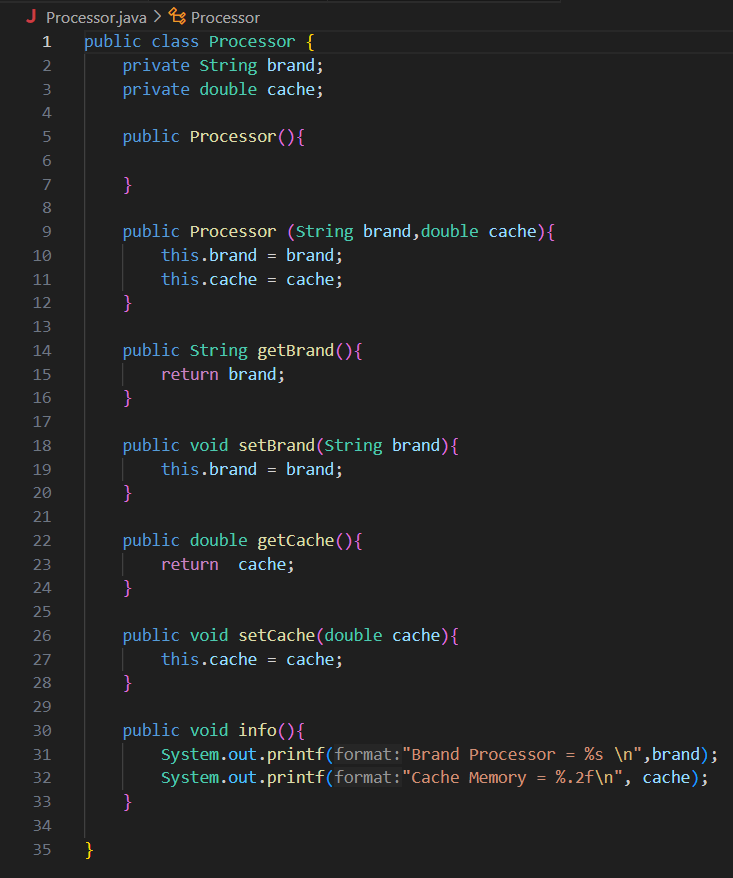
**Jobsheet 04 ‑ Relationships Between Classes**

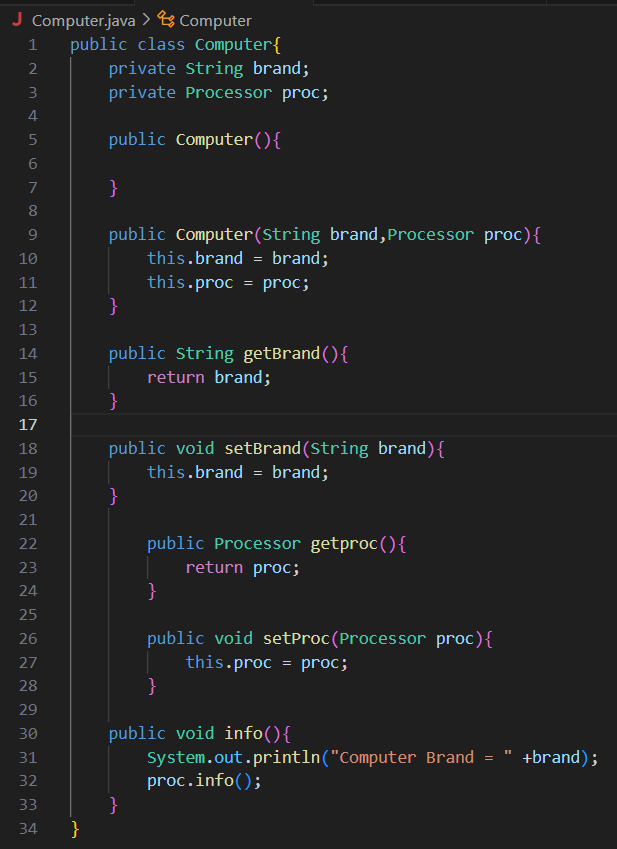
Erwan Majid/08/2i

Link github: <https://github.com/Majid5654/Semester-3/tree/Main/JAVA%20OOP/Week%205>

