S1 DATA STRUCTURE EXTERNAL LAB EXAM 2021

SUBMITTED BY

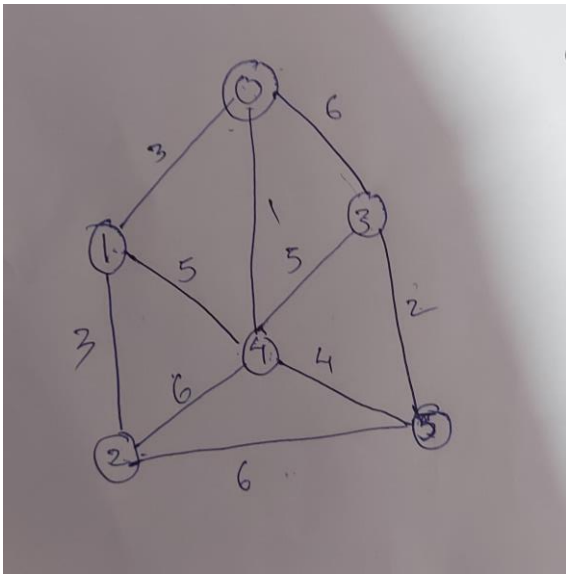
Robin Monachan

TKM20MCA-2032

QUESTIONS

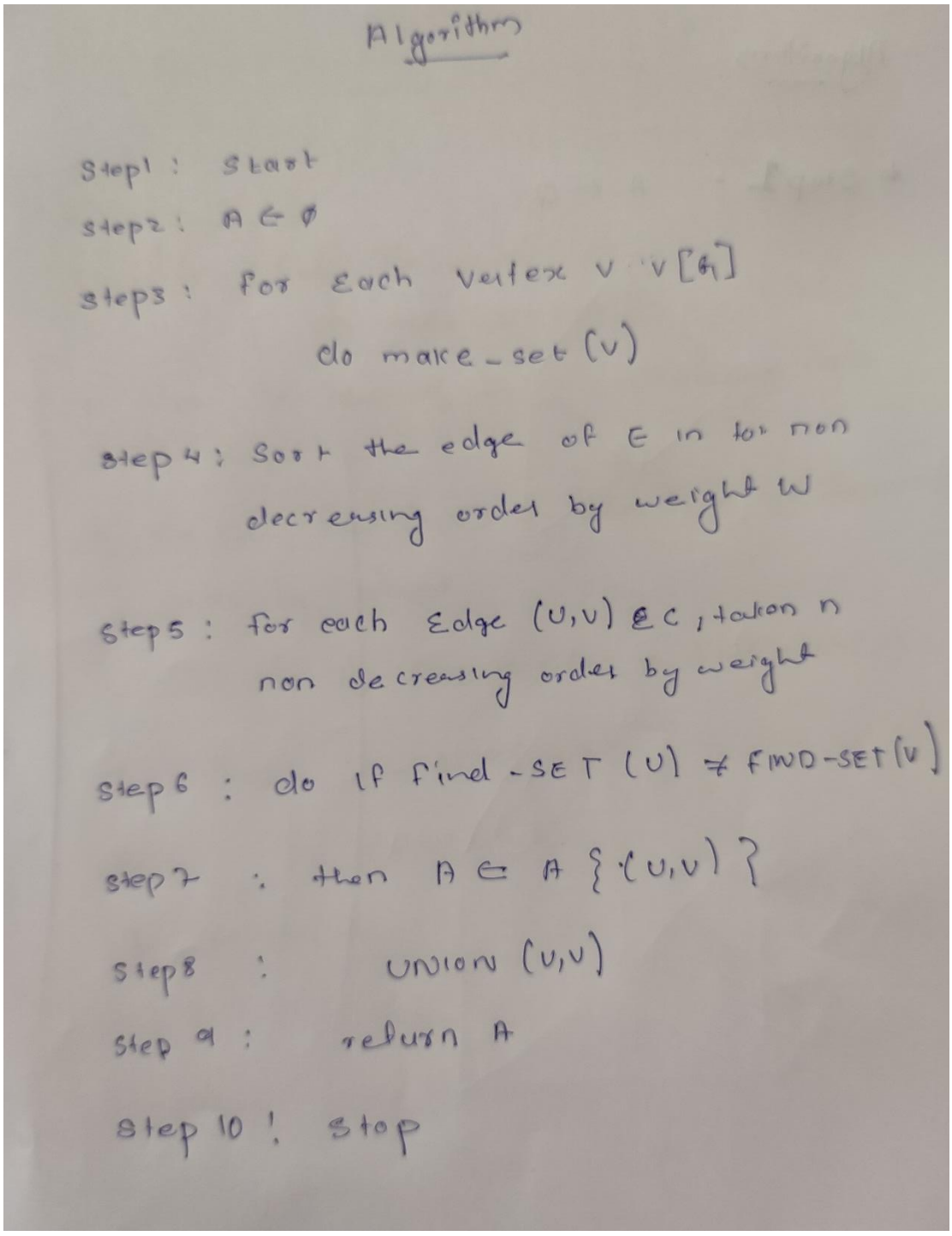
1.DEVELOP A PROGRAM TO GENERATE A MINIMUM SPANNING TREE USING KRUSKAL ALGORITHM FOR THE GIVEN GRAPH AND COMPUTE THE TOTAL COST

