1. Minimum matrix multiplication algorithm

import java.io.\*;

import java.util.\*;

import java.time.LocalDate;

import java.lang.\*;

public class MinMultiplication

{

static int[][] dp = new int[100][100];

static int matrixMultMin(int[] p, int i, int j)

{

if (i == j)

{

return 0;

}

if (dp[i][j] != -1)

{

return dp[i][j];

}

dp[i][j] = Integer.MAX\_VALUE;

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

{

dp[i][j] = Math.min(

dp[i][j], matrixMultMin(p, i, k)

+ matrixMultMin(p, k + 1, j)

+ p[i - 1] \* p[k] \* p[j]);

}

return dp[i][j];

}

public static int MatrixChainOrder(int[] p, int n)

{

int i = 1, j = n - 1;

return matrixMultMin(p, i, j);

}

// Driver Code

public static void main (String[] args)

{

long start=0;

long end=0;

int w=0;

int arr[] = { 1,2,3,4,5,10,20,30,34, 354, 567, 987, 636 ,678,789,799};

int n= arr.length;

start=System.nanoTime();

end=System.nanoTime();

for (int[] row : dp)

Arrays.fill(row, -1);

System.out.println("n = "+n+" and time taken in nanoseconds is "+(end-start));

}

}

**Output:**

harshavaidhyam@Harshas-MacBook-Pro Pitt term-1 % /usr/bin/env /Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java -XX:+ShowCodeDet

ailsInExceptionMessages -cp /Users/harshavaidhyam/Library/Application\ Support/Code/User/workspaceStorage/91b3d36993f9364f97484f699e8e3b35/redhat.java/

jdt\_ws/jdt.ls-java-project/bin MinMultiplication

**n = 16 and time taken in nanoseconds is 125**

1. Print optimal order algorithm

import java.io.\*;

import java.util.\*;

import java.time.LocalDate;

import java.lang.\*;

public class OptimalOrder

{

static char name;

static void OrderPara(int i, int j, int n, int[][] bracket)

{

if (i == j)

{

System.out.print(name++);

return;

}

System.out.print('(');

OrderPara(i, bracket[j][i], n, bracket);

OrderPara(bracket[j][i] + 1, j, n, bracket);

System.out.print(')');

}

static void matrixChainOrder(int[] p, int n)

{

int[][] m = new int[n][n];

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

{

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

{

int j = i + L - 1;

m[i][j] = Integer.MAX\_VALUE;

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

{

int q = m[i][k] + m[k + 1][j] + p[i - 1] \* p[k] \* p[j];

if (q < m[i][j])

{

m[i][j] = q;

m[j][i] = k;

}

}

}

}

name = 'A';

System.out.print("Optimal Parenthesization is: ");

OrderPara(1, n - 1, n, m);

}

public static void main(String[] args)

{

long start=0;

long end=0;

start=System.nanoTime();

end= System.nanoTime();

int[] arr = { 1,2};

int n = arr.length;

matrixChainOrder(arr, n);

System.out.println("\n Total time is"+(end-start));

}

}

Output:

harshavaidhyam@Harshas-MacBook-Pro Pitt term-1 % /usr/bin/env /Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java -XX:+ShowCodeDet

ailsInExceptionMessages -cp /Users/harshavaidhyam/Library/Application\ Support/Code/User/workspaceStorage/91b3d36993f9364f97484f699e8e3b35/redhat.java/

jdt\_ws/jdt.ls-java-project/bin OptimalOrder

**Optimal Parenthesization is: A**

**Total time in nano seconds is125**

**PLOT:**

Chart, line chart

Description automatically generated