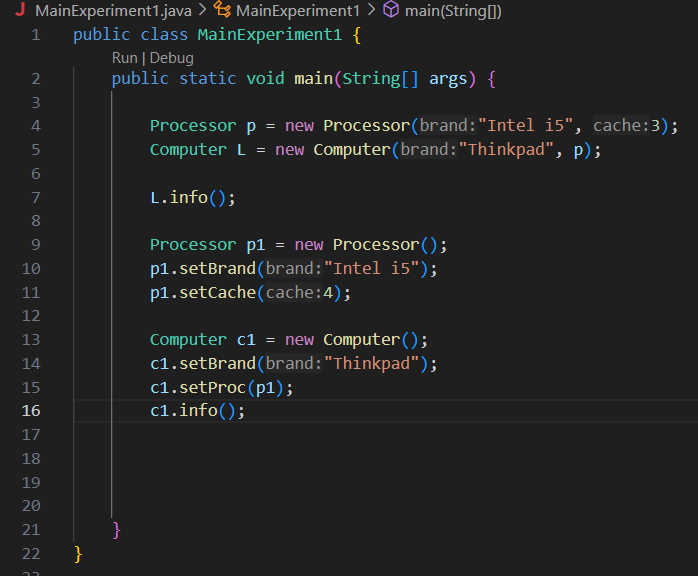
* **Experiment 1**

Class Processor:

****

Class computer:  


Main:



**Questions**

Based on experiment 1, answer the related questions:

1. In the Processor class and Computer class, there are setter and getter methods for each attribute. What the purpose of method setters and getters are ?

-To protect attributes from direct access.The attributes can only be accessed and modified through getter and setter methods, allowing better control over the validation and restriction of attribute value changes.

2. Within the Processor class and Computer class, there are each a default constructor and a parameterized constructor. How is the use of the two types of constructors different ?

-Default constructor to not set any value in first initital but it use to set value later

- parameterized constructor to set value in first initial it can reduces the chances of using uninitialized attributes.

3. Consider the Computer class, which of the 2 attributes (brand and proc), which attribute is object type ?

-proc: it because this attribute is of the type Processor, which is a class

4. Look at the Computer class, which line shows that the Computer class has a relation with the Processor class ?

- private Processor proc;

- This line establishes that the Computer class has an association with the Processor class. The attribute proc is of type Processor

5. Pay attention to the Computer class, What is the syntax of proc.info() ?

- calling a method of an object (the proc object of type Processor), which outputs the details of the processor (brand and cache memory)

6. In the MainExperiment1 class , there is a line of code:

Computer c = new Laptop("Thinkpad", p);.

* What is p ?

- p is an instance of the Processor class, initialized with the brand "Intel i5" and a cache size of 3.0.

And what happens if the line of code is changed to:

Computer c = new Laptop("Thinkpad", new Processor("Intel i5",

3));

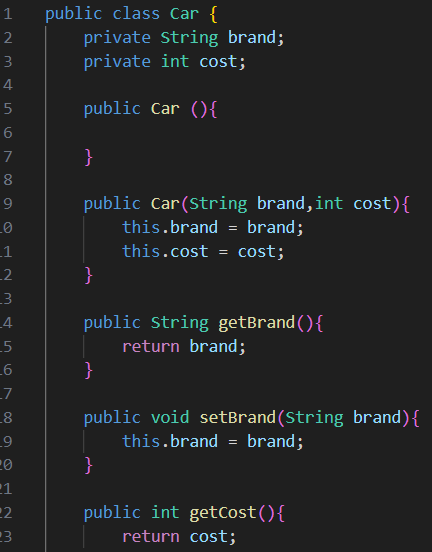
* How are the results of the program running, are there any changes ?

-The output is no change,it still same as before

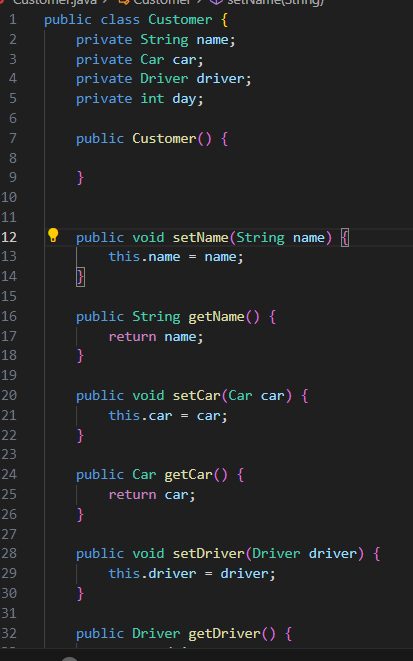
-But in the code work it change ,it make new instance of Processor which is created with the same parameters ("Intel i5", 3)

