**CHANDIGARH UNIVERSITY**

**UNIVERSITY INSTITUTE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



|  |  |
| --- | --- |
| **Submitted By: Kanishk Soni Submitted To: Er. Tanu Dhiman** | |
| **Subject Name** | **Design and Analysis of Algorithm Lab** |
| **Subject Code** | **20CSP-312** |
| **Branch** | **BE-CSE** |
| **Semester** | **5th** |

**Worksheet - 5**

**Student Name:** Kanishk Soni **UID:** 20BCS9398

**Branch:** BE CSE **Section/Group:** 20BCS\_WM\_707-B

**Semester:** 5 **Date of Performance:** 02/10/2022

**Subject Name:** Design & Analysis of Algorithm **Subject Code:** 20CSP-312

1. **Aim/Overview of the practical:** Code and analyze to find an optimal solution to matrix chain multiplication using dynamic programming.
2. **Task to be done:** Find the minimum multiplication operations required for multiply n matrices.
3. **Algorithms:**

1. Build a matrix dp[][] of size N\*N for memoization purposes.

2. Use the same recursive call as done in the above approach:

3. When we find a range (i, j) for which the value is already calculated, return the minimum value for that range (i.e., dp[i][j]).

4. Otherwise, perform the recursive calls as mentioned earlier.

5. The value stored at dp[0][N-1] is the required answer.

1. **Code:**

#include <iostream>

#include <climits>

using namespace std;

int matrixChain(int n, int order[]) {

int i, j, k;

int tempValue;

int dp[n + 1][n + 1];

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

dp[i][i] = 0;

}

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

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

j = i + size - 1;

dp[i][j] = INT\_MAX;

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

tempValue = dp[i][k] + dp[k + 1][j] + order[i - 1] \* order[k] \* order[j];

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

dp[i][j] = tempValue;

}

}

}

}

return dp[1][n];

}

int main() {

int i, j;

int n;

cout << "Enter the number of matrices in the chain(greater than 1): ";

cin >> n;

int order[n + 1];

cout << "Enter the order array of the matrix chain (" << n + 1 << " elements): " << endl;

for(i = 0; i <= n; i++) {

cin >> order[i];

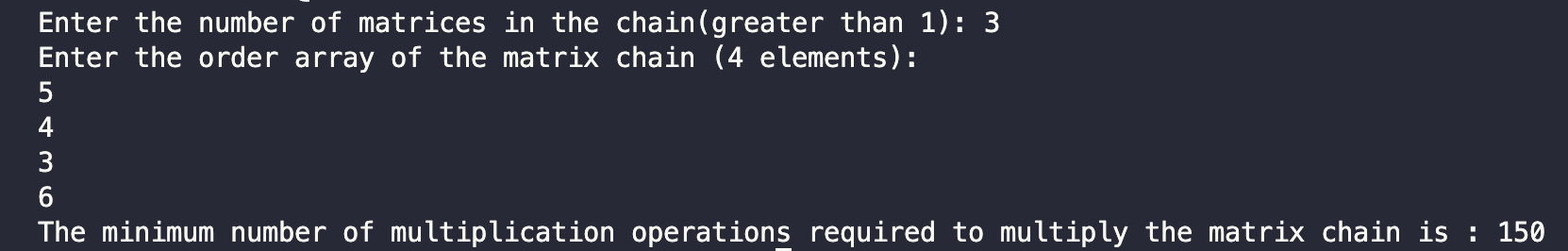
}

cout << "The minimum number of multiplication operations required to multiply the matrix chain is : "<<matrixChain(n,order)<< endl;

return 0;

}

1. **Complexity:** O(n^3)
2. **Result/Output:**



**Learning outcomes (What I have learnt):**

1. Create a program keeping in mind the time complexity
2. Create a program keeping in mind the space complexity
3. Steps to make optimal algorithm
4. Learnt about matrix application using dynamic programming