2. DEVELOP A PROGRAM TO IMPLEMENT BFS AND DFS



1. DEVELOP A PROGRAM TO GENERATE A MINIMUM SPANNING TREE USING KRUSKAL ALGORITHM FOR THE GIVEN GRAPH AND COMPUTE THE TOTAL COST

Algorithm



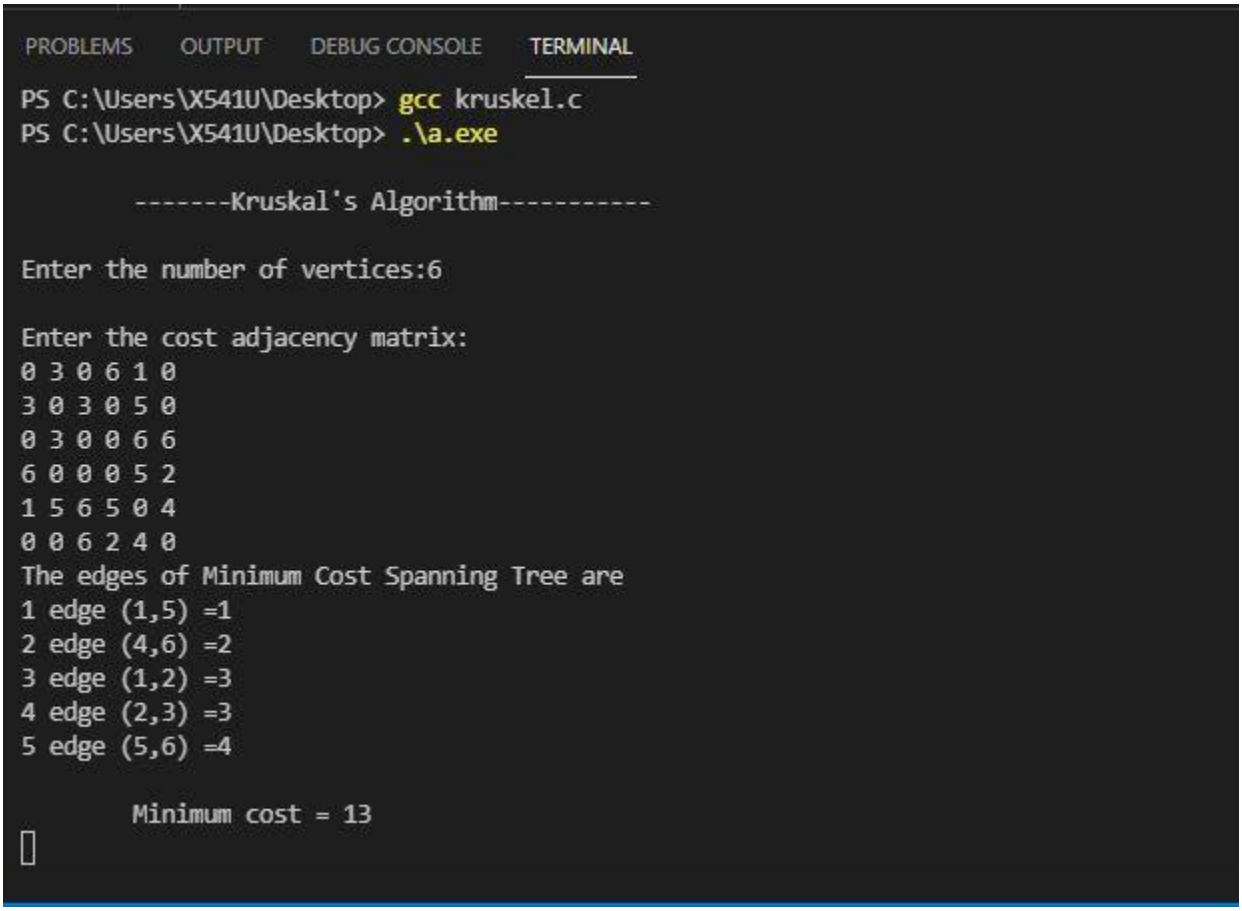
Program

```
#include <stdio.h>
   #include <conio.h>
   #include <stdlib.h>
   int i,j,k,a,b,u,v,n,ne=1;
   int min,mincost=0,cost[9][9],parent[9];
   int find(int);
   int uni(int,int);
   void main()
   {
          printf("\n\t-------Kruskal's Algorithm----------\n");
          printf("\nEnter the number of vertices:");
          scanf("%d",&n);
          printf("\nEnter the cost adjacency matrix:\n");
          for(i=1;i<=n;i++)
          {
          for(j=1;j<=n;j++)
          {
          scanf("%d",&cost[i][j]);
          if(cost[i][j]==0)
          cost[i][j]=999;
          }
          }
