Classwork - Sep 24

#sagar-sir

```
class Car{
    private String color;
    private int speed;

    public void setColor(String) { color = c; }
    public void setSpeed(int s) { if(s >= 0) speed = s;}

    public String getColor() { return color; }
    public int getSpeed() { return  speed; }

    public void drive() {
        System.out.println(color + " car driving at " + speed + " km/h");
    }

    public class Main {

    public static void main(String[] args) {

        Car myCar = new Car();

        myCar.setColor("Green");
        myCar.setSpeed(70);
        myCar.drive();
        System.out.println("I know the color is: " + myCar.getColor());
    }
}
```

Basic Comprehension Questions

1. What is the purpose of getter and setter methods?

-> Getter and setter methods are used to encapsulate properties of a class. They are to prevent direct access to properties and instead provide access to them through methods; setters and getters.

Setter methods are used to change the value of a property. Example: `setSpeed()`.
Getter methods are used to retrieve the value of a property. Example: `getColor()`.

2. Why are the color and speed variables declared as private?

-> We are encapsulating these variables and using getter and setter methods for their access control.

3. What does the drive() method do when called?

-> It prints the color of the car and the speed it is driving at to the standard output. In the above example, we use the `setColor` method to set the color property to `Green` and `setSpeed` method to set the speed property to `70`. When we call the `drive()` method it prints this to the terminal:

```
Green car driving at 70 km/h
```

4. What output will this program produce when executed?

-> It produces this output:

```
Green car driving at 70 km/h
```

Code Analysis Questions

1. Why does the setSpeed() method include a condition if(s >= 0)?

-> This is to make sure that a negative value of speed is not assigned to the `speed` variable, since speed cannot be negative.

2. What would happen if we tried to access myCar.color directly in the Main class?

-> It would throw an error because the color property has `private` access modifier, meaning it cannot be accessed by other classes.

3. How many Car objects are created in this program?

-> Just one `Car` object.

4. What is the initial state of a Car object when it's first created?

-> When a `Car` object is created, it's instance variables `color` and `speed` are initialized to their default values in Java because no constructor has explicitly set them.

So,
color (a `String`) = `null`
speed (a `int`) = 0