**PROGRAM**

#include<iostream.h>

#include<conio.h>

class prime

{

                int a,k,i;

              public:

              prime(int x)

              {

                            a=x;

              }

              void calculate()

              {

                 k=1;

                {

                     for(i=2;i<=a/2;i++)

       if(a%i==0)

                     {

                              k=0;

                              break;

                     }

                     else

                    {

                            k=1;

                  }

                }

              }

void show()

              {

                if(k==1)

                  cout<< “\n\tA is prime Number. ";

                else

                  cout<<"\n\tA is Not prime.";

              }

};

void main()

{

    clrscr();

    int a;

    cout<<"\n\tEnter the Number:";

    cin>>a;

    prime obj(a);

    obj.calculate();

    obj.show();

    getch();

}

**Output:**

****



**PROGRAM:**

#include<iostream.h>

#include<conio.h>

class copy

{

           int var,fact;

              public:

                copy(int temp)

                {

                 var = temp;

                }

                double calculate()

                {

                            fact=1;

                            for(int i=1;i<=var;i++)

                            {

                            fact = fact \* i;

                            }

                            return fact;

                }

};

void main()

{

    clrscr();

    int n;

    cout<<"\n\tEnter the Number : ";

    cin>>n;

    copy obj(n);

    copy cpy=obj;

    cout<<"\n\t"<<n<<" Factorial is:"<<obj.calculate();

    cout<<"\n\t"<<n<<" Factorial is:"<<cpy.calculate();

    getch();

}

**Output**





**PROGRAM**

#include <iostream.h>

using namespace std;

class myclass {

int a, b;

public:

friend int sum(myclass x);

void set\_ab(int i, int j);

};

void myclass::set\_ab(int i, int j)

{

a = i;

b = j;

}

// Note: sum() is not a member function of any class.

int sum(myclass x)

{

/\* Because sum() is a friend of myclass, it can directly access a and b. \*/

return x.a + x.b;

}

int main()

{

myclass n;

n.set\_ab(3, 4);

cout << sum(n);

return 0;

}

**Output:**



**PROGRAM:**

#include<iostream.h>

#include<conio.h>

 class emp

{

   public:

     int eno;

     char name[20],des[20];

     void get()

     {

              cout<<"Enter the employee number:";

              cin>>eno;

              cout<<"Enter the employee name:";

              cin>>name;

              cout<<"Enter the designation:";

              cin>>des;

     }

};

class salary:public emp

{

     float bp,hra,da,pf,np;

   public:

     void get1()

     {

              cout<<"Enter the basic pay:";

              cin>>bp;

              cout<<"Enter the Humen Resource Allowance:";

              cin>>hra;

              cout<<"Enter the Dearness Allowance :";

              cin>>da;

              cout<<"Enter the Profitablity Fund:";

              cin>>pf;

     }

void calculate()

     {

              np=bp+hra+da-pf;

     }

     void display()

     {

              cout<<eno<<"\t"<<name<<"\t"<<des<<"\t"<<bp<<"\t"<<hra<<"\t"<<da<<"\t"<<pf<<"\t"<<np<<"\n";

     }

};

 void main()

{

    int i,n;

    char ch;

    salary s[10];

    clrscr();

    cout<<"Enter the number of employee:";

    cin>>n;

    for(i=0;i<n;i++)

    {

              s[i].get();

              s[i].get1();

              s[i].calculate();

    }

    cout<<"\ne\_no \t e\_name\t des \t bp \t hra \t da \t pf \t np \n";

    for(i=0;i<n;i++)

    {

              s[i].display();

    }

    getch();

}

**Output:**



**PROGRAM**

#include<iostream.h>

#include<conio.h>

class student

{

    protected:

       int rno,m1,m2;

    public:

                void get()

              {

                            cout<<"Enter the Roll no :";

                            cin>>rno;

                            cout<<"Enter the two marks   :";

                            cin>>m1>>m2;

              }

};

class sports

{

    protected:

       int sm;                   // sm = Sports mark

    public:

                void getsm()

