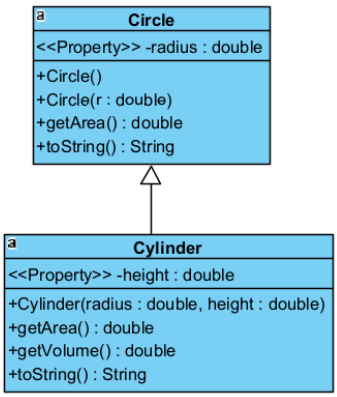
CMPS251 – Object Oriented Programming – Quiz-3

Student Name	
ID Number	
Total (100)	

A. Extend the Circle class. Create the Cylinder subclass. Override, reuse, and add attributes and methods whenever needed. Use proper annotation. See the output of the Cylinder class below. (40 Points)

```
public class Circle {
        private double radius;
        public Circle() {
        }
        public Circle (double r) {
                radius = r; }
        public void setRadius(double r) {
                if (r >= 0)
                        radius = r; }
        public double getRadius() {
                return radius;   }
        public double getArea() {
                return Math.PI * radius * radius;        }
        public String toString() {
                String details =
                String.format(" Radius: %.2f Area: %.2f.\n",
        radius, getArea());
                return details; }}
```



`Circle  c1 = new Circle(4.5);` `System.out.println(c1); // Output - - - - - - - -- >`	`Radius: 4.50 Area: 63.62.`
`Cylinder cc = new Cylinder(3.5, 7);` `System.out.println(cc); // Output - - - - - - - -- >`	`Height: 7.0 Radius: 3.50 Area: 230.91.` `Volume: 269.391570045324`

$$Volume = \pi r^2 h \qquad Surface\ Area = 2\pi r^2 + 2\pi r h$$

```
//Write complete Cylinder class starting from here.
```

DO NOT DUPLICATE CODE UNNECESSARILY. Reuse existing code whenever possible.

B. Complete the Code – 20 points.

```
//Complete the following code using the circles ArrayList.
// Find the total volume of all Cylinder objects in the list.
ArrayList<Circle> circles = new ArrayList<Circle>();
circles.add(new Circle(4.5));
circles.add(new Cylinder(3, 8.7));
circles.add(new Circle(14.5));
circles.add(new Cylinder(5, 5.5));
```

C. COMPOSITION: Write a complete Class - 40 points.

```
public class Address {
    private String street;
    private String city;
    private String state;public Address(String
street, String city, String state) {
        this.street = street;
        this.city = city;
        this.state = state; }
public String getStreet() {
        return street; }
public String getCity() {
        return city; }
public String getState() {
        return state;}
@Override
public String toString() {
  return street + ", " + city + ", " + state ; }}
```

```
public class Room {
    private String name;
    private double area;
public Room(String name, double area) {
        this.name = name;
        this.area = area;
public String getName() {
        return name; }
public double getArea() {
        return area; }
@Override
public String toString() {
 return "Room:" + name + ", Area:" + area ;} }
```

Consider the **House** class represented in the following UML Diagram:
It has **one Address** and **multiple rooms**. **Implement** all the code of the **House** class.
See sample RUN. Pay attention to the toString output.

```
+--------------+
|     House    |
+--------------+
| - rooms: ArrayList<Room> |
| - address: Address |
+--------------+
| + House(Address) |
| + void addRoom(Room) |
| + double getTotalArea() |
| + int getRoomCount() |
| + String toString() |
+--------------
```

```
Address address = new Address("123 Main St", "Anytown", "CA");
House house = new House(address);
Room room1 = new Room("living room", 200);
Room room2 = new Room("bedroom", 150);
house.addRoom(room1);
house.addRoom(room2);
System.out.println(house);
```

Sample Output

```
House: 2 rooms. Total Area=350.0
        Room: living room, Area:200.0
        Room: bedroom, Area:150.0
Address: 123 Main St, Anytown, CA
```

DO NOT DUPLICATE CODE UNNECESSARILY. Reuse existing code whenever possible.