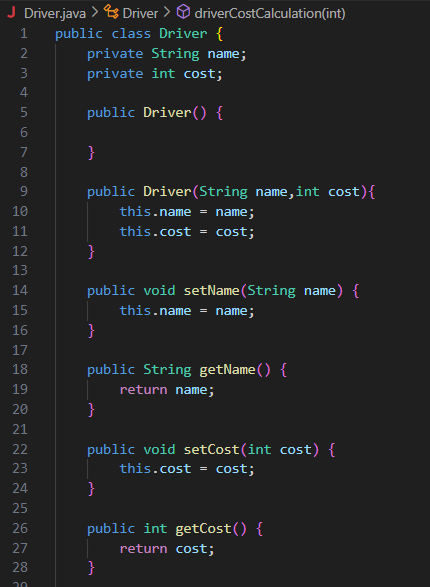
* **Experiment 2**

Class car:



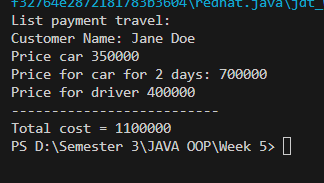
Class Customer:



Class driver:

Class main:  
 

Result:



Questions

1. See the Customer Class. In Which program line that shows the Customer class has a

relation with the Car class and Driver class ?

- private Car car;

private Driver driver;

2. Pay attention to the method of calculating the Cost of Driver in the Driver class, and the

method of calculating the Cost of a Car in the Car class. Why do you think that method must

have a day argument ?

- because the total cost is dependent on the number of days the service

-To calculate the total cost, you need to multiply the cost per day by the number of days

3. Pay attention to the code from the Customer class. What do method for

car.carCostCalculation(day) and

driver.driverCostCalculation(day) ?

- for calculating the total cost of the car rental and driver hire, respectively, based on the number of days the customer

4. See the MainExperiment2 class. What are the code cust.setCar(c) and

cust.setDriver(d) ?

- cust.setCar(c): This set Car object with c (a car with the brand "Avanza") with the Customer object cust. It links the customer to the car they are renting.

- cust.setDriver(d): This set the Driver object with d (a driver named "John Doe") with the Customer object cust. It links the customer to the driver they are hiring.

5. See the MainExperiment2 class. What do method for

cust.totalCostCalculation() ?

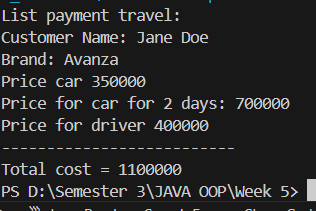
- responsible for calculating the total cost that the customer incurs for renting a car and hiring a driver over a specified number of days.

6. See the MainExperiment2 class, try adding to the last line of the main method and observe

the changes as they run!

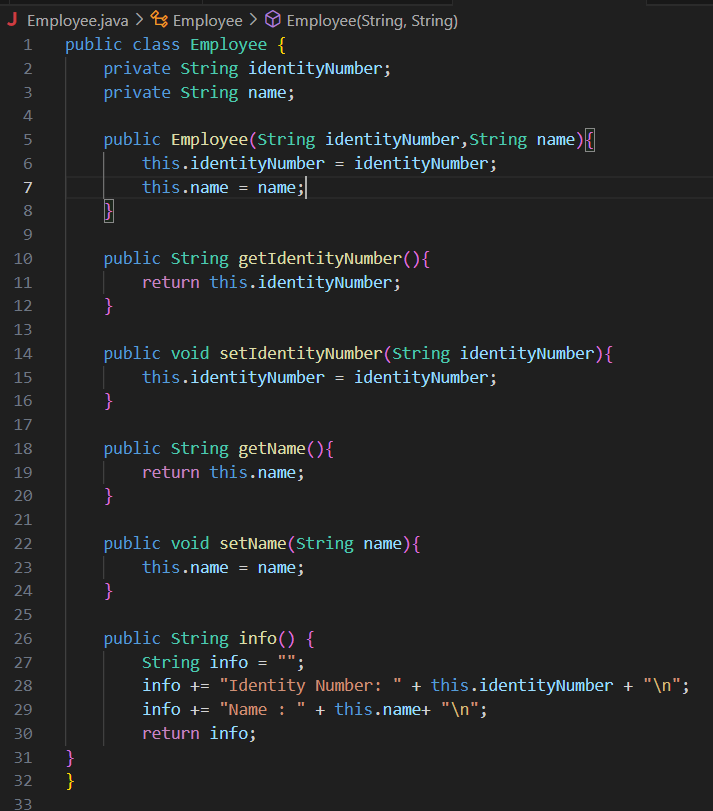
System.out.println(cust.getCar().getBrand());