              {

                 cout<<"\nEnter the sports mark :";

                 cin>>sm;

              }

};

class statement:public student,public sports

{

    int tot,avg;

    public:

    void display()

              {

                 tot=(m1+m2+sm);

                 avg=tot/3;

                 cout<<"\n\n\tRoll No    : "<<rno<<"\n\tTotal      : "<<tot;

               cout<<"\n\tAverage    : "<<avg;

              }

};

void main()

{

   clrscr();

   statement obj;

   obj.get();

   obj.getsm();

   obj.display();

   getch();

}

**Output:**

              

**PROGRAM**

#include <iostream>

using namespace std;

class Polygon {

 protected:

 int width, height;

 public:

 void set\_values (int a, int b)

 { width=a; height=b; }

};

class Rectangle: public Polygon {

 public:

 int area()

 { return width\*height; }

};

class Triangle: public Polygon {

 public:

 int area()

 { return width\*height/2; }

};

int main () {

 Rectangle rect;

 Triangle trgl;

 Polygon \* ppoly1 = &rect;

 Polygon \* ppoly2 = &trgl;

 ppoly1->set\_values (4,5);

 ppoly2->set\_values (4,5);

 cout << rect.area() << '\n';

 cout << trgl.area() << '\n';

 return 0;

}

 **Output:**



**PROGRAM**

#include<iostream.h>

#include<stdlib.h>

#include<conio.h>

#define pi 3.14

class fn

{

      public:

        void area(int); //circle

        void area(int,int); //rectangle

        void area(float ,int,int);  //triangle

};

void fn::area(int a)

{

 cout<<"Area of Circle:"<<pi\*a\*a;

}

void fn::area(int a,int b)

{

      cout<<"Area of rectangle:"<<a\*b;

}

void fn::area(float t,int a,int b)

{

      cout<<"Area of triangle:"<<t\*a\*b;

}

void main()

{

     int ch;

     int a,b,r;

     clrscr();

     fn obj;

     cout<<"\n\t\tFunction Overloading";

     cout<<"\n1.Area of Circle\n2.Area of Rectangle\n3.Area of Triangle\n4.Exit\n:”;

     cout<<”Enter your Choice:";

     cin>>ch;

     switch(ch)

     {

              case 1:

                cout<<"Enter Radious of the Circle:";

                cin>>r;

                obj.area(r);

                break;

              case 2:

                cout<<"Enter Sides of the Rectangle:";

                cin>>a>>b;

                obj.area(a,b);

                break;

              case 3:

                cout<<"Enter Sides of the Triangle:";

                cin>>a>>b;

                obj.area(0.5,a,b);

                break;

              case 4:

                exit(0);

     }

getch();

}

**Output:**



****

****

**PROGRAM**

#include<iostream.h>

#include<conio.h>

class base

{

    public:

      virtual void show()

      {

                cout<<"\n  Base class show:";

      }

      void display()

      {

              cout<<"\n  Base class display:" ;

      }

};

class drive:public base

{

   public:

      void display()

      {

              cout<<"\n  Drive class display:";

      }

      void show()

      {

              cout<<"\n  Drive class show:";

      }

};

void main()

{

   clrscr();

   base obj1;

   base \*p;

   cout<<"\n\t P points to base:\n"  ;

   p=&obj1;

   p->display();

   p->show();

   cout<<"\n\n\t P points to drive:\n";

   drive obj2;

   p=&obj2;

   p->display();

   p->show();

   getch();

}

**Output:**



**PROGRAM**

#include<iostream.h>

#include<conio.h>

class complex

{

     int a,b,c;

    public:

        complex(){}

        void getvalue()

       {

                 cout<<"Enter the Two Numbers:";

                 cin>>a>>b;

       }

  void operator++()

      {

                 a=++a;

                 b=++b;

       }

       void operator--()

       {

                 a=--a;

                 b=--b;

        }

        void display()

        {

                 cout<<a<<"+\t"<<b<<"i"<<endl;

         }

};

 void main()

