CSE 2003

DATA STRUCTURES AND ALGORITHMS

Lab FAT

L19+L20 | SJT317

FALL SEMESTER 2020-21

by

SHARADINDU ADHIKARI

19BCE2105

Question 1

Problem:

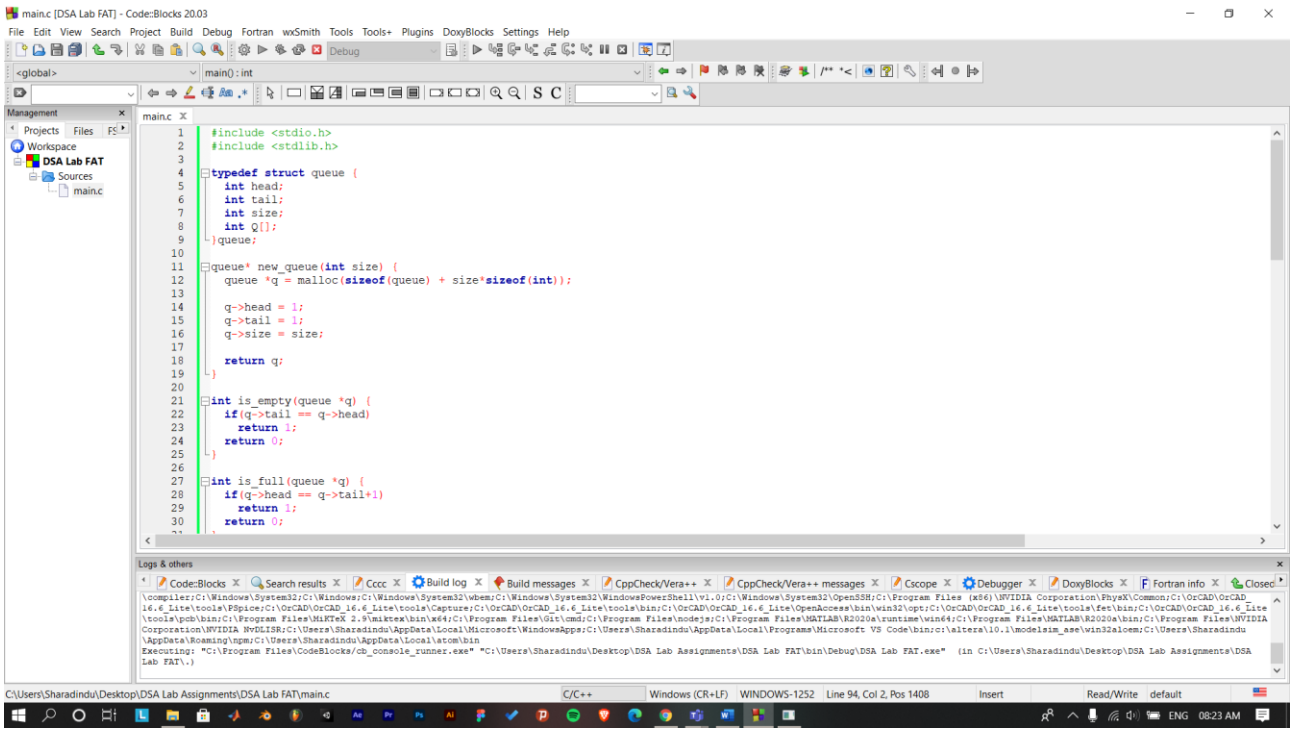
Write a program to implement a queue by using an array and linked list.

