Competitive Programming

(https://github.com/MalavikaJayakumar/Competitive-Programming-Problems)

1. **DFS**
   a. **Store your graph using adjacency matrix(as we have done in the lab session)**
   b. **Write separate functions for the important operations(so that u can reuse those functions in the next programs too)**
   c. **DFS() should visit the nodes in the correct order and print them based on the increasing order of their Finishing time**

Code:

```
# include<iostream>
using namespace std;

int n,e;

int printgraph(int *m)
{
	int i,j;
	cout<<"\n Adjacency graph: \n";
	for(i=0;i<n;i++)
	{
		for(j=0;j<n;j++)
		{
			cout<<*((m+(i*n))+j)<<" ";
		}
		cout<<endl;
	}
	return 0;
}

int readgraph(int *m)
{

	int x,y,i;
	cout<<"\n Enter the starting and ending nodes: ";
	for(i=0;i<e;i++)
	{
		cin>>x>>y;
		*((m+(x*n))+y)=1;
		*((m+(y*n))+x)=1;
	}
	return 0;
}

int dfs(int *m,int v)
{
	int j,k,stk[n],top,visit[n],visited[n];
	cout<<"\n DFS order of nodes: ";
	cout<<v<< " ";
	visited[v]=1;
```

```
	k=1;
	while(k<n)
	{
		for(j=(n-1);j>=0;j--)
			if((*((m+(v*n))+j))!=0 && visited[j]!=1 && visit[j]!=1)
			{
				visit[j]=1;
				stk[top]=j;
				top++;
			}
		v=stk[--top];
		cout<<v<<" ";
		k++;
		visit[v]=0;
		visited[v]=1;
	}
	return 0;
}

int main()
{
	int v;
	cout<<"Enter number of vertices: ";
	cin>>n;
	cout<<"\n Enter number of edges: ";
	cin>>e;
	int m[n][n]={};
	readgraph((int *)m);
	printgraph((int *)m);
	cout<<"\n Enter start node of dfs: ";
	cin>>v;
	dfs((int *)m,v);
	return 0;
}
```

Output:

```
Enter number of vertices: 4

 Enter number of edges: 4

 Enter the starting and ending nodes: 0 1
0 2
1 3
2 3

 Adjacency graph:
0 1 1 0
1 0 0 1
1 0 0 1
0 1 1 0

 Enter start node of dfs: 0

 DFS order of nodes: 0 1 3 2

------------------
```