{

     clrscr();

     complex obj;

     obj.getvalue();

     obj++;

     cout<<"Increment Complex Number\n";

     obj.display();

     obj--;

     cout<<"Decrement Complex Number\n";

     obj.display();

     getch();

}

**Output:**



**PROGRAM**

#include<iostream.h>

#include<conio.h>

class complex

{

              int a,b;

    public:

              void getvalue()

              {

                 cout<<"Enter the value of Complex Numbers a,b:";

                 cin>>a>>b;

              }

              complex operator+(complex ob)

              {

                            complex t;

                            t.a=a+ob.a;

                            t.b=b+ob.b;

                            return(t);

              }

              complex operator-(complex ob)

              {

                            complex t;

                            t.a=a-ob.a;

                            t.b=b-ob.b;

                            return(t);

              }

              void display()

              {

                            cout<<a<<"+"<<b<<"i"<<"\n";

              }

};

void main()

{

   clrscr();

   complex obj1,obj2,result,result1;

   obj1.getvalue();

   obj2.getvalue();

   result = obj1+obj2;

   result1=obj1-obj2;

   cout<<"Input Values:\n";

   obj1.display();

   obj2.display();

   cout<<"Result:";

   result.display();

  result1.display();

   getch();

}

**Output:**



**PROGRAM**

#include<iostream.h>

#include<conio.h>

template<class t>

void swap(t &x,t &y)

{

   t temp=x;

   x=y;

   y=temp;

}

void fun(int a,int b,float c,float d)

{

   cout<<"\na and b before swaping :"<<a<<"\t"<<b;

   swap(a,b);

   cout<<"\na and b after swaping  :"<<a<<"\t"<<b;

   cout<<"\n\nc and d before swaping :"<<c<<"\t"<<d;

   swap(c,d);

   cout<<"\nc and d after swaping  :"<<c<<"\t"<<d;

}

void main()

{

    int a,b;

    float c,d;

    clrscr();

    cout<<"Enter A,B values(integer):";

    cin>>a>>b;

    cout<<"Enter C,D values(float):";

    cin>>c>>d;

    fun(a,b,c,d);

    getch();

}

**Output:**



**PROGRAM**

#include<iostream.h>

#include<conio.h>

void main()

{

   int a,b,c;

   float  d;

   clrscr();

   cout<<"Enter the value of a:";

   cin>>a;

   cout<<"Enter the value of b:";

   cin>>b;

   cout<<"Enter the value of c:";

   cin>>c;

   try

   {

              if((a-b)!=0)

              {

                 d=c/(a-b);

                 cout<<"Result is:"<<d;

              }

              else

              {

                 throw(a-b);

              }

   }

   catch(int i)

   {

              cout<<"Answer is infinite because a-b is:"<<i;

   }

   getch();

}

**Output:**

              



**PROGRAM**

#include<iostream.h>

#include<conio.h>

void test(int x)

{

   try

   {

              if(x>0)

                 throw x;

        else

                 throw 'x';

   }

   catch(int x)

   {

              cout<<"Catch a integer and that integer is:"<<x;

   }

   catch(char x)

   {

              cout<<"Catch a character and that character is:"<<x;

   }

}

void main()

{

   clrscr();

   cout<<"Testing multiple catches\n:";

   test(10);

   test(0);

   getch();

}

**Output:**



**PROGRAM**

#include<iostream.h>

 #include<vector>

using namespace std;

 int main()

{

 vector vec;

 int i;

cout << "vector size = " << vec.size() << endl;

 for(i = 0; i < 5; i++)

{

vec.push\_back(i);

}

cout << "extended vector size = " << vec.size() << endl;

for(i = 0; i < 5; i++)

{

cout << "value of vec [" << i << "] = " << vec[i] << endl;

 }

vector::iterator v = vec.begin();

 while( v != vec.end())

{

 cout << "value of v = " << \*v << endl;

 v++;

}

return 0;