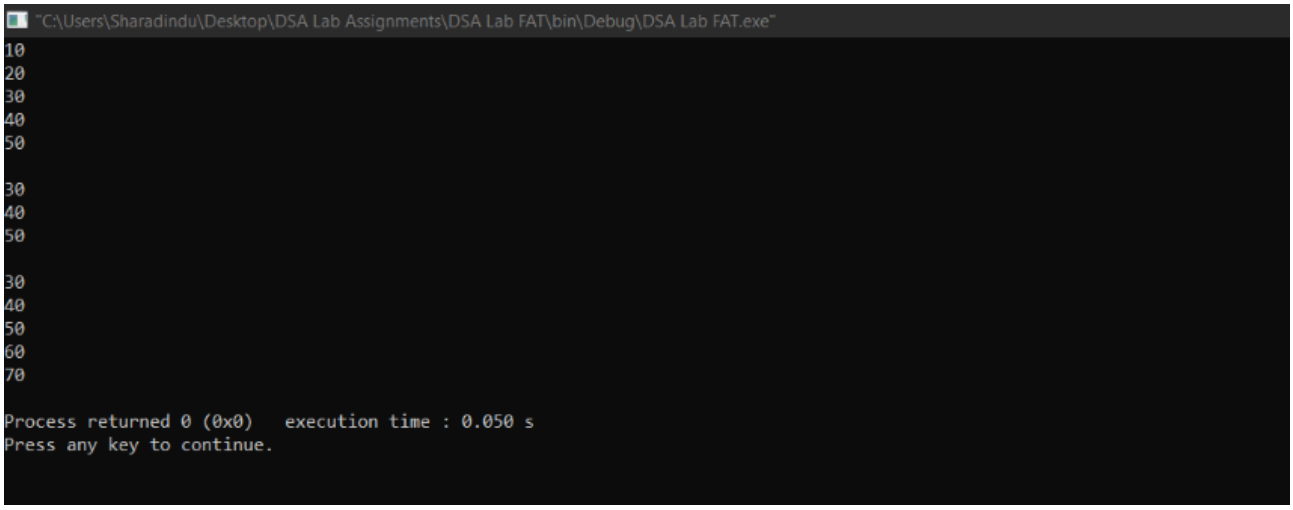
Code, SS, & Output in CMD:

Queue using Array:

```
#include <stdio.h>
#include <stdlib.h>

typedef struct queue {
  int head;
  int tail;
  int size;
  int Q[];
}queue;

queue* new_queue(int size) {
  queue *q = malloc(sizeof(queue) + size*sizeof(int));

  q->head = 1;
  q->tail = 1;
  q->size = size;

  return q;
}

int is_empty(queue *q) {
  if(q->tail == q->head)
    return 1;
  return 0;
}

int is_full(queue *q) {
  if(q->head == q->tail+1)
    return 1;
  return 0;
}

void enqueue(queue *q, int x) {
  if(is_full(q)) {
    printf("Queue Overflow\n");
  }
  else {
    q->Q[q->tail] = x;
    if(q->tail == q->size)
      q->tail = 1;
    else
      q->tail = q->tail+1;
  }
}
```

```
int dequeue(queue *q) {
  if(is_empty(q)) {
    printf("Underflow\n");
    return -1000;
  }
  else {
    int x = q->Q[q->head];
    if(q->head == q->size) {
      q->head = 1;
    }
    else {
      q->head = q->head+1;
    }
    return x;
  }
}


void display(queue *q) {
  int i;
  for(i=q->head; i<q->tail; i++) {
    printf("%d\n",q->Q[i]);
    if(i == q->size) {
      i = 0;
    }
  }
}


int main() {
  queue *q = new_queue(10);
  enqueue(q, 10);
  enqueue(q, 20);
  enqueue(q, 30);
  enqueue(q, 40);
  enqueue(q, 50);
  display(q);

  printf("\n");

  dequeue(q);
  dequeue(q);
  display(q);

  printf("\n");

  enqueue(q, 60);
  enqueue(q, 70);
  display(q);
  return 0;
}
```



