**TOWER OF HANOI**

#include <stdio.h>

#include <time.h>

#include<stdlib.h>

void

towerOfHanoi (int n, char from\_rod, char to\_rod, char aux\_rod)

{

if (n == 1)

{

printf ("move disk from %c to %c\n", from\_rod, to\_rod);

return;

}

towerOfHanoi (n - 1, from\_rod, aux\_rod, to\_rod);

printf ("move disk from %c to %c\n", from\_rod, to\_rod);

towerOfHanoi (n - 1, aux\_rod, to\_rod, from\_rod);

}

int

main (int argc, const char \*argv[])

{

clock\_t start, stop;

int n;

printf ("Enter the number of plates :\n");

scanf ("%d", &n);

start = clock ();

towerOfHanoi (n, 'a', 'c', 'b');

stop = clock ();

double time = (double) (stop - start) / CLOCKS\_PER\_SEC;

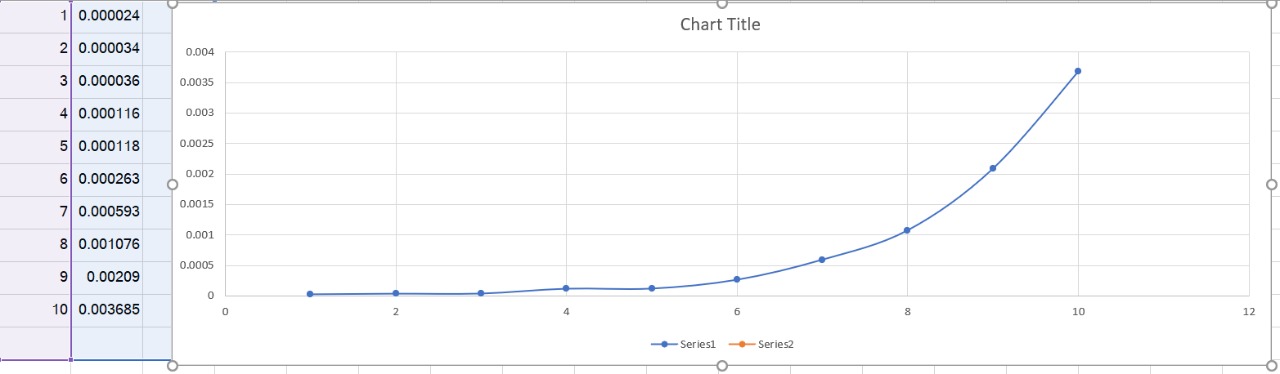
printf ("the exicution time of the program is %f seconds", time);

return 0;

}

Text

Description automatically generated



**DEPTH FIRST SEARCH**

#include <stdio.h>

#include <time.h>

#include<stdlib.h>

int a[20][20], reach[20], n, trav[20], c = 0;

void

dfs (int v)

{

int i;

trav[c++] = v;

reach[v] = 1;

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

if (a[v][i] && !reach[i])

dfs (i);

}

void

main ()

{

int i, j, count = 0;

clock\_t start, stop;

printf ("\n Enter number of vertices:");

scanf ("%d", &n);

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

{

reach[i] = 0;

for (j = 1; j <= n; j++)

a[i][j] = 0;

}

printf ("\n Enter the adjacency matrix:\n");

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

for (j = 1; j <= n; j++)

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

start = clock ();

dfs (1);

stop = clock ();

printf ("\n");

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

{

if (reach[i])

count++;

}

if (count == n){

printf ("\n Graph is connected\n");

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

printf ("%d\t", trav[i]);

}

else

printf ("\n Graph is not connected\n");

double time = (double) (stop - start) / CLOCKS\_PER\_SEC;

printf ("\n the exicution time of the program is %f seconds", time);

}

Text

Description automatically generated

