```java
// Function to calculate minimum cost for matrix chain multiplication.
// The cost for MCM is number of multiplication operations needed to performed.
// We can solve this problem by using Dynamic Programming(tabular method)
static int matrixChainMultiplication(int N, int arr[]) {
    // storing multiplication operation or cost of multiplication.
    // m[i][j] = minimum cost for compute the matrix multiplication M[i]*...*M[j]
    // where the dimension of M[i] is (arr[i - 1] x arr[i])
    int m[][] = new int[N][N];
    // d = sequence length of multiplication.
    for (int d = 1; d < N - 1; d++) {
        for (int i = 1; i < N - d; i++) {
            int j = i + d;
            int min = 999999999;
            for (int k = i; k <= j - 1; k++) {
                // operations for multiplication
                int cost = m[i][k] + m[k + 1][j] + (arr[i - 1] * arr[k] * arr[j]);
                if (cost < min) {
                    min = cost;
                }
            }
            m[i][j] = min;
        }
    }
    return m[1][N - 1];
}
```