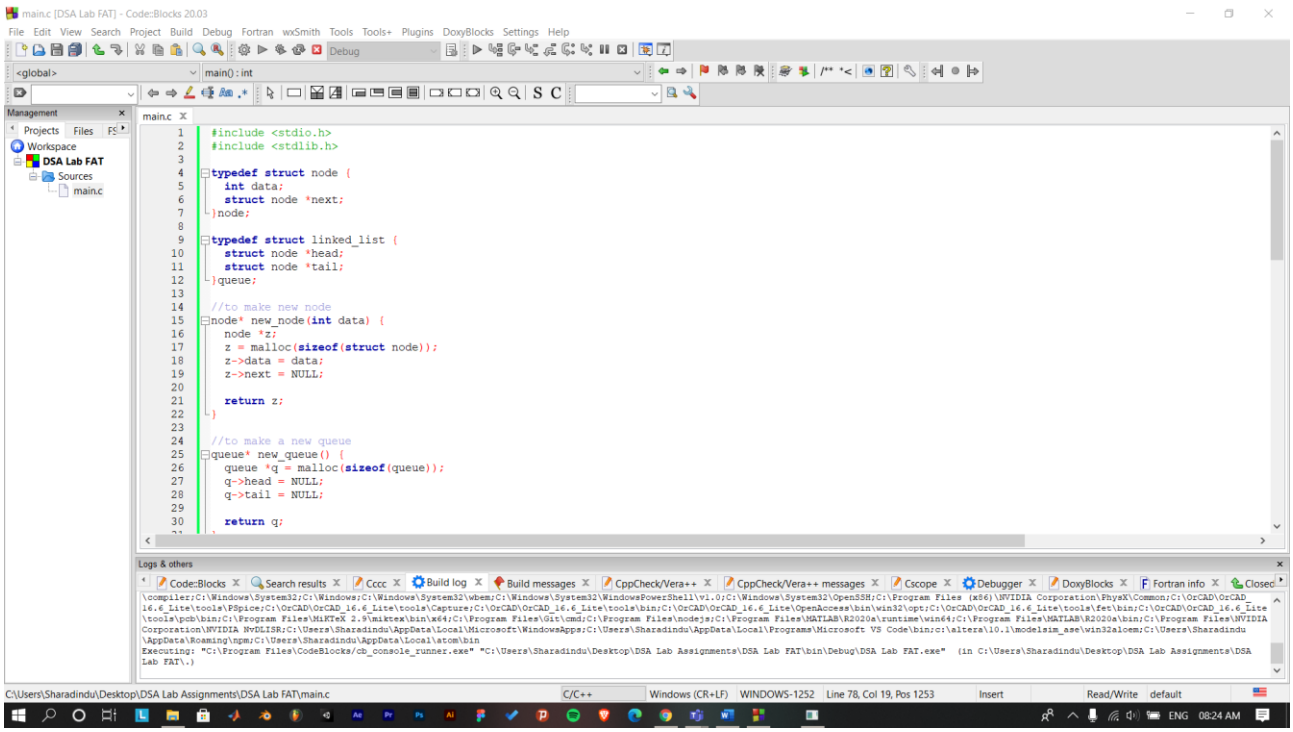


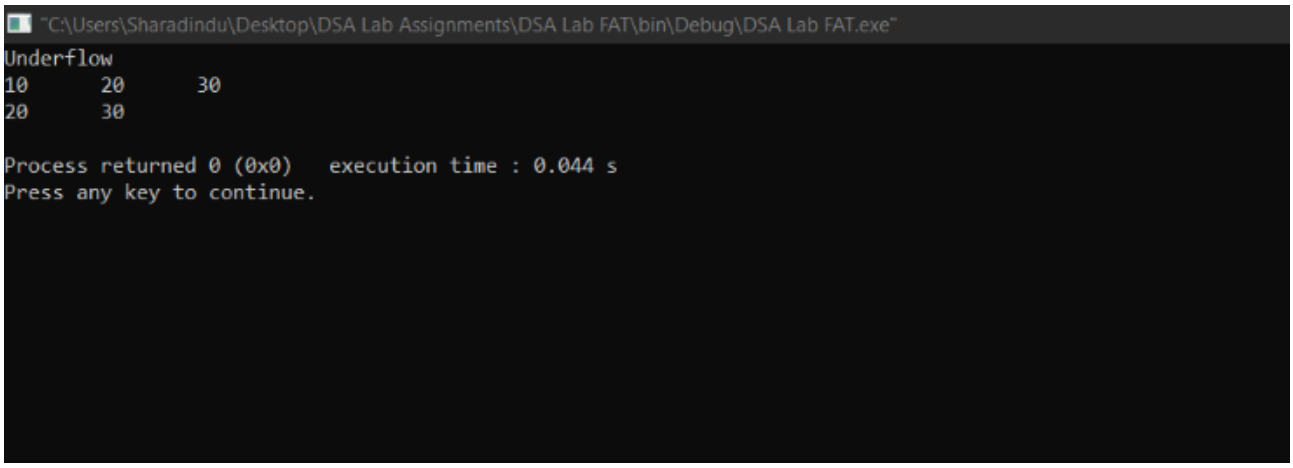
Queue using Linked List:

```
#include <stdio.h>
#include <stdlib.h>

typedef struct node {
  int data;
  struct node *next;
}node;

typedef struct linked_list {
  struct node *head;
  struct node *tail;
}queue;
```

```
//to make new node
node* new_node(int data) {
  node *z;
  z = malloc(sizeof(struct node));
  z->data = data;
  z->next = NULL;

  return z;
}

//to make a new queue
queue* new_queue() {
  queue *q = malloc(sizeof(queue));
  q->head = NULL;
  q->tail = NULL;

  return q;
}

void traversal(queue *q) {
  node *temp = q->head; //temporary pointer to point to head

  while(temp != NULL) { //iterating over queue
    printf("%d\t", temp->data);
    temp = temp->next;
  }

  printf("\n");
}

int is_empty(queue *q) {
  if(q->head == NULL)
    return 1;
  return 0;
}

void enqueue(queue *q, node *n) {
  if(is_empty(q)) {
    q->head = n;
    q->tail = n;
  }
  else {
    q->tail->next = n;
    q->tail = n;
  }
}

int dequeue(queue *q) {
  if(is_empty(q)) {
    printf("Underflow\n");
    return -1000;
  }
  else {
    int x = q->head->data;
    node *temp = q->head;
    q->head = q->head->next;
    free(temp);
    return x;
  }
}

int main() {
  queue *q = new_queue();
```

```
    node *a, *b, *c;
    a = new_node(10);
    b = new_node(20);
    c = new_node(30);

    dequeue(q);
    enqueue(q, a);
    enqueue(q, b);
    enqueue(q, c);

    traversal(q);
    dequeue(q);
    traversal(q);


    return 0;
}
```





