1BM19CS176

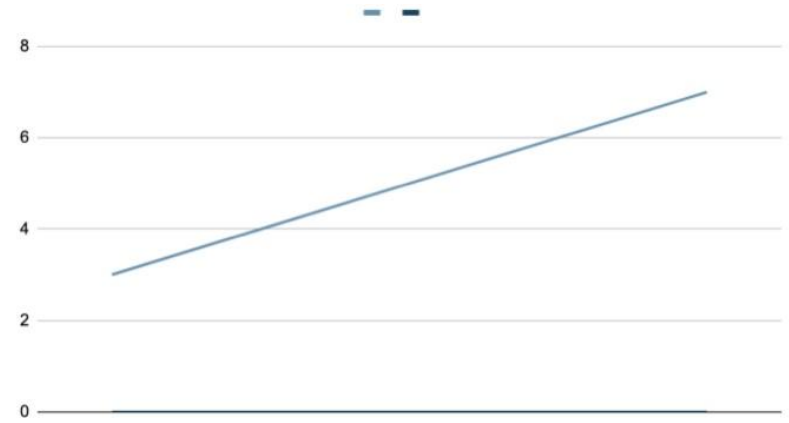
BFS

```
#include<stdio.h>
#include <time.h>
int a[20][20],q[20],visited[20],n,i,j,f=0,r=-1;

void bfs(int v)
{
   for(i = 1; i <= n; i++)
   if(a[v][i] && !visited[i])
   q[++r] = i;
   if(f <= r)
   {

      visited[q[f]] = 1;
      bfs(q[f++]);

   }

}

int main()
{
   int v;
   clock_t start, end;
   double t;
   printf("Enter the number of vertices: ");
   scanf("%d",&n);
   for(i=1; i <= n; i++)
   {

      q[i] = 0;

      visited[i] = 0;

   }

   printf("\nEnter graph data in matrix form:\n");
   for(i=1; i<=n; i++)
   {
      for(j=1;j<=n;j++)
      {
         scanf("%d", &a[i][j]);
      }
```

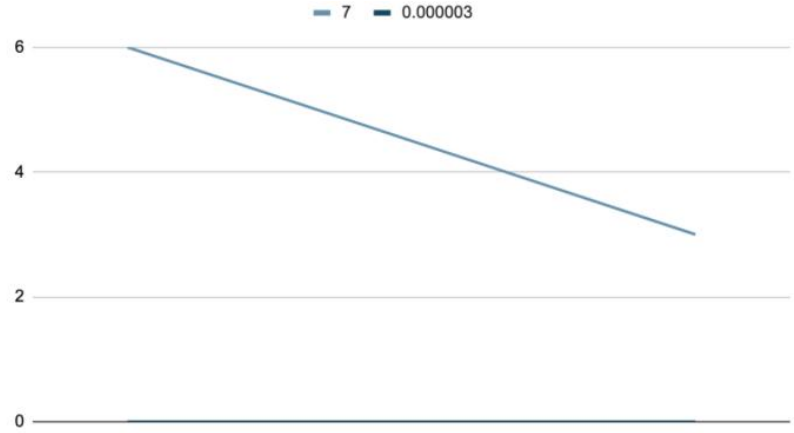
```
    }

    printf("Enter the starting vertex: ");
    scanf("%d", &v);
    bfs(v);
    printf("\nThe node which are reachable are:");
    for(i=1; i <= n; i++)
    {
        if(visited[i])
        printf(" %d", i);
        else
        {
            printf("\nBFS is not possible. All nodes are not reachable!");
            break;
        }

    }
    start = clock();
    bfs(v);
    end = clock();
    t = ((double) (end - start)) / CLOCKS_PER_SEC;
    printf("\n");
    printf("\nTime taken by BFS : %lf\n", t);
    printf("\n");
    return 0;
}
```

```
Enter the number of vertices: 3

Enter graph data in matrix form:
1 1 1
0 1 0
0 1 1
Enter the starting vertex: 1

The node which are reachable are: 1 2 3

Time taken by BFS : 0.000002
```

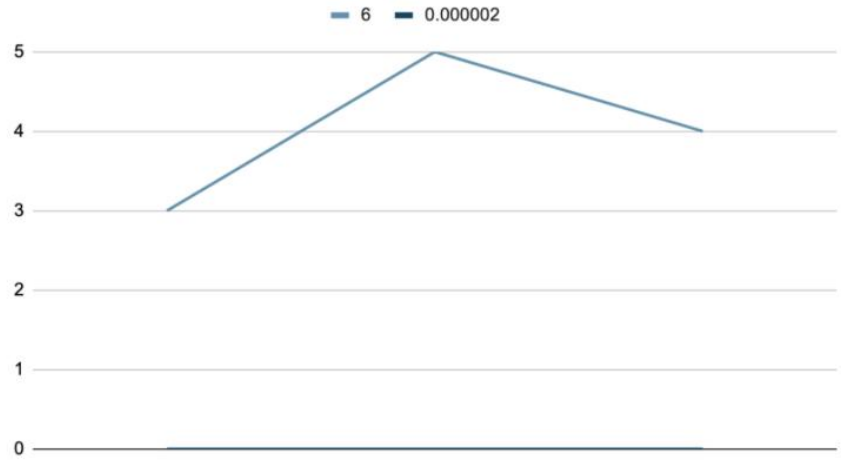
Graphs-
BFS(ascending order):



