 2; len < n; ++len) {

        for (int i = 0; i < n - len; ++i) {

            int j = i + len;

            dp[i][j] = INT_MAX;

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

                int cost = dp[i][k] + dp[k][j] + arr[i] * arr[k] * arr[j];

                if (cost < dp[i][j]) {

                    dp[i][j] = cost;

                }

            }

        }

    }

    return dp[0][n - 1];

}

int main() {

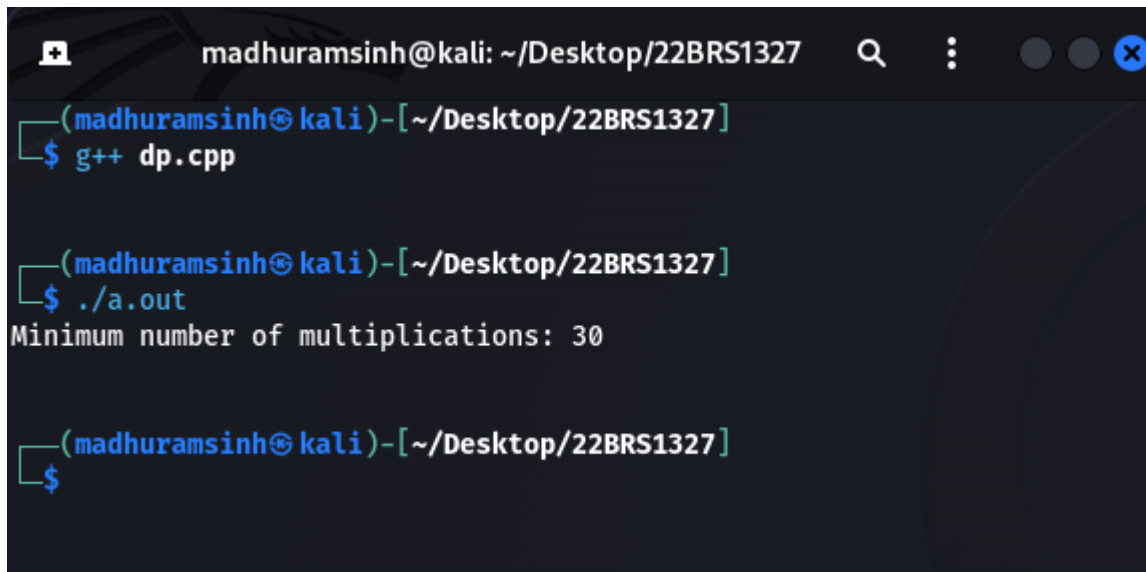
    vector<int> arr = {1, 2, 3, 4, 3};

    cout << "Minimum number of multiplications: " << minMultiplications(arr) << endl;

    return 0;

}
```

Output:

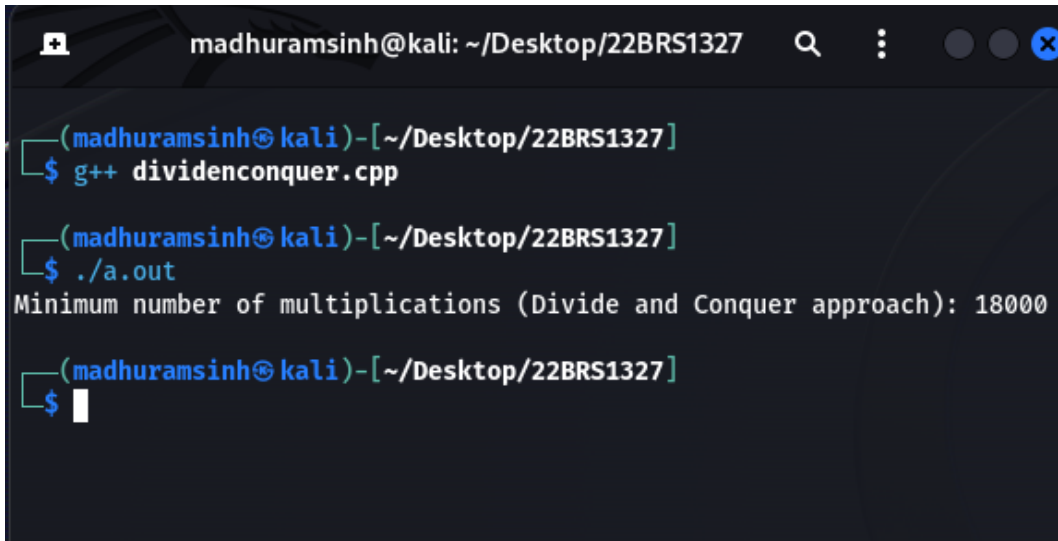
A terminal window with a dark background. The title bar shows the user 'madhuramsinh@kali' and the directory '~/Desktop/22BRS1327'. The prompt is '(madhuramsinh@kali)-[~/Desktop/22BRS1327]'. The user enters '\$ g++ dp.cpp' and the prompt changes to '\$ '. The user then enters './a.out' and the program outputs 'Minimum number of multiplications: 30'. The prompt returns to '(madhuramsinh@kali)-[~/Desktop/22BRS1327]' and the user enters '\$' again.