}

**OUTPUT:**

 vector size = 0

extended vector size = 5

value of vec [0] = 0

value of vec [1] = 1

value of vec [2] = 2

value of vec [3] = 3

value of vec [4] = 4

value of v = 0

value of v = 1

value of v = 2

value of v = 3

value of v = 4

**PROGRAM**

#include <iostream.h>

#include <fstream.h>

#include<conio.h>

#include<stdlib.h>

void main()

{

ofstream out\_obj;

out\_obj.open(“emp.dat”);

out\_obj<<”Rahul\n”;

out\_obj<<”Lakhana\n”;

out\_obj<<”Nandan\n”;

out\_obj<<”Archana\n”;

out\_obj<<”Yogesh\n”;

out\_obj.close();

out\_obj.open(“dept.dat”);

out\_obj<<”Accounts\n”;

out\_obj<<”Proof\n”;

out\_obj<<”Marketing\n”;

out\_obj<<”DTP\n”;

out\_obj<<”Graphics design\n”;

out\_obj.close();

char data[80];

ifstream in\_obj;

in\_obj.open(“emp.dat”);

cout<<”\n following are contents of emp.dat file…\n”;

while(in\_obj)

{

in\_obj.getline(data,80);

cout<<”\n”<<data;

}

in\_obj.close();

in-obj.open(“dept.dat”);

cout<<”\n following are contents of dept.dat file…\n”;

while(in\_obj)

{

in\_obj.getline(data,80);

cout<<”\n”<<data;

}

in\_obj.close();

getch();

}

**Sample Output:**

following are contents of emp.dat file…

Rahul

Lakhana

Nandan

Archana

Yogesh

following are contents of dept.dat file…

Accounts

Proof

Marketing

DTP

Graphics design

**PROGRAM**

#include<iostream.h>

#include<conio.h>

#define operand(x) (x>='a'&&x<='z'||x>='A'&&x<='Z')

int top=-1;

class stack

{

char infix[20],stack[20],postfix[20];

public:

void getexpression()

{

 cout<<"enter the expression";

 cin>>infix;

}

void displayexpression()

{

 cout<<postfix;

}

stack()

{

}

void infixtopostfix();

int priority(char);

void push(char);

char pop();

};

void main()

{

stack a,b;

clrscr();

 a.getexpression();

 a.infixtopostfix();

 a.displayexpression();

 b.infixtopostfix();

 b.displayexpression();

 getch();

}

void stack::infixtopostfix()

{

 int j,l=0;

 char x,y;

 for(j=0;(x=infix[j])!='\0';j++)

 {

 if(operand(x))

 postfix[l++]=x;

 else if(x=='(')

 push(x);

 else if(x==')')

 {

 while((y=pop())!='(')

 postfix[l++]=y;

 }

 else

 {

 while(priority(stack[top])>=priority(x)&&stack[top]!='(')

 postfix[l++]=pop();

 push(x);

 }

 }

 while(top>=0)

 postfix[l++]=pop();

}

void stack::push(char n)

{

 stack[++top]=n;

}

 int stack::priority(char x)

{

 int y;

 y=(x=='('?3:x=='\*'?2:x=='/'?2:x=='+'?1:x=='-'?1:-1);

 return y;

}

char stack::pop()

{

 char n;

 n=stack[top];

 top--;

 return(n);

}

**OUTPUT:**



**PROGRAM**

# include <iostream.h>

# include <conio.h>

# include <stdlib.h>

class cq

{

 private:

 int rear,front,x,d,\*a,csize;

 public:

 int fullq();

 int emptyq();

 void enq();

 void deq();

 void display();

 cq()

 {

 rear=0;

 front=0;

 csize=0;

 cout<<"Enter the size"<<endl;

 cin>>x;

 a=new int[x];

 }

};

int cq::fullq()

{

 if(csize==x)

 return 1;

 else

 return 0;

}

int cq::emptyq()

{

 if(csize==0)

 return 1;

 else

 return 0;

}

void cq::enq()

{

 if(!fullq())

 {

 cout<<"Enter the value to be inserted"<<endl;

 cin>>d;

 a[rear]=d;

 rear=(rear+1)%x;

 csize++;

 }

 else

 cout<<"The queue is full"<<endl;

