**LAPORAN PRAKTIKUM ANALISIS ALGORITMA**

**Ditujukan Untuk Memenuhi Tugas Praktikum**

****

**Disusun Oleh:**

**Mutia Karimah (140810170002)**

**2018/2019**

**TEKNIK INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS PADJADJARAN**

1. Algoritma matrix chain multiplication

**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++)

{

//i is the first matrix of the chain

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

{

//j is the first matrix of the chain

j=i+size-1;

//now, calculate the min. multiplications required to compute product of the chain with matrices i, i+1,...,j

//First initialize the result to infinity and then replace if lesser results are obtained

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

//now, divide the chain of matrices i....j into two sub-chains i...k and k+1...j and use the already computed results of these sub-chains to compute the result of original chain

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

{

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

//if tempValue is lesser than the current value of dp[i][j], replace it

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

{

dp[i][j]=tempValue;

}

}

}

}

//return the min. multiplication operations for the original matrix

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];

//order of matrix i will be given by order[i-1]\*order[i]

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

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

{

cin>>order[i];

}

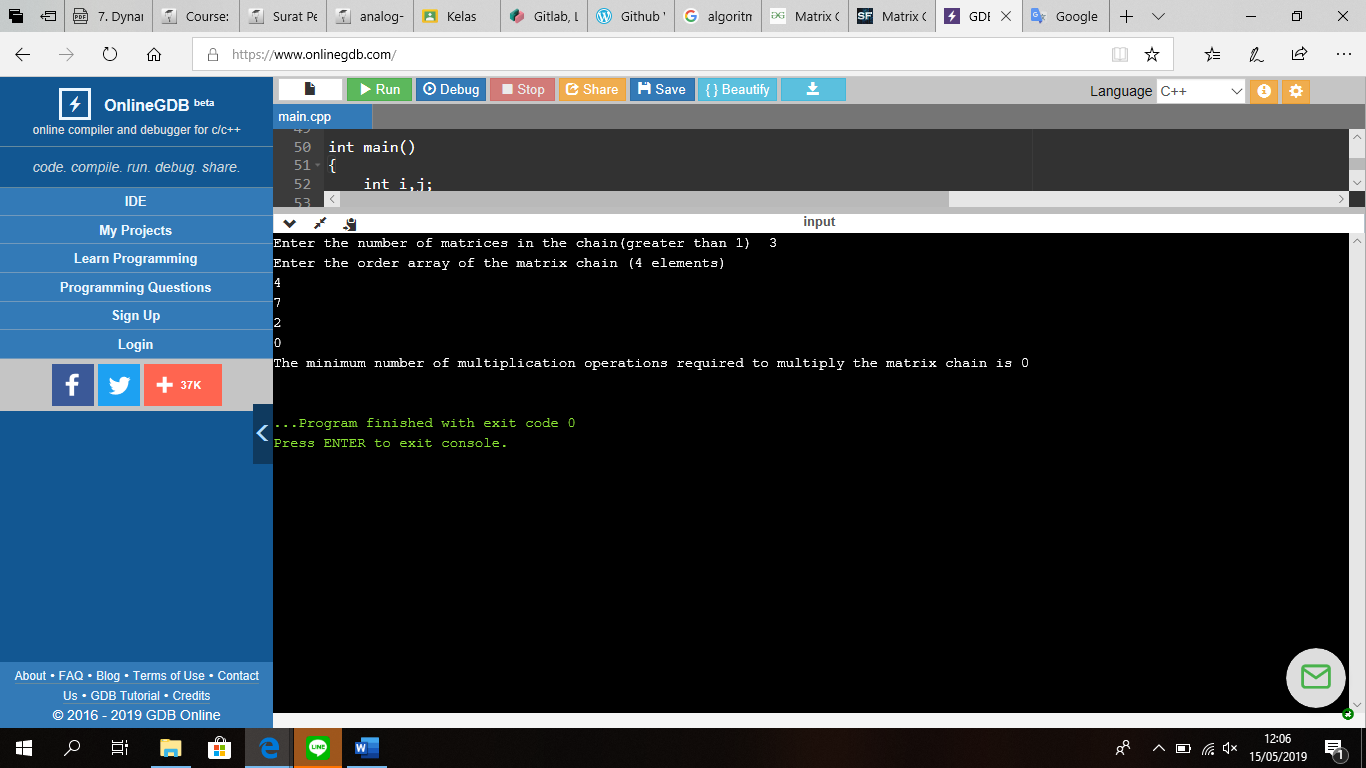
cout<<"The minimum number of multiplication operations required to multiply the matrix chain is "<<matrixChain(n,order);