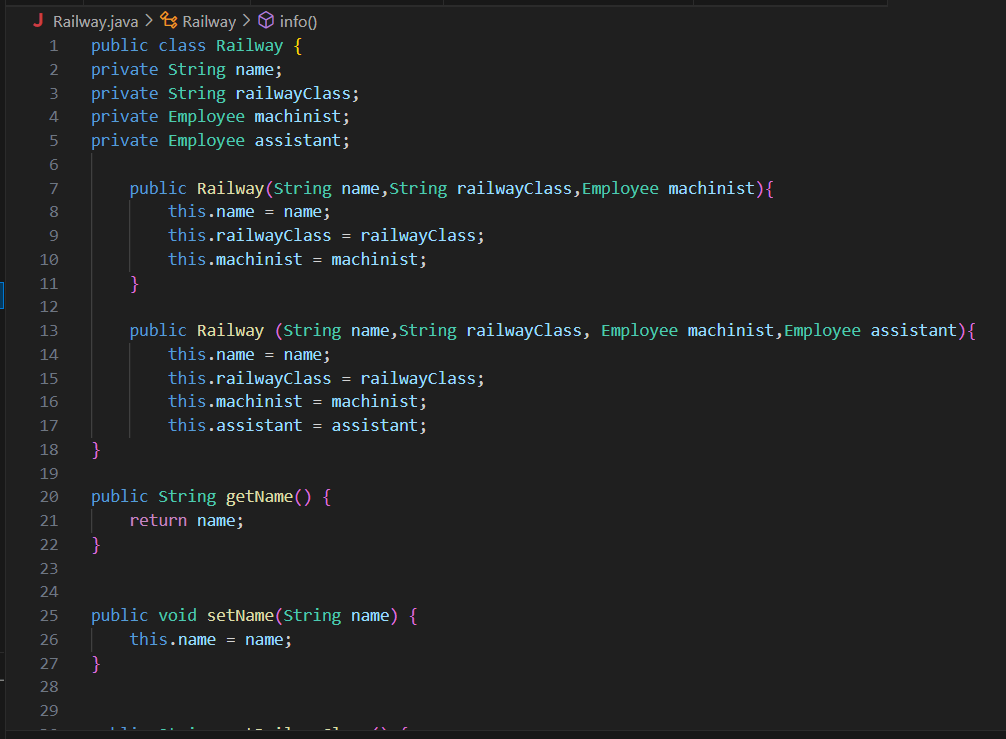
So what is the syntax for cust.getCar().getBrand() in the main method ?



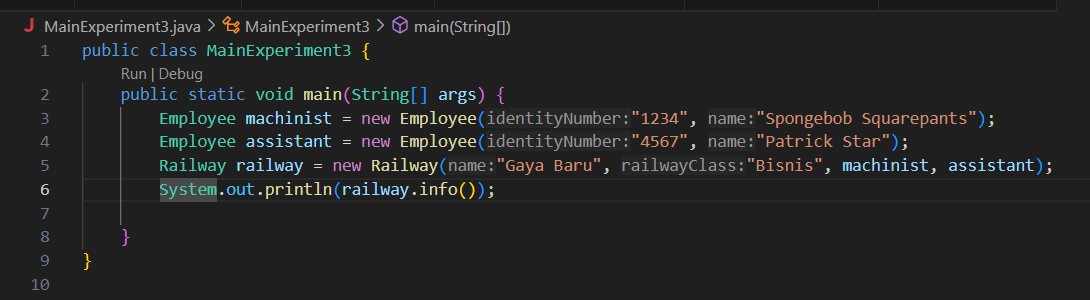
-To get what brand car that customer order

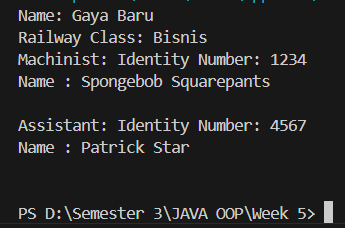
* **Experiment 3**

Class Employee:

Class Railway:  


Class main:



Output:

Questions

1. The info() method in the Railway class, the line of code this.machinist.info() and this.assistant.info() is used for what ?

- to delegate the responsibility of printing or displaying specific details of the machinist and assistant employees to their respective Employee objects

2. Create a new main program with the name MainQuestion class in the same package. Add the following code to the main() method !

