BFS Traversal

```
#include< stdio.h>
#include< math.h>
int n;
int adj[10][10]
int visited [50]
int q [20]
int front =-1;
int rear =-1;
Void enque(int v)
=> {
        if (front==0 && rear=n-1)
        {
            printf ("Queue Full\n");
        }
    else if (front==-1 && rear==-1)
      {
          ~~p.~~ front = rear=0;
      }
    else
        rear++;
   q[rear]=v
}
```

```
int  dequeue()
{  int val
   if (front == -1 || front > rear)
   {
        front = -1;
        return -1;
   }
   val = q[front]
    if (front == rear || front > rear)
    {
        front = -1;
         rear = -1;
    }
    else
     {
        front ++;
     }
    return val
}
```

```
void bfs (int v]
 {
     for (int i=0; i<n;i++)
      {
         if [adj[v][i]==1 && visted[i]..
         }
           enqueue[i]
           printf("%d", i);
           visited[i]=1'
          }
      }
     int val=dequeue()
     if (val!=-1)
        bfs (i)
  }
int main()
 {
     int flag=1, v;
     printf ("Enter Number of vertex\n")
     Scanf ("%d", &n);
     printf ("Enter adjacent matrix\n")
      for (int i=0; i< n;i++)
          {
              for(int j=0; j<n;j++)
                    Scan ("%d", &a[i][j]
          }
     printf ("Enter start vertex\n")
     Scanf ("%d", &v);
     printf ("FOREST1 \n");
      printf ("%d\t", v);
      visited [v]=1
```

```
bfs(v)
printf ("FOREST2\n");
for (int i = 0; i < n; i++)
{
    if (visited[i] = = 0)
    {
        flag = 0;
        printf ("%d\t", i)
        visited[i] = 1
        bfs(i)
        break;
    }   }

if (flag == 1)
{
    printf (" GRAPH CONNECTED
}
}
```