Write a program to create a class shape with functions to find area of the shapes and display the name of the shape and other essential component of the class. Create derived classes circle, rectangle and trapezoid each having overridden functions area and display. Write a suitable program to illustrate virtual functions and virtual destructor.

```
#include <iostream>
#include <cstring>
#define pi 3.1415
using namespace std;
class shape
{
 protected:
  string sname;
  float sarea;
 public:
  shape()
  {
    sname = "shape";
    sarea = 0;
  }
  shape(float a, string n="shape")
  {
    sname = n;
    sarea = a;
  }
  virtual float area()
  {
    return sarea;
  }
  string name()
  {
    cout << "Shape " << sname << endl;
    return sname;
  }
  virtual ~shape()
  {
    cout << "Destructor of Shape " << endl;
  }
};
```

```
class circle:public shape
{
 protected:
  float radius;
 public:
  circle(int r, string n = "circle")
  {
    radius = r;
    sname = n;
  }
  float area()
  {
    sarea = pi * radius * radius;
    return shape::area();
  }
  string name()
  {
    cout << "Circle " << sname << endl;
    return sname;
  }
   ~circle()
  {
    cout << "Circle destructor" << endl;
  }

};
class rectangle:public shape
{
 protected:
  float length, breadth;
 public:
  rectangle(float l, float b, string n="rectangle"):length(l),breadth(b)
  {
    sname = n;
  }
  float area()
  {
    sarea = length*breadth;
    return shape::area();
  }
```

```
 string name()
 {
  cout << "Rectangle " << sname << endl;
  return sname;
 }
  ~rectangle()
 {
  cout << "Rectangle destructor" << endl;
 }
};
class trapezoid:public shape
{
 protected:
 float paralleside[2];
 float nonparallelside[2];
 public:
 trapezoid(float a1, float a2, float b1, float b2, string n= "Trapezoid")
 {
  paralleside[0] = a1;
  paralleside[1] = a2;
  nonparallelside[0] = b1;
  nonparallelside[1] = b2;
  sname = n;
 }
 float area()
 {
  sarea =
(paralleside[0]+paralleside[1])/2.0*(nonparallelside[0]+nonparallelside[1])/2.0;
  return shape::area();
 }
 string name()
 {
  cout << "Trapezoid " << sname << endl;
  return sname;
 }
  ~trapezoid()
 {
  cout << "Trapezoid destructor" << endl;
 }
};
```

```
int main()
{
  shape *sh;
  sh = new circle(4,"ball");
  sh->name();
  cout << sh->area() << endl;
  delete(sh);
  sh = new trapezoid(200,400, 100 , 100,"fancy stadium");;
  sh->name();
  cout << sh->area() << endl;
  delete(sh);
  sh = new rectangle(240,240,"ground");;
  sh->name();
  cout << sh->area() << endl;
  delete(sh);
  return 0;
}

#include<iostream>//or
#define PI 3.141592
using namespace std;
class shape
{
public:
   virtual void getdata()=0;
   virtual void area()=0;
   virtual void display()=0;
   virtual ~shape()
   {
      cout<<"Base class destructor called"<<endl;
   }
};
class circle:public shape
{
   float r,a;
public:
   void getdata()
   {
      cout<<endl<<"Enter the radius:";
      cin>>r;
```

```
    }
    void area()
    {
        a=PI*r*r;
    }
    void display()
    {
        cout<<"\nArea="<<a<<endl;
        cout<<"Circle"<<endl;
    }
    ~circle()
    {
        cout<<"Circle class destructor called"<<endl;
    }
};
class rectangle:public shape
{
    float l,b,a;
public:
    void getdata()
    {
        cout<<endl<<"Enter length:";
        cin>>l;
        cout<<"\nEnter breadth:";
        cin>>b;
    }
    void area()
    {
        a=l*b;
    }
    void display()
    {
        cout<<"\nArea="<<a<<endl;
        cout<<"Rectangle"<<endl;
    }
    ~rectangle()
    {
        cout<<"Rectangle class destructor called"<<endl;
    }
};
```

```
class trapezoid:public shape
{
   float l1,l2,h,a;
public:
   void getdata()
   {
      cout<<endl<<"Enter the length of first parallel side:";
      cin>>l1;
      cout<<"\nEnter the length of the second parallel side:";
      cin>>l2;
      cout<<"\nEnter the height:";
      cin>>h;
   }
   void area()
   {
      a=h*(l1+l2)/2;
   }
   void display()
   {
      cout<<"Area="<<a<<endl;
      cout<<"Trapezoid"<<endl;
   }
   ~trapezoid()
   {
      cout<<"Trapezoid class destructor called"<<endl;
   }
};
int main()
{
   shape *bobj;
   bobj=new circle;
   bobj->getdata();
   bobj->area();
   bobj->display();
   delete bobj;
   bobj=new rectangle;
   bobj->getdata();
   bobj->area();
   bobj->display();
   delete bobj;
```

```
    bobj=new trapezoid;
    bobj->getdata();
    bobj->area();
    bobj->display();
    delete bobj;
    return 0;
}
```