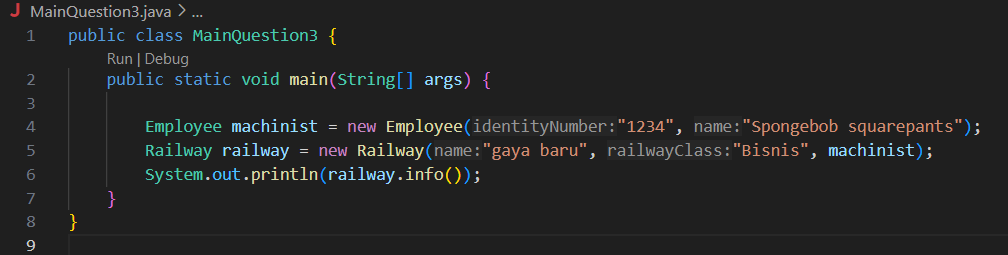
Employee machinist = new Employee("1234", "Spongebob

Squarepants");

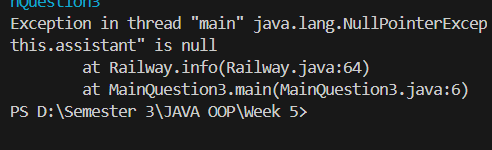
Railway railway = new Railway("Gaya Baru", "Bisnis",

machinist);

System.out.println(railway.info());

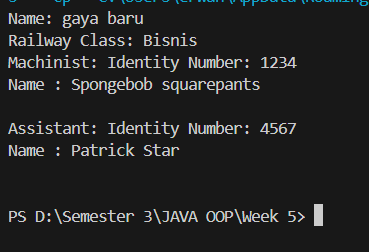
- 

3. What is the output from the main program ? Why did this happen ?

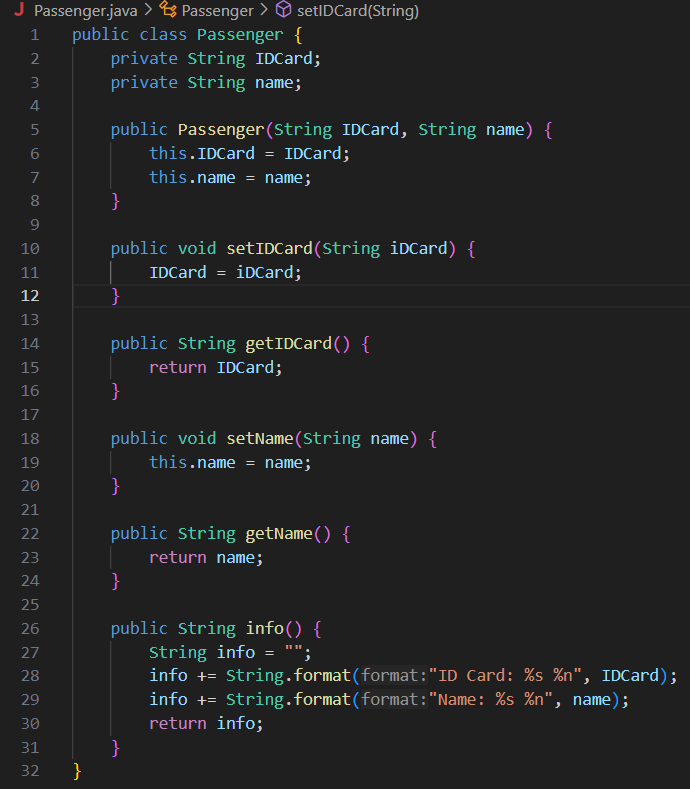
- 

-because the railway info is include assistant info,that’s why we must create assistant

