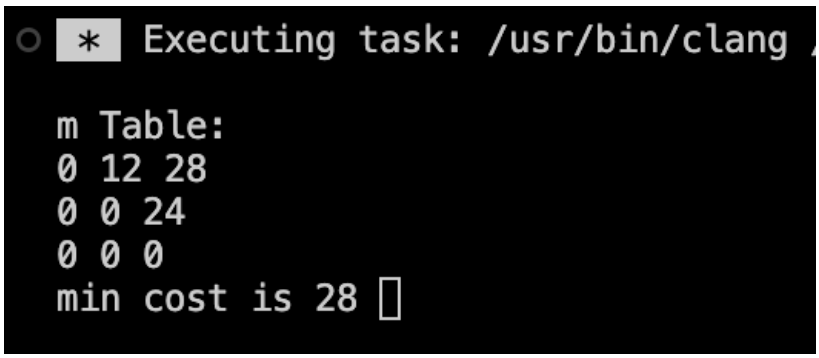
Name	Stephen David Vaz
UID no.	2021700070
Experiment No.	5

AIM:	Dynamic Programming - Matrix Chain Multiplication
PROBLEM STATEMENT:	Apply the concept of dynamic programming to solve the problem of finding the minimum cost i.e. multiplications required to perform Matrix Chain Multiplications
ALGORITHM/ THEORY:	Matrix Chain Multiplication can be solved using dynamic programming. We can define the minimum number of scalar multiplications needed to iteratively compute the product of a chain of matrices. We start with subchains of length 1 and then compute the minimum cost for subchains of increasing length until we have the minimum cost for the entire chain. The time complexity of this algorithm is O(n^3), where n is the number of matrices in the chain. **Algorithm:** 1. Define the subproblem: Find the minimum number of scalar multiplications needed to compute the product of a chain of matrices. 2. Find the recurrence relation: Let M[i,j] be the minimum number of scalar multiplications needed to compute the product of the chain of matrices from matrix i to matrix j. We can define M[i,j] recursively as follows: M[i,j] = min(M[i,k] + M[k+1,j] + a[i-1] x a[k] x a[j]) for $i \leq k < j$ 3. Initialize the base case: M[i,i] = 0 for $1 \leq i \leq n$, where n is the number of matrices in the chain. 4. Solve the subproblems: Compute the minimum cost for subchains of increasing length until we have the minimum cost for the entire chain. 5. Return the final answer: The minimum cost for the entire chain is stored in M[1,n], where n is the number of matrices in the chain.

PROGRAM:

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

int matrixmin(int p[], int n)
{
   int m[n][n];
   memset(m, 0, sizeof(m[0][0]) * n * n);

   int i, j, k, L, q;

   for (i = 1; i < n; i++)
       m[i][i] = 0;

   for (L = 2; L < n; L++) {
       for (i = 1; i < n - L + 1; i++) {
           j = i + L - 1;
           m[i][j] = INT_MAX;
           for (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;
           }
       }
   }
   printf("m Table:\n");
   for (int i = 1; i < n; i++)
   {
       for (j = 1; j < n; j++)
       {
           printf("%d ", m[i][j]);
       }
       printf("\n");
   }

   return m[1][n - 1];
}

int main()
{
   int arr[] = { 2, 3, 2, 4 };
   int size = sizeof(arr) / sizeof(arr[0]);

   printf("min cost is %d ",
       matrixmin(arr, size));

   getchar();
   return 0;
}
```

RESULT:



For Matrix Chain [2,3,2,4]

CONCLUSION:	Successfully understood the application of dynamic programming in matrix chain multiplication. Also, understood how to find the minimum cost of matrix multiplications.