          printf("The edges of Minimum Cost Spanning Tree are\n");
          while(ne < n)
          {
          for(i=1,min=999;i<=n;i++)
```

```
        {
        for(j=1;j <= n;j++)
        {
        if(cost[i][j] < min)
        {
        min=cost[i][j];
        a=u=i;
        b=v=j;
        }
        }
        }
        u=find(u);
        v=find(v);
        if(uni(u,v))
        {
        printf("%d edge (%d,%d) =%d\n",ne++,a,b,min);
        mincost +=min;
        }
        cost[a][b]=cost[b][a]=999;
        }
        printf("\n\tMinimum cost = %d\n",mincost);
        getch();
}
int find(int i)
{
        while(parent[i])
        i=parent[i];
        return i;
}
```

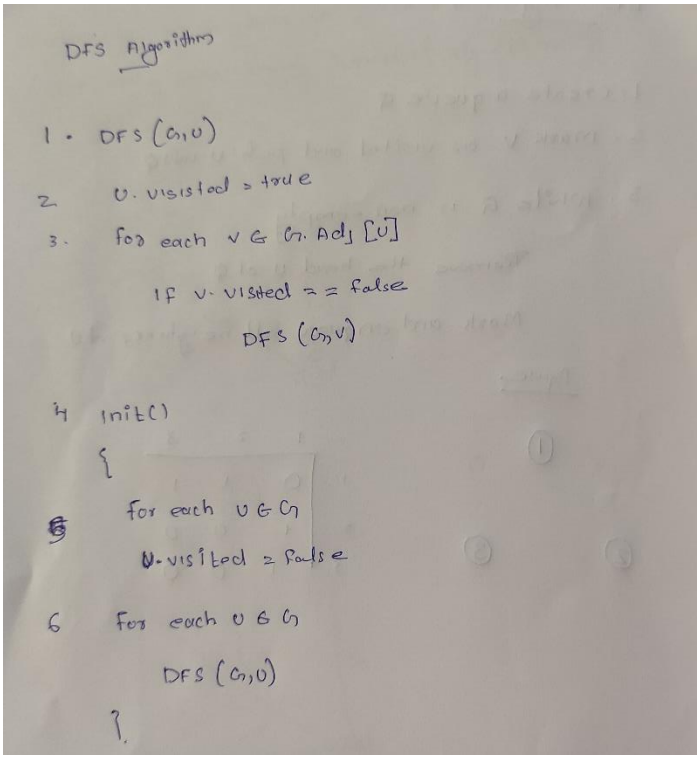
```
int uni(int i,int j)
{
        if(i!=j)
        {
        parent[j]=i;
        return 1;
        }
        return 0;
}
```