Question 2

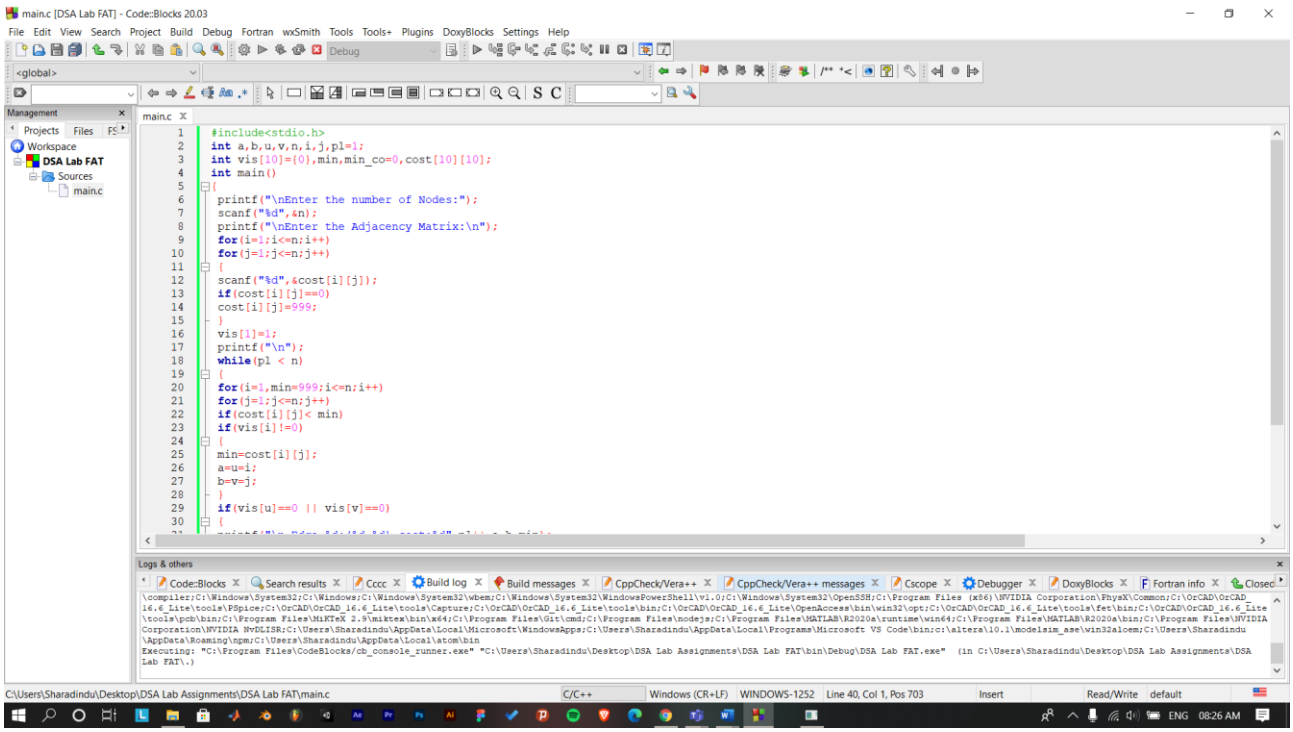
Problem:

Write a Program to implement of Prim's and Kruskal's algorithm.

Code, SS, & Output in CMD:

Prim's algorithm:

