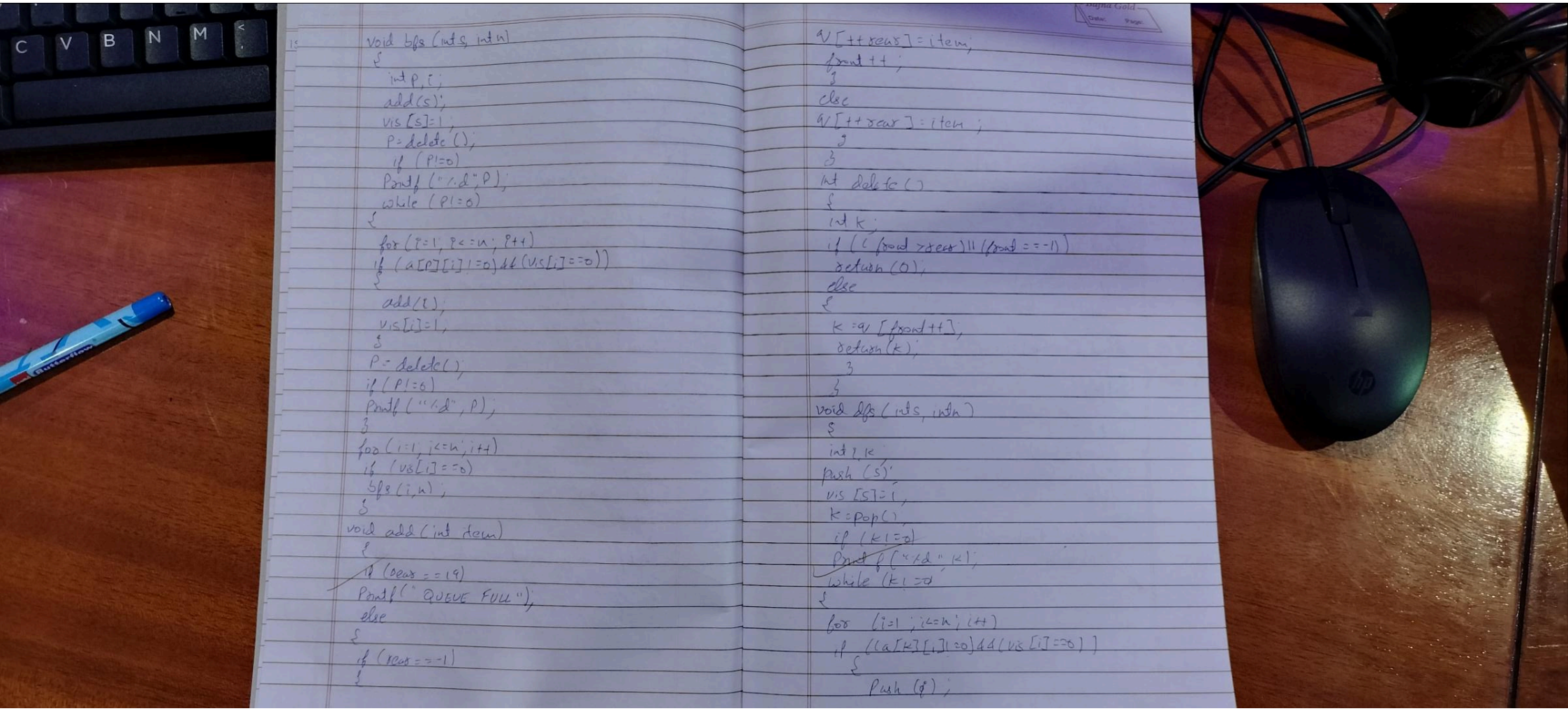
15/6/23

```
#include<stdio.h>
int q[20], top=-1, front=-1, rear=-1, a[20][20],
    vis[20], stack[20];
int delete();
void add(int item);
void bfs(int s, int n);
void dfs(int s, int n);
void push(int item);
int pop();

void main()
{
 int n, i, s, ch, j;
 char c, dummy;
 Printf("Enter the number vertices");
 Scanf("%d", &n);
 for(i=1; i<=n; i++)
  {
  for(j=1; j<=n; j++)
  {
 Printf("Enter 1 if %d has a node with %d else
     0", i, j);
 Scanf("%d", &a[i][j]);
  }
 }
 Printf("ADJACENCY MATRIX IS\n");
  for(i=1; i<=n; i++)
  {
   for(j=1; j<=n; j++)
   Printf("%d", a[i][j]);
  }
  Printf("\n");
  }
  do
  {
  for(i=1, i<=n; i++)
  vis[i]=0;
  Printf("\n MENU");
  Printf("\n1.B.F.S");
  Printf("\n2.D.F.S");
  Printf("\nEnter your choice");
  Scanf("%d", &ch);
  Printf("Enter the source vertex");
  Scanf("%d", &s);

  Switch(ch)
  {
  case 1: bfs(s, n);
  break;
  case 2:
  dfs(s, n);
  break;
  }
  Printf("Do you want to continue (Y/N)?");
  Scanf("%c", &dummy);
  Scanf("%c", &c);
  } while((c=='y')||(c=='Y'));
  }
```



```
void bfs (int s, int n)
{
  int p, i;
  add(s);
  vis[s]=1;
  p=delete();
  if (p!=0)
  Printf ("%d",p);
  while (p!=0)
  {
  for(i=1; i<=n; i++)
  if (a[p][i]!=0)&&(vis[i]==0))
  {
  add(i);
  vis[i]=1;
  }
  p=delete();
  if(p!=0)
  Printf("%d", p);
  }
  for(i=1; i<=n; i++)
