**Experiment 12**

**Write a program to implement the dynamic algorithm to solve the Matrix-chain multiplication problem.**

**Program:-**

#include <stdio.h>

#include <stdlib.h>

#include <limits.h>

#include <conio.h>

#include <time.h>

int MatrixChainMultiplication(int p[], int n)

{

int m[n][n];

int i, j, k, L, q;

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

{

m[i][i] = 0;

}

// Loop through chain lengths (2 to n-1)

for (int L = 2; L < n; L++)

{

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

{

j = i + L - 1;

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

for (int 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;

}

}

}

}

return m[1][n - 1];

}

int main() {

int n, i;

double time;

clock\_t start = clock();

printf("Enter number of matrices\n");

scanf("%d", &n);

n++;

int arr[n];

printf("Enter dimensions\n");

start = clock();

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

printf("Enter d%d: ", i);

scanf("%d", &arr[i]);

}

// Calculate and display minimum cost

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

printf("Minimum number of multiplications: %d\n", MatrixChainMultiplication(arr, size));

clock\_t end = clock();

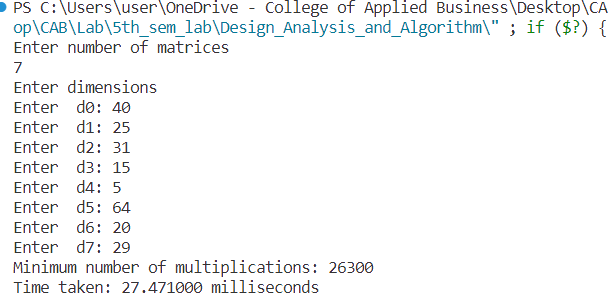
time = ((double)(end - start) + 1000) / CLOCKS\_PER\_SEC;

printf("Time taken: %lf milliseconds\n", time);

return 0;

}

**Output:**



**Conclusion:**

This experiment had been conducted in a 64-bit system with 16 GB RAM and Processor 12th Gen Intel(R) Core (TM) i5-12500H 3.10 GHz. The algorithm was implemented in C programming language in Visual Studio Code 1.85.1 Code Editor. The time taken by this algorithm for 7 number of input size is 27.47100 milliseconds. The running time is analyzed as O(n2).