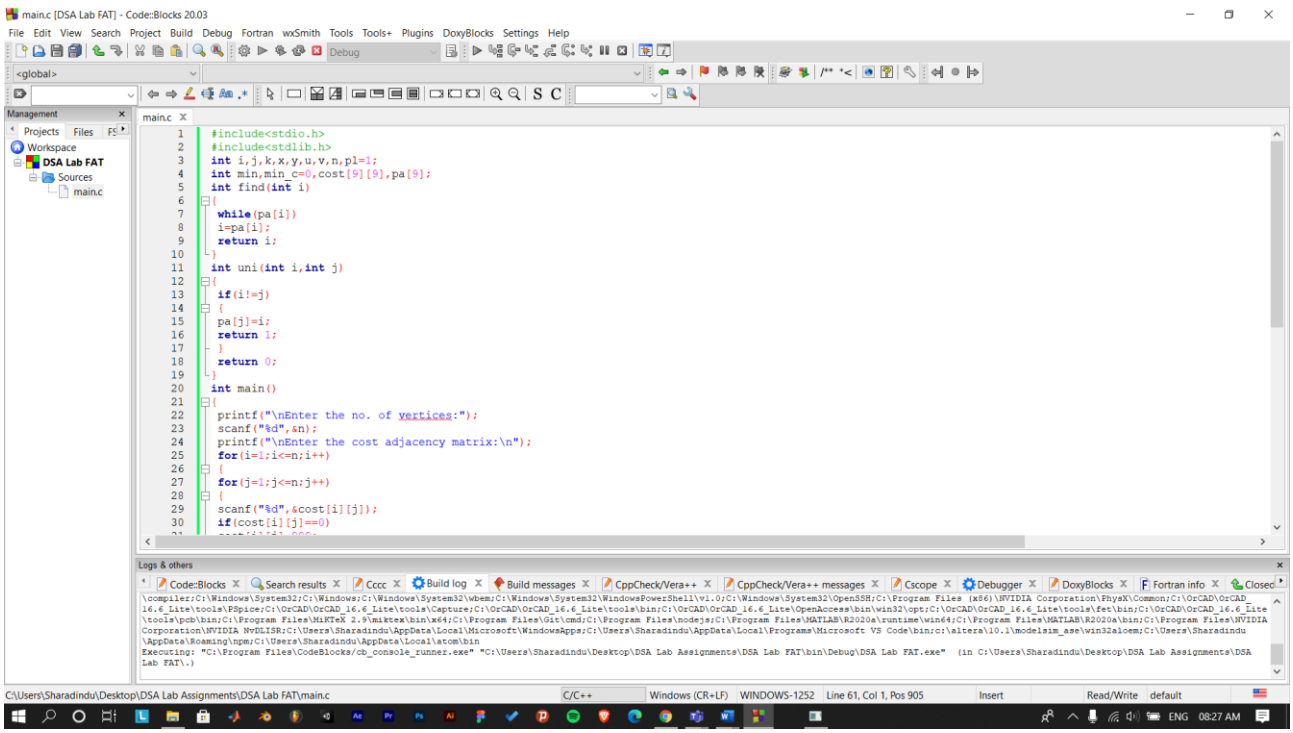
```
#include<stdio.h>
int a,b,u,v,n,i,j,pl=1;
int vis[10]={0},min,min_co=0,cost[10][10];
int main()
{
 printf("\nEnter the number of Nodes:");
 scanf("%d",&n);
 printf("\nEnter the Adjacency Matrix:\n");
 for(i=1;i<=n;i++)
 for(j=1;j<=n;j++)
 {
 scanf("%d",&cost[i][j]);
 if(cost[i][j]==0)
 cost[i][j]=999;
 }
 vis[1]=1;
 printf("\n");
 while(pl < n)
 {
 for(i=1,min=999;i<=n;i++)
 for(j=1;j<=n;j++)
 if(cost[i][j]< min)
 if(vis[i]!=0)
 {
 min=cost[i][j];
 a=u=i;
 b=v=j;
 }
 if(vis[u]==0 || vis[v]==0)