}

void cq::deq()

{

 if(!emptyq())

 {

 d=a[front];

 cout<<"The value retrieved is"<<d;

 front=(front+1)%x;

 csize--;

 }

 else

 cout<<"The queue is empty"<<endl;

}

void cq::display()

{

 int i,j;

 i=front;

 j=csize;

 if(j==0)

 {

 cout<<"queue is empty"<<endl;

 return;

 }

 cout<<"The value in queue is"<<endl;

 while(j)

 {

 cout<<a[i]<<" ";

 j--; i++;

 }

}

void main()

{

 clrscr();

 cq c1;

 int ch;

 while(1)

 {

 cout<<"\n1.enqueue operation\n2.dequeue operation\n3.display\n 4.exit\n"<<endl;

 cout<<"\nEnter your choice\n"<<endl;

 cin>>ch;

 switch(ch)

 {

 case 1:

 c1.enq();

 break;

 case 2:

 c1.deq();

 break;

 case 3:

 c1.display();

 break;

 case 4:

 exit(0);

 break;

 default:

 cout<<"Wrong choice"<<endl;

 }

 }

}

**Output:**

Enter the size 4

1.enqueue operation

2.dequeue operation

3.display

4.exit

Enter your choice 1

Enter the value to be inserted

2 3 4

1.enqueue operation

2.dequeue operation

3.display

4.exit

Enter your choice 3

The value in queue is 2 3 4

1.enqueue operation

2.dequeue operation

3.display

4.exit

Enter your choice 3

The value in queue is 2 3 4

1.enqueue operation

2.dequeue operation

3.display

4.exit

Enter your choice 2

The value retrieved is 2 3 4

1.enqueue operation

2.dequeue operation

3.display

4.exit

Enter your choice 4

**PROGRAM**

#include<stdio.h>

#include<conio.h>

typedef struct node \*tree;

tree findmin(tree);

tree findmax(tree);

tree insert(int,tree);

tree del(int,tree);

void disp(tree);

struct node

{

 int data;

 tree left,right;

}\*t=NULL;

void main()

{

 int ch,n;

 tree a;

 clrscr();

 printf("1.insert 2.delete 3.findmax 4.findmin 5.disp 6.exit");

 do

 {

 printf("enter choie");

 scanf("%d",&ch);

 switch(ch)

 {

 case 1:

 printf("enter no to insert");

 scanf("%d",&n);

 t=insert(n,t);

 break;

 case 2:

 printf("enter no to delete");

 scanf("%d",&n);

 t=del(n,t);

 break;

 case 3:

 a=findmax(t);

 printf("%d",a->data);

 break;

 case 4:

 a=findmin(t);

 printf("%d",a->data);

 break;

 case 5:

 disp(t);

 break;

 case 6:

 exit (0);

 break;

 default:

 printf("enter correct choice");

 }

 }while(ch!=6);

 getch();

}

tree insert(int n,tree t)

{

 if(t==NULL)

 {

 t=(tree)malloc(sizeof(struct node));

 t->data=n;

 t->left=t->right=NULL;

 }

 else if(n<t->data)

 t->left=insert(n,t->left);

 else if(n>t->data)

 t->right=insert(n,t->right);

 else

 printf("already exist");

 return t;

}

tree del(int n,tree t)

{

 tree p;

 if(t==NULL)

 printf("element not found");

 else if(n<t->data)

 t->left=del(n,t->left);

 else if(n>t->data)

 t->right=del(n,t->right);

 else if(t->left&&t->right)

 {

 p=findmin(t->right);

 t->data=p->data;

 t->right=del(t->data,t->right);

 }

 else

 {

 p=t;

 if(t->left==NULL)

 t=t->right;

 else

 t=t->left;

 free(p);

 }

 return t;

}

void disp(tree t)

{

 if(t!=NULL)

 {

 disp(t->left);

 printf("%d\n",t->data);

 disp(t->right);

 }

}

tree findmin(tree t)

{

 if(t==NULL)

 return NULL;

 else if(t->left==NULL)

 return t;

 else return findmin(t->left);