  if (vis[i]==0)
  bfs(i,n);
  }
void add(int item)
  {
  if (rear==19)
  Printf(" QUEUE FULL");
  else
  {
  if (rear==-1)
  {
  q[++rear]=item;
  front++;
  }
  else
  q[++rear]=item;
  }
  }
int delete()
  {
  int k;
  if ((front>rear)||(front==-1))
  return(0);
  else
  {
  k=q[front++];
  return(k);
  }
  }
void bfs(int s, int n)
  {
  int p, k;
  push(s);
  vis[s]=1;
  k=pop();
  if(k!=0)
  Printf("%d", k);
  while(k!=0)
  {
  for (i=1; i<=n; i++)
  if ((a[k][i]!=0)&&(vis[i]==0))
  {
  push(i);
```

```
vis[i]=1;
 }
k=pop();
if(k!=0)
Printf("%d", k);
 }
for(i=1;i<n;i++)
 if(vis[i]==0)
 dfs(i,n);
}
void push(int item)
{
 if(top==19)
 Printf("Stack overflow");
 else
stack[++top]=item;
 }
int pop()
{
int k;
if(top==-1)
return(0);
else
{
k=stack[top--];
return(k);
 }
}
```

Output

V_1 V_2

Enter Verti. 2 1

0 0
0 0
0 1
0 0
0 0
0 0
0 0
0 0
0 0
0 1
0 0

Adjaccency matrix is

0 1 1 1
0 0 0 1
0 0 0 0
0 0 1 0

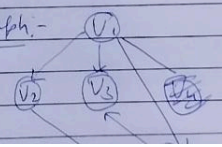
MENU

1 BFS
2 DFS

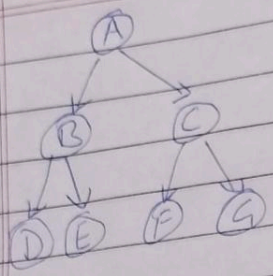
BFS = 1 4 3 2

DFS = 1 2 3 4

Graph:-



=>



BFS - A B C D E F G

DFS - A B D E C F G

	A	B	C	D	E	F	G
A	0	1	1	0	0	0	0
B	0	0	0	1	1	0	0
C	0	0	0	0	0	1	1
D	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0
F	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0

BFS

Source Vertex - 1

1 2 3 4 5 6 7

DFS :

1 2 4 5 3 6 7

15/9/23