 {
 printf("\n Edge %d:(%d %d) cost:%d",pl++,a,b,min);
 min_co+=min;
 vis[b]=1;
 }
 cost[a][b]=cost[b][a]=999;
 }
 printf("\n Minimun cost=%d",min_co);
 return 0;
}
```

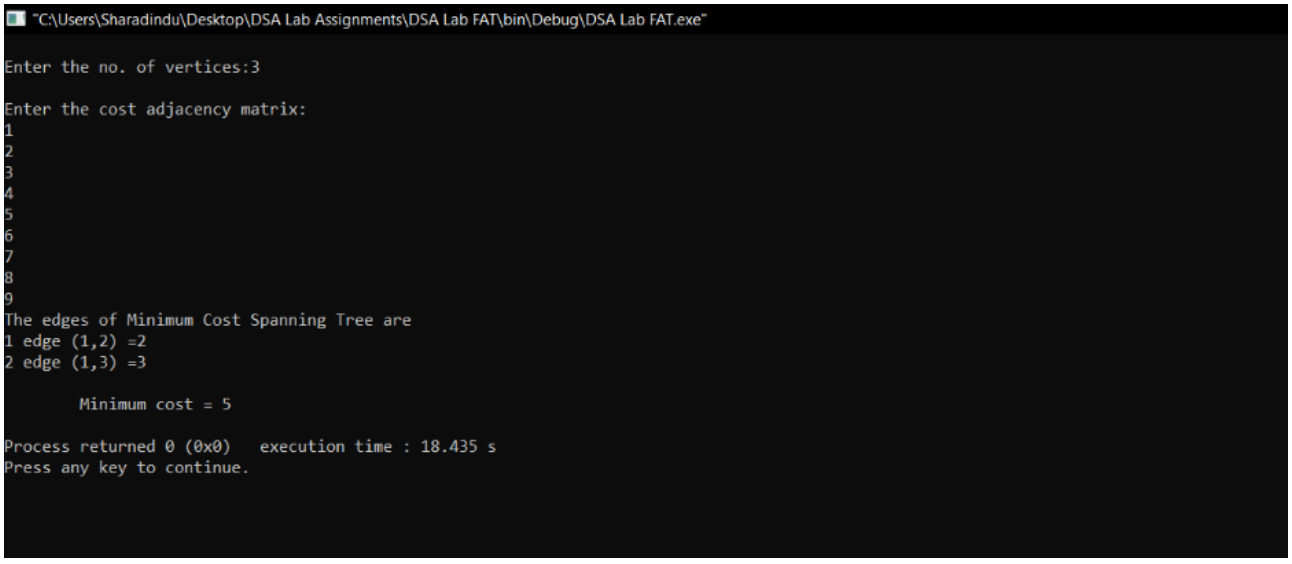



```
"C:\Users\Sharadindu\Desktop\DSA Lab Assignments\DSA Lab FAT\bin\Debug\DSA Lab FAT.exe"

Enter the number of Nodes:3

Enter the Adjacency Matrix:
1
2
3
4
5
6
7
8
9


 Edge 1:(1 2) cost:2
 Edge 2:(1 3) cost:3
 Minimun cost=5
Process returned 0 (0x0)   execution time : 26.427 s
Press any key to continue.
```

