Abstract Class Question-Practical session 6/07/2023

Name- G.O Wickramaratne

ID- 28039

Question 01

```
// if a class is a abstract atleast there should be atleast one abstract method
public abstract class Container {
    public abstract double volume();


}


public class CylindricalContainer extends Container {
   private double height;
   private double radius;
   public CylindricalContainer(double height,double radius){
       this.height=height;
       this.radius=radius;
   }
   public double volume(){
       return 3.1459f *radius *radius*height;
   }

}

public class Abstractq1 {

    public static void main(String[] args) {
        CylindricalContainer a1= new CylindricalContainer(8.76,12.45);
        System.out.println("Volume of the cylinder is "+a1.volume());

    }
}
```

Question 02

```
// Base class for all players
class Player {
    private int x;
    private int y;

    public Player(int x, int y) {
        this.x = x;
        this.y = y;
    }
    public void moveUp() {
        y--;
        System.out.println("Player moved up");
    }

    public void moveDown() {
        y++;
        System.out.println("Player moved down");
    }

    public void moveLeft() {
        x--;
        System.out.println("Player moved left");
    }

    public void moveRight() {
        x++;
        System.out.println("Player moved right");
    }
}

// Player that moves in the opposite direction
class OppositePlayer extends Player {
    public OppositePlayer(int x, int y) {
        super(x, y);
    }

    @Override
    public void moveUp() {
        super.moveDown();
    }

    @Override
    public void moveDown() {
        super.moveUp();
    }

    @Override
    public void moveLeft() {
        super.moveRight();
    }

    @Override
    public void moveRight() {
```

	```
	        super.moveLeft();
	    }
	}

	public class Main {
	    public static void main(String[] args) {
	        Player regularPlayer = new Player(0, 0);
	        OppositePlayer oppositePlayer = new OppositePlayer(0, 0);

	        regularPlayer.moveUp();
	        regularPlayer.moveLeft();
	        oppositePlayer.moveDown();
	        oppositePlayer.moveRight();
	    }
	}
	```