Class Shape is fully implemented.

```
class Shape
{
        STYPE type;
public:
        Shape();
        Shape(STYPE type);
        virtual void print();
        virtual double area();
        virtual bool contains(const Shape *s);
        //Intentionally made non-virtual
        bool operator==(const Shape &rhs);
        virtual ~Shape(){}
};
```

Class Point2D is fully implemented.

```
class Point2D : public Shape
{
        double x,y;
public:
        Point2D();
        Point2D(STYPE argtype, int argx, int argy);
        void print();
        double area();
        bool contains(const Shape * rhs);
        bool contains(const Point2D * rhs);
        bool operator==(const Point2D &rhs);

        double getX() const { return x; }
        double getY() const { return y; }
        ~Point2D(){}
};
```

Class Rectangle has some functions which must be implemented.

```
class Rectangle : public Shape
{
        Point2D topRight;
        Point2D bottomLeft;
Public:
        Rectangle();//implement this
        Rectangle(STYPE type, Point2D tr, Point2D bl);//implement this
        void print(); //implement this
        double area(); //implement this
        bool contains(const Shape *rhs); //implement this to check containment for point2d only
```

```
        bool operator==(const Rectangle &rhs); //implement this
        ~Rectangle(){}
}
```

Class Circle has some functions which must be implemented.

```
class Circle : public Shape
{
        Point2D center;
        double radius;
public:
        Circle();
        Circle(STYPE type, Point2D c, double r); //implement this
        double area(); //implement this
        void print(); //implement this
        bool contains(const Shape * s); //implemented partially
        bool contains(const Point2D * p); //implement this
        bool contains(const Circle * rhs); //implement this
        bool contains(const Rectangle * p); //implement this
        bool operator==(const Circle &rhs); //implement this
        ~Circle(){}
};
```

Class Point3D has some functions which must be implemented.

```
class Point3D : public Point2D
{
   double z;
public:
   Point3D(); //already implemented
   Point3D(STYPE type, double argx, double argy, double argz); //implement this
   void print(); //already implemented
   //double area(); Not needed, base version is sufficient
        bool contains(const Shape *rhs); //implement this
        bool contains(const Point3D *rhs);  //implement this
        ~Point3D(){}
};
```

Class Shape3D is fully implemented.

```
class Shape3D : public Shape
{
public:
        Shape3D();
        Shape3D(STYPE type);
        virtual double volume();
```

```
        ~Shape3D(){}
};
```

Class Sphere has some functions which must be implemented.

```
class Sphere : public Shape3D
{
        Point3D center;
        double radius;
public:
        Sphere();//implement this
        Sphere(STYPE type, Point3D c, double r);//implement this
        void print(); //implement this
        double area(); //implement this
        double volume(); //implement this
        bool contains(const Shape * rhs); //implement this for checking containgment of Point3D,
Sphere inside sphere
        bool contains(const Point3D * p); //implement this
        bool contains(const Sphere * s); //implement this
        ~Sphere(){}
};
```