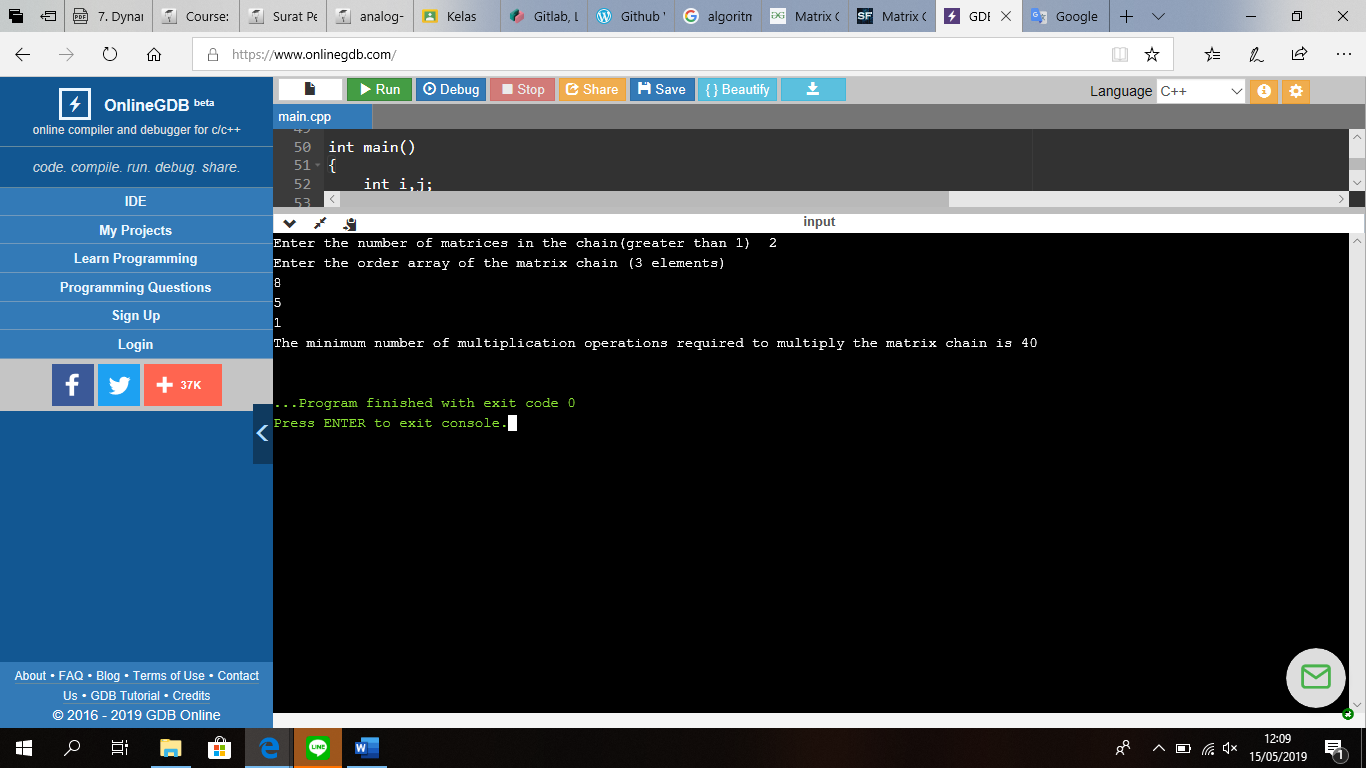
cout<<endl;

return 0;

}

**Screenshoot :**





1. Algo LCS

#include <bits/stdc++.h>

int max(int a, int b);

/\* Returns length of LCS for X[0..m-1], Y[0..n-1] \*/

int lcs(char\* X, char\* Y, int m, int n)

{

if (m == 0 || n == 0)

return 0;

if (X[m - 1] == Y[n - 1])

return 1 + lcs(X, Y, m - 1, n - 1);

else

return max(lcs(X, Y, m, n - 1), lcs(X, Y, m - 1, n));

}

/\* Utility function to get max of 2 integers \*/

int max(int a, int b)

{

return (a > b) ? a : b;

}

/\* Driver program to test above function \*/

int main()