BFS(decending order):



BFS(random order):



DFS:

```
#include <stdio.h>
#include <time.h>
void dfs(int n, int cost[10][10], int u, int s[])
{
int v;
```

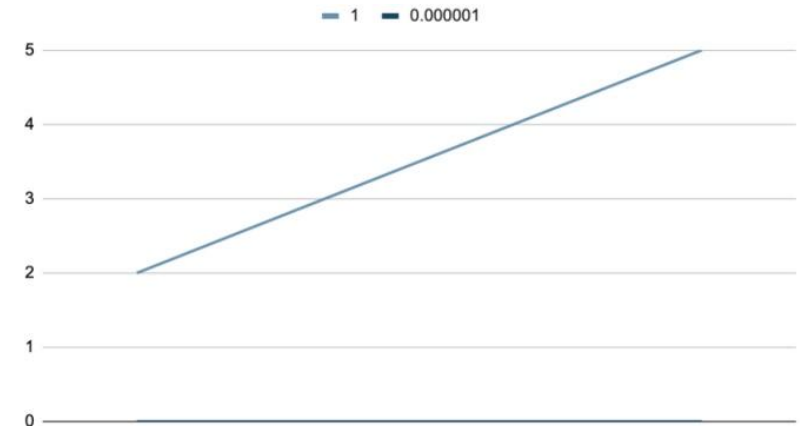
```
s[u]=1;
for(v=0;v<n;v++)
{
if((cost[u][v]==1) && (s[v]==0))
dfs(n,cost,v,s);
}
}
int main()
{
int n,i,j,cost[10][10],s[10],con,flag;
clock_t start, end;
double t;
printf("Enter the number of nodes\n");
scanf("%d",&n);
printf("Enter the adjacency matrix\n");
for(i=0;i<n;i++)
{
for(j=0;j<n;j++)
scanf("%d",&cost[i][j]);
}
con=0;
for(j=0;j<n;j++)
{
for(i=0;i<n;i++)
s[i]=0;
dfs(n,cost,j,s);
flag=0;
for(i=0;i<n;i++)
{
if(s[i]==0)
flag=1;

}
if(flag==0)
con=1;

}
if(con==1)
printf("\nGraph is connected\n");
else
printf("\nGraph is not connected\n");
start = clock();
 dfs(n,cost,j,s);
 end = clock();
```

```
 t = ((double) (end - start)) / CLOCKS_PER_SEC;
 printf("\n");
 printf("\nTime taken by DFS : %lf\n", t);
 printf("\n");
return 0;
}
```

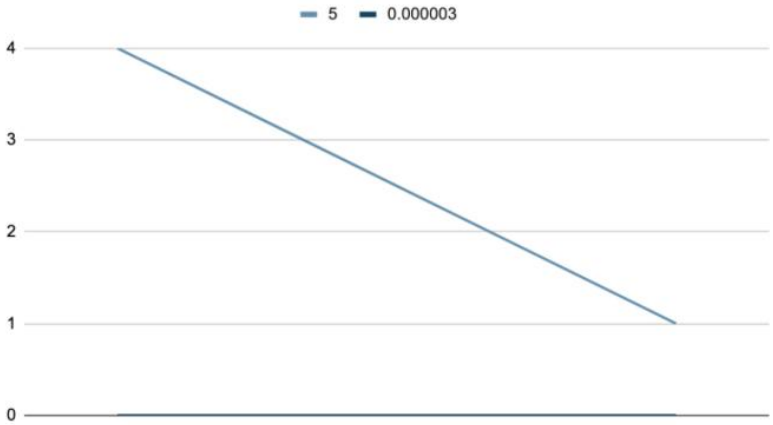
```
Enter the number of nodes
4
Enter the adjacency matrix
0 1 0 0
0 0 1 0
0 0 0 1
1 0 0 0

Graph is connected


Time taken by DFS : 0.000002
```

```
Enter the number of nodes
2
Enter the adjacency matrix
1 2
2 3

Graph is not connected


Time taken by DFS : 0.000002
```

Graphs-
DFS(ascending order):



DFS(descending order):



DFS(random order):