Kruskal's algorithm:

```
#include<stdio.h>
#include<stdlib.h>
int I,j,k,x,y,u,v,n,pl=1;
int min,min_c=0,cost[9][9],pa[9];
int find(int i)
{
 while(pa[i])
 i=pa[i];
 return I;
}
int uni(int I,int j)
{
 if(i!=j)
 {
 pa[j]=I;
 return 1;
 }
 return 0;
}
int main()
{
 printf("\nEnter the no. 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(pl < n)
 {
 for(i=1,min=999;i<=n;i++)
 {
 for(j=1;j <= n;j++)
 {
 if(cost[i][j] < min)
 {
 min=cost[i][j];
 x=u=I;
 y=v=j;
 }
 }
 }
 u=find(u);
 v=find(v);
 if(uni(u,v))
 {
 printf("%d edge (%d,%d) =%d\n",pl++,x,y,min);
 min_c +=min;
 }
 cost[x][y]=cost[y][x]=999;
 }
 printf("\n\tMinimum cost = %d\n",min_c);
 return 0;
}
```


main.c [DSA Lab FAT] - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Management
Projects Files
Workspace
DSA Lab FAT
Sources
main.c
#include<stdio.h>
#include<stdlib.h>
int i,j,k,x,y,u,v,n,pl=1;
int min,min_c=0,cost[9][9],pa[9];
int find(int i)
{
while(pa[i])
i=pa[i];
return i;
}
int uni(int i,int j)
{
if(i!=j)
{
pa[j]=i;
return 1;
}
return 0;
}
int main()
{
printf("\nEnter the no. 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)
Logs & others
Code::Blocks
Search results
Cccc
Build log
Build messages
CppCheck/Vera++
CppCheck/Vera++ messages
Cscope
Debugger
DoxyBlocks
Fortran info
Executing: "C:\Program Files\CodeBlocks/cb_console_runner.exe" "C:\Users\Sharadindu\Desktop\DSA Lab Assignments\DSA Lab FAT\bin\Debug\DSA Lab FAT.exe" (in C:\Users\Sharadindu\Desktop\DSA Lab Assignments\DSA Lab FAT\.)
C:\Users\Sharadindu\Desktop\DSA Lab Assignments\DSA Lab FAT\main.c
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 61, Col 1, Pos 905
Insert
Read/Write
default
ENG
08:27 AM


"C:\Users\Sharadindu\Desktop\DSA Lab Assignments\DSA Lab FAT\bin\Debug\DSA Lab FAT.exe"
Enter the no. of vertices:3
Enter the cost adjacency matrix:
1
2
3
4
5
6
7
8
9
The edges of Minimum Cost Spanning Tree are
1 edge (1,2) =2
2 edge (1,3) =3
Minimum cost = 5
Process returned 0 (0x0) execution time : 18.435 s
Press any key to continue.