{

char X[] = "RENDANG";

char Y[] = "NANGKA";

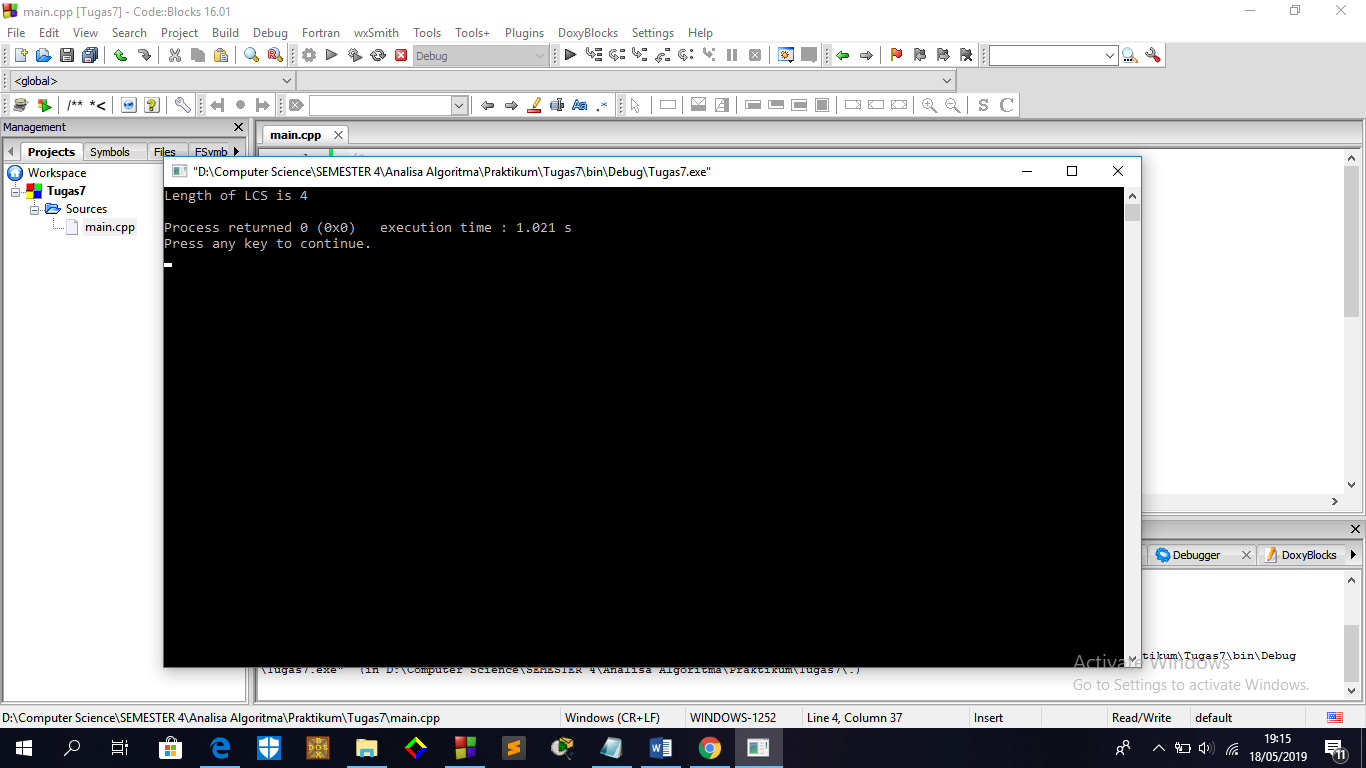
int m = strlen(X);

int n = strlen(Y);

printf("Length of LCS is %d\n", lcs(X, Y, m, n));

return 0;

}



1. Bandingkan

Perhitungan manual

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | j | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| i |  | Yj | r | e | n | d | a | n | g |
| 0 | Xi | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | n | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 2 | a | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 2 |
| 3 | n | 0 | 0 | 0 | 1 | 1 | 2 | 3 | 3 |
| 4 | g | 0 | 0 | 0 | 1 | 1 | 2 | 3 | 4 |
| 5 | k | 0 | 0 | 0 | 1 | 1 | 2 | 3 | 4 |
| 6 | a | 0 | 0 | 0 | 1 | 1 | 2 | 3 | 4 |
|  |  |  |  |  | n |  | a | n | g |

Perhitungan dengan program :

X : r e n d a n g

Y : n a n g k a

Huruf yang sama ada 4 yaitu n a n g