4. Fix the Railway class so the program can run !

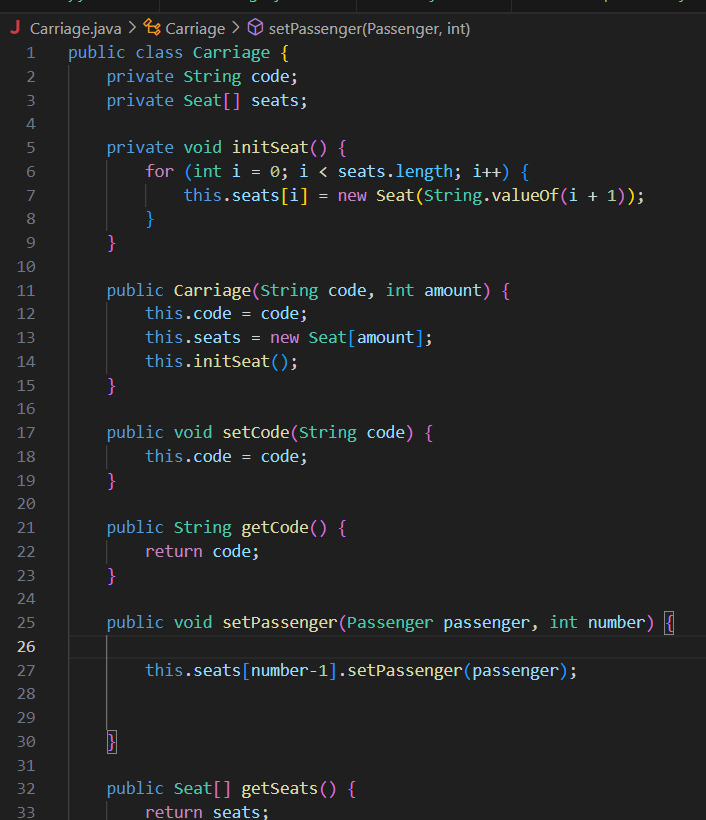


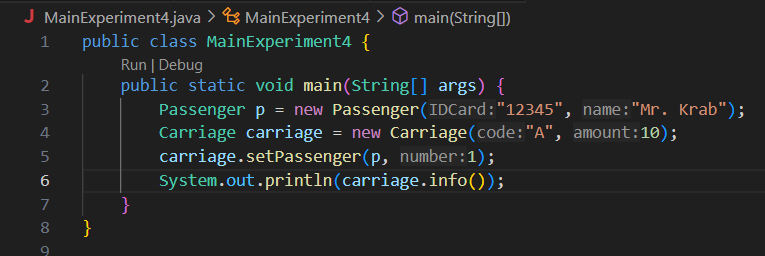
* **Experiment 4**

Class Passenger:

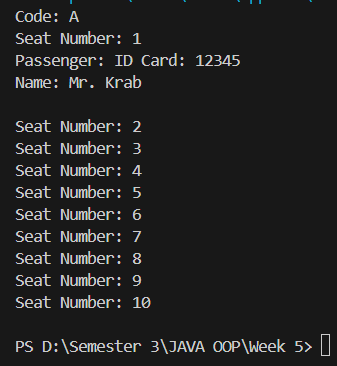
Class Seat:



Class carriage:

Main:  


Output:



Questions

1. In the MainExperiment4 class, what is amount of seats in carriage A ?

- 10 seats in the carriage A

2. Pay attention to the code snippet in the info() method in the Seat class. What does the

code mean ?

...

if (this.passenger != null) {

info += "Passenger: " + passenger.info() + "\n";

}

...

- The line if (this.passenger != null) checks if the passenger attribute of the Seat object is not null.if not null calls the info() method of the Passenger object: passenger.info()

-If the passenger is null, it means no passenger has been assigned to the seat. If it is not null, then a Passenger object is present in the seat.

3. Why the setPassengers() method in Carriage class, the value of number is reduced by the number 1 ?

-because first element in an array is at index 0, the second element is at index 1, and so on. In seat numbering in a train, the first seat would be labeled as seat 1, not seat 0.

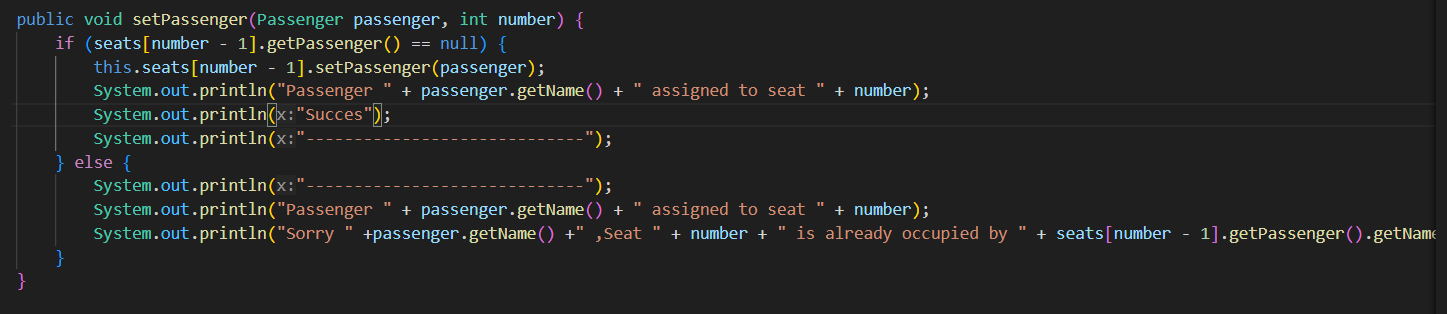
By subtracting 1 from the number, the program maps the user-friendly seat number (starting from 1) to the correct zero-based index in the seats[] array.