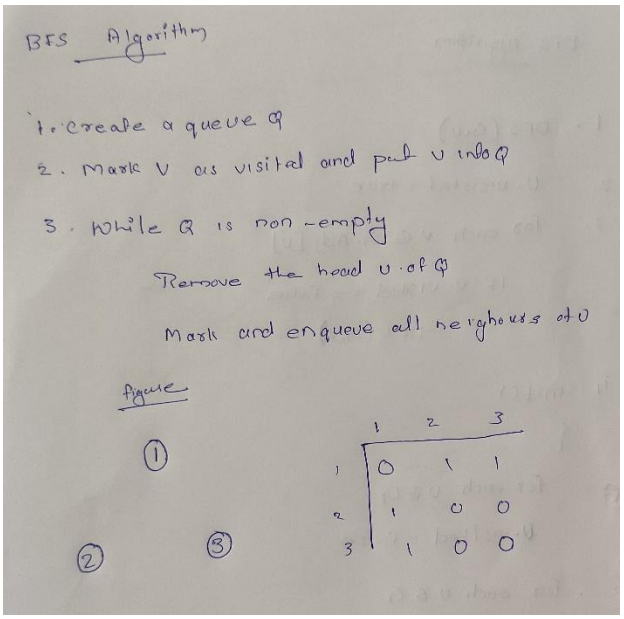
OUTPUT


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PS C:\Users\X541U\Desktop> gcc kruskel.c
PS C:\Users\X541U\Desktop> .\a.exe
-------Kruskal's Algorithm-----------
Enter the number of vertices:6
Enter the cost adjacency matrix:
0 3 0 6 1 0
3 0 3 0 5 0
0 3 0 0 6 6
6 0 0 0 5 2
1 5 6 5 0 4
0 0 6 2 4 0
The edges of Minimum Cost Spanning Tree are
1 edge (1,5) =1
2 edge (4,6) =2
3 edge (1,2) =3
4 edge (2,3) =3
5 edge (5,6) =4
Minimum cost = 13

2. DEVELOP A PROGRAM TO IMPLEMENT BFS AND DFS

Algorithm





Program

```
#include<stdio.h>
#include <stdlib.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 of 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 ***********\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("\nMENU");
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 U 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 dfs(int s,int n)
{
int i,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

```
PS C:\Users\X541U\Desktop\ds> gcc dfsbfs.c
PS C:\Users\X541U\Desktop\ds> .\a.exe
Enter the number of vertices: 3
Enter 1 if 1 has a node with 1 else 0: 0
Enter 1 if 1 has a node with 2 else 0: 1
Enter 1 if 1 has a node with 3 else 0: 1
Enter 1 if 2 has a node with 1 else 0: 1
Enter 1 if 2 has a node with 2 else 0: 0
Enter 1 if 2 has a node with 3 else 0: 0
Enter 1 if 3 has a node with 1 else 0: 1
Enter 1 if 3 has a node with 2 else 0: 0
Enter 1 if 3 has a node with 3 else 0: 0
********* ADJACENCY MATRIX ************
 0 1 1
 1 0 0
 1 0 0

MENU
1.B.F.S
2.D.F.S
ENTER YOUR CHOICE1
ENTER THE SOURCE VERTEX :1
 1 2  3
DO U WANT TO CONTINUE(Y/N) ? y

MENU
1.B.F.S
2.D.F.S
ENTER YOUR CHOICE2
ENTER THE SOURCE VERTEX :1
 1  3  2
DO U WANT TO CONTINUE(Y/N) ? n
PS C:\Users\X541U\Desktop\ds>
```