}

tree findmax(tree t)

{

 if(t==NULL)

 return NULL;

 else if(t->right==NULL)

 return t;

 else

 return findmax(t->right);

}

**OUTPUT:**

****

**PROGRAM**

#include<stdio.h>

#include<conio.h>

#define operand(x) (x>='a'&&x<='z'||x>='A'&&x<='Z')

typedef struct node \*tree;

void push(tree);

tree pop();

void conversion();

void postfix(tree);

void infix(tree);

void prefix(tree);

struct node

{

 char data;

 tree left,right;

}\*T=NULL;

char post[20];

tree stack[20];

int top=-1;

void main()

{

 clrscr();

 printf("enter postfix expression");

 scanf("%s",post);

 conversion();

 printf("\npostfix expression is ");

 postfix(T);

 printf("\ninfix expression is ");

 infix(T);

 printf("\nprefix expression is ");

 prefix(T);

 getch();

}

void conversion()

{

 int i;

 tree a,b,c;

 char x;

 for(i=0;(x=post[i])!='\0';i++)

 {

 if(operand(x))

 {

 a=(tree)malloc(sizeof(struct node));

 a->data=x;

 a->left=a->right=NULL;

 push(a);

 }

else

 {

 a=pop();

 b=pop();

 c=(tree)malloc(sizeof(struct node));

 c->data=x;

 c->right=a;

 c->left=b;

 push(c);

 }

 }

 T=stack[top];

}

void push(tree a)

{

 stack[++top]=a;

}

tree pop()

{

 tree a;

 a=stack[top];

 top--;

 return(a);

}

void postfix(tree a)

{

 if(a!=NULL)

 {

 postfix(a->left);

 postfix(a->right);

 printf("%c",a->data);

 }

}

void infix(tree a)

{

 if(a!=NULL)

 {

 infix(a->left);

 printf("%c",a->data);

 infix(a->right);

 }

}

void prefix(tree a)

{

 if(a!=NULL)

 {

 printf("%c",a->data);

 prefix(a->left);

 prefix(a->right);

 }

}

**OUTPUT:**

****

**PROGRAM**

# include <iostream.h>

# include <conio.h>

class prim

{

 private:

 struct vertex

 {

 int visit;

 int cost;

 int par;

 };

 vertex v[10];

 int a[10][10],n,ver;

 public:

 prim();

 void read();

 void chose();

 void display();

};

prim::prim()

{

 cout<<"Enter the number of nodes"<<endl;

 cin>>n;

 for(int i=1;i<=n;i++)

 {

 v[i].visit=0;

 v[i].cost=0;

 v[i].par=0;

 for(int j=1;j<=n;j++)

 a[i][j]=0;

 }

 ver=0;

}

void prim::read()

{

 for(int i=1;i<=n;i++)

 {

 for(int j=i+1;j<=n;j++)

 {

 if(i!=j)

 {

 cout<<"Enter the values of"<<i<<","<<j<<endl;

 cin>>a[i][j];

 a[j][i]=a[i][j];

 } } } }

void prim::chose()

{

 int min=0;

 int k=0,loc,loc1;

 v[1].visit=1;

 v[1].cost=0;

 v[1].par=0;

 while(k!=n-1)

 {

 int flag=0;

 for(int i=1;i<=n;i++)

 {

 if(v[i].visit)

 {

 for(int j=1;j<=n;j++)

 {

 if(a[i][j] && !flag && !v[j].visit)

 {

 min=a[i][j];

 loc=i;

 loc1=j;

 ver++;

 flag++;

 }

 if(a[i][j] && min>=a[i][j] && !v[j].visit && flag)

 {

 min=a[i][j];

 loc=i;

 loc1=j;

 } } } }

 v[loc1].visit=1;

 v[loc1].cost=min;

 v[loc1].par=loc;

 a[loc][loc1]=0;

 a[loc1][loc]=0;

 k++;

 } }

void prim::display()

{

 int d=0;

 if(ver==n-1)