4. Instantiation of new budi object with the Passenger type, then insert the new object in

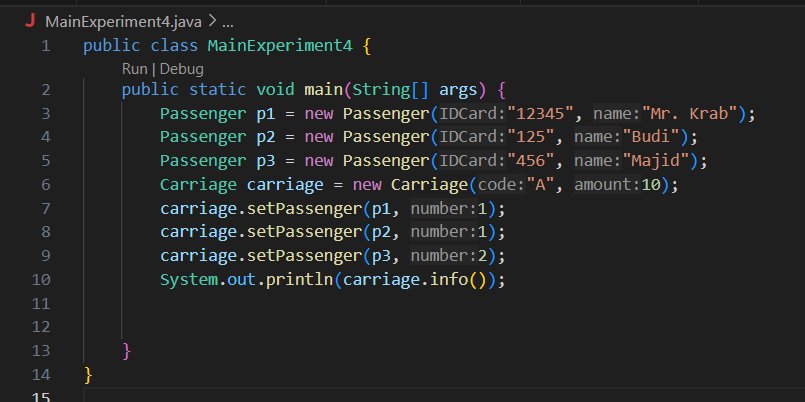
the carriage with the carriage.setPassenger(budi, 1) method. What’s happening ?

-error

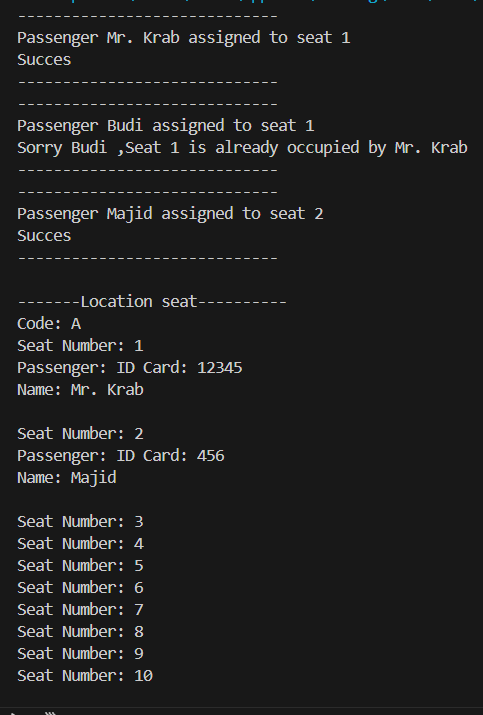


5. Modify the program so that it is not allowed to occupy the seat of another passenger !

Main:



Output:



* **IV. Assignment**

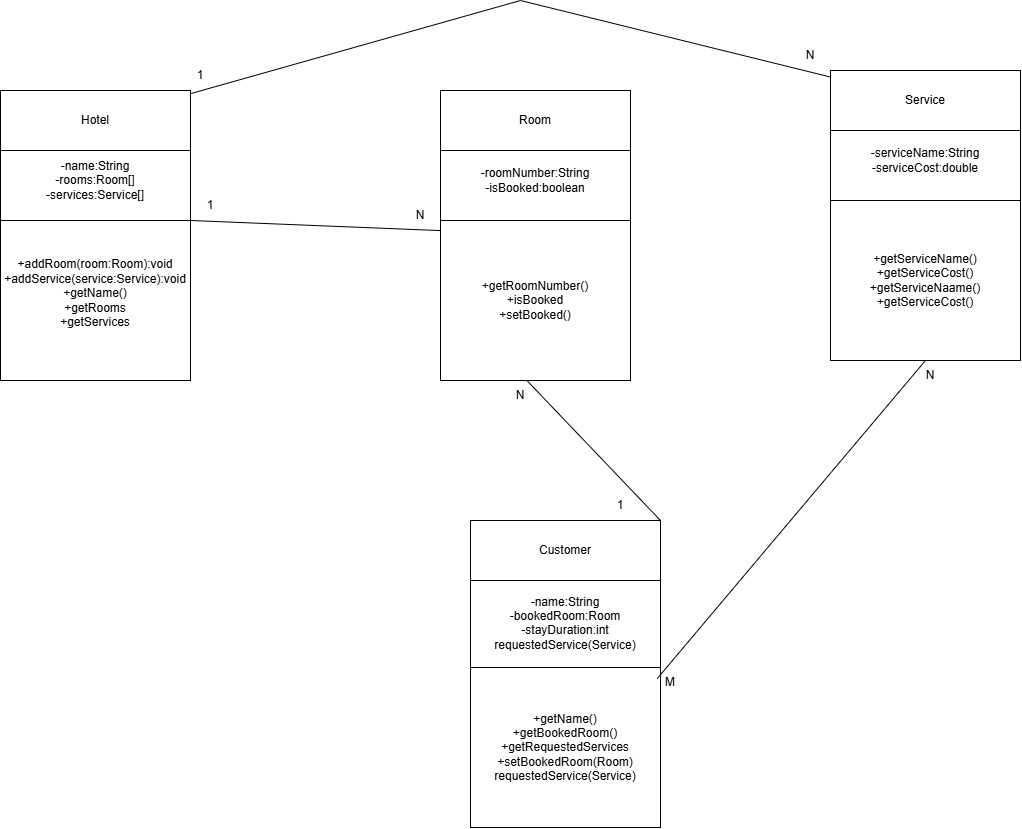
Create a case study, design a class diagram, then implement in the code! The case studies should