```
(madhuramsinh@kali)-[~/Desktop/22BRS1327]  
$ g++ dp.cpp  
  
(madhuramsinh@kali)-[~/Desktop/22BRS1327]  
$ ./a.out  
Minimum number of multiplications: 30  
  
(madhuramsinh@kali)-[~/Desktop/22BRS1327]  
$
```

Q2 Matrix chain multiplication using divide n conquer approach

```
#include <iostream>  
  
#include <vector>  
  
#include <climits>  
  
using namespace std;  
  
int matrixChainMultiplicationDC(const vector<int>& p, int i, int j) {  
    if (i == j) return 0;  
  
    int minCost = INT_MAX;  
  
    for (int k = i; k < j; ++k) {  
        int cost = matrixChainMultiplicationDC(p, i, k) +  
                    matrixChainMultiplicationDC(p, k + 1, j) +  
                    p[i - 1] * p[k] * p[j];  
  
        minCost = min(minCost, cost);  
    }  
  
    return minCost;  
}  
  
int main() {  
    vector<int> p = {10, 20, 30, 40};  
  
    int result = matrixChainMultiplicationDC(p, 1, p.size() - 1);  
  
    cout << "Minimum number of multiplications (Divide and Conquer approach): " << result << endl;  
  
    return 0;}
```

Output:



```
madhuramsinh@kali: ~/Desktop/22BRS1327
(madhuramsinh@kali)-[~/Desktop/22BRS1327]
$ g++ dividenconquer.cpp
(madhuramsinh@kali)-[~/Desktop/22BRS1327]
$ ./a.out
Minimum number of multiplications (Divide and Conquer approach): 18000
(madhuramsinh@kali)-[~/Desktop/22BRS1327]
$
```

Q2 Matrix chain multiplication using Greedy approach

Code:

```
#include <iostream>

#include <vector>

#include <limits>

using namespace std;

int matrixChainMultiplicationGreedy(const vector<int>& p) {

    int n = p.size();

    int minMultiplications = 0;

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

        int cost = p[i - 1] * p[i] * p[i + 1];

        minMultiplications += cost;

    }

    return minMultiplications;

}

int main() {

    vector<int> p = {1,2,3,4,3};

    int result = matrixChainMultiplicationGreedy(p);

    cout << "Estimated number of multiplications (Greedy approach): " << result << endl;

    return 0;

}
```

Output:

```
madhuramsinh@kali: ~/Desktop/22BRS1327
(madhuramsinh@kali)-[~/Desktop/22BRS1327]
$ g++ greedy.cpp
(madhuramsinh@kali)-[~/Desktop/22BRS1327]
$ ./a.out
Estimated number of multiplications (Greedy approach): 66
(madhuramsinh@kali)-[~/Desktop/22BRS1327]
$
```