 {

 cout<<"Cost"<<" "<<"visit"<<" "<<"parent"<<" "<<"current node"<<endl;

 for(int i=1;i<=n;i++)

 {

 d=d+v[i].cost;

 cout<<v[i].cost<<" ";

 cout<<v[i].visit<<" ";

 cout<<v[i].par<<" "<<i<<endl;

 }

 cout<<"Minimum cost is"<<d<<endl;

 }

 else

 cout<<"No spanning tree exists"<<endl;

}

void main()

{

 clrscr();

 prim p1;

 p1.read();

 p1.chose();

 p1.display();

 getch();

}

**Output:**

Enter the number of nodes

3

Enter the values of1,2

67

Enter the values of1,3

87

Enter the values of2,3

97

Cost visit parent current node

0 1 0 1

67 1 1 2

87 1 1 3

Minimum cost is154

**PROGRAM**

# include <iostream.h>

# include <conio.h>

class krusk

{

 private:

 struct edge

 {

 int cost;

 int sou;

 int des;

 int select;

 };

 int ecount,a[10][10],\*par,n,flag;

 edge \*e;

 public:

 krusk();

 void read();

 void chose\_edge();

 void union1(int,int);

 int find(int);

 void display();

};

krusk::krusk()

{

 cout<<"Enter the number of nodes"<<endl;

 cin>>n;

 par=new int[n+1];

 for(int i=1;i<=n;i++)

 {

 par[i]=-1;

 for(int j=1;j<=n;j++)

 {

 a[i][j]=0; a[j][i]=0;

 }

 }

 ecount=0; flag=1;

}

void krusk::read()

{

 for(int i=1;i<=n;i++)

 {

 for(int j=i+1;j<=n;j++)

 {

 cout<<"Enter the values of"<<i<<","<<j<<endl; cin>>a[i][j];

 a[j][i]=a[i][j];

 if(a[i][j])

 ecount++;

 } }

 e=new edge[ecount];

}

void krusk::union1(int i,int j)

{

 int t=par[i]+par[j];

 if(par[i]>par[j])

 {

 par[i]=j;

 par[j]=t;

 }

 else

 {

 par[j]=i;

 par[i]=t;

 } }

int krusk::find(int i)

{

 int j=i;

 while(par[j]>0)

 j=par[j];

 int k=i; int temp;

 while(k!=j)

 {

 temp=par[k];

 par[k]=j;

 k=temp;

 }

 return j;

}

void krusk::chose\_edge()

{

 int k=1;

 for(int i=1;i<=n;i++)

 {

 for(int j=i+1;j<=n;j++)

 {

 if(a[i][j])

 {

 e[k].cost=a[i][j];

 e[k].sou=i;

 e[k].des=j;

 e[k].select=0;

 k++;

 }

 }

 }

 for(i=1;i<=ecount;i++)

 {

 for(int j=1;j<ecount;j++)

 {

 if(e[j].cost>e[j+1].cost)

 {

 int temp=e[j].cost;

 e[j].cost=e[j+1].cost;

 e[j+1].cost=temp;

 temp=e[j].sou;

 e[j].sou=e[j+1].sou;

 e[j+1].sou=temp;

 temp=e[j].des;

 e[j].des=e[j+1].des;

 e[j+1].des=temp;

 } } }

 int t=0;

 k=1;

 int ecount1=ecount;

 while(t<n-1 && ecount1!=0)

 {

 int u=e[k].sou;

 int v=e[k].des;

 ecount1--;

 if(find(u)!=find(v))

 {

 e[k].select=1;

 t++;

 union1(find(u),find(v));

 }

 k++;

 }

 if(t<n-1)

 {

 cout<<"No spanning tree exists"<<endl;

 flag--;

 } }

void krusk::display()

{

 int mincost=0;

 if(flag)

 {

 cout<<"cost "<<"source "<<"dest "<<endl;

 for(int i=1;i<=ecount;i++)

 {

 if(e[i].select!=0)

 {

 mincost=mincost+e[i].cost;

 cout<<e[i].cost<<" "<<e[i].sou<<" "<<e[i].des<<" ";