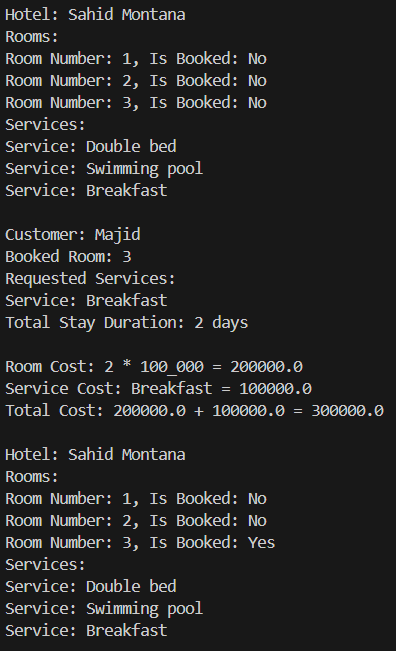
represent the relation class of experiments have been done on this matter, involving at least 4 classes

(the main class does not count)

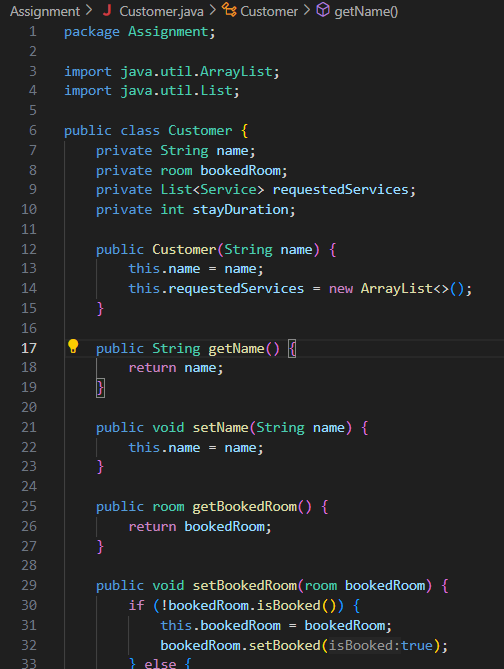
**-System booking hotel sahid Montana**

(Hotel,Room,Service,Customer)

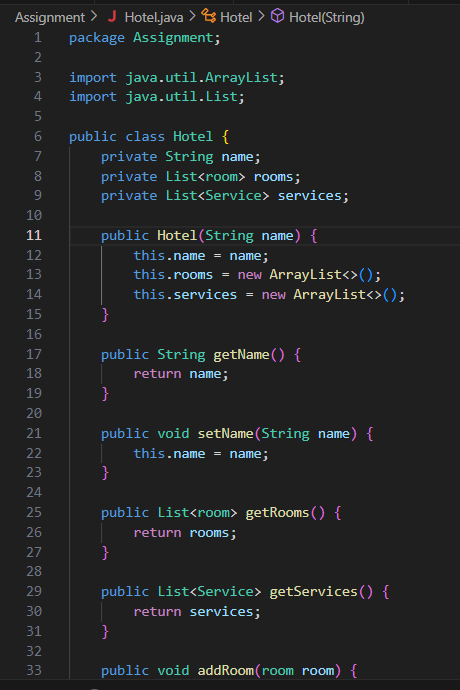