 } }

 cout<<"Cost of minimum spanning tree is"<<mincost<<endl;

 } }

void main()

{

 clrscr();

 krusk k1;

 k1.read();

 k1.chose\_edge();

 k1.display();

 getch();

}

**Output:**

Enter the number of nodes

3

Enter the values of1,2

34

Enter the values of1,3

56

Enter the values of2,3

78

cost source dest

34 1 2

56 1 3

Cost of minimum spanning tree is 90

**PROGRAM**

# include <iostream.h>

# include <conio.h>

class dij

{

 private:

 struct path

 {

 int dis;

 int par;

 int visit;

 };

 path \*p;

 int n,a[10][10],sou,des;

 public:

 dij();

 void read();

 void chose();

 void display();

 void print(int);

};

dij::dij()

{

 cout<<"Enter the number of nodes"<<" "; cin>>n;

 cout<<"Enter the source node"<<" "; cin>>sou;

 cout<<"Enter the destination node"<<" "; cin>>des;

 p=new path[n];

 for(int i=0;i<=n;i++)

 {

 p[i].dis=0;

 p[i].par=0;

 p[i].visit=0;

 for(int j=0;j<=n;j++)

 a[i][j]=0;

 }

}

void dij::read()

{

 for(int i=1;i<=n;i++)

 {

 for(int j=1;j<=n;j++)

 {

 if(i!=j)

 {

 cout<<"Enter the values of"<<i<<","<<j<<" ";

 cin>>a[i][j];

 }

 }

 }

}

void dij::chose()

{

 int flag,v=0;

 p[sou].dis=0;

 p[sou].par=0;

 p[sou].visit=1;

 while(v<=n-1)

 {

 for(int i=1;i<=n;i++)

 {

 if(p[i].visit)

 {

 for(int j=1;j<=n;j++)

 {

 if(a[i][j] && !p[j].visit && (p[j].dis > p[i].dis+a[i][j] || !p[j].dis))

 {

 p[j].dis=p[i].dis+a[i][j];

 p[j].par=i;

 }

 }

 }

 }

 int min=0,flag=0,loc;

 for(i=1;i<=n;i++)

 {

 if(p[i].dis && !flag && !p[i].visit)

 {

 min=p[i].dis;

 loc=i;

 flag++;

 }

 if(p[i].dis && min>p[i].dis && !p[i].visit)

 {

 min=p[i].dis;

 loc=i;

 }

 }

 p[loc].visit=1;

 v++;

 }

}

void dij::display()

{

 int i=des,flag=0;

 while(i<=n)

 {

 if(p[i].visit)

 {

 cout<<"shortest path from "<<sou<<"to "<<des<<"with distance "<<p[i].dis<<endl;

 print(i);

 cout<<i<<endl;

 break;

 }

 else

 flag++;

 i++;

 }

 if(flag)

 cout<<"The node "<<des<<" is not reachable"<<endl;

}

void dij::print(int i)

{

 if(i!=sou)

 {

 i=p[i].par;

 print(i);

 cout<<i<<"--->";

 }

 else

 return;

}

void main()

{

 clrscr();

 dij d1;

 d1.read();

 d1.chose();

 d1.display();

 getch();

}

**Output:**

Enter Number of Node: 5

Enter the source node 2

Enter the destination node 3

Enter the values of1,2 12

Enter the values of1,3 23

Enter the values of1,4 22

Enter the values of1,5 44

Enter the values of2,1 333

Enter the values of2,3 55

Enter the values of2,4 22

Enter the values of2,5 5

Enter the values of3,1 3

Enter the values of3,2 7

Enter the values of3,4 234

Enter the values of3,5 645

Enter the values of4,1 3423

Enter the values of4,2 678

Enter the values of4,3 234

Enter the values of4,5 768

Enter the values of5,1 5

Enter the values of5,2 76

Enter the values of5,3 646

Enter the values of5,4 4

Shortest path from 2to 3with distance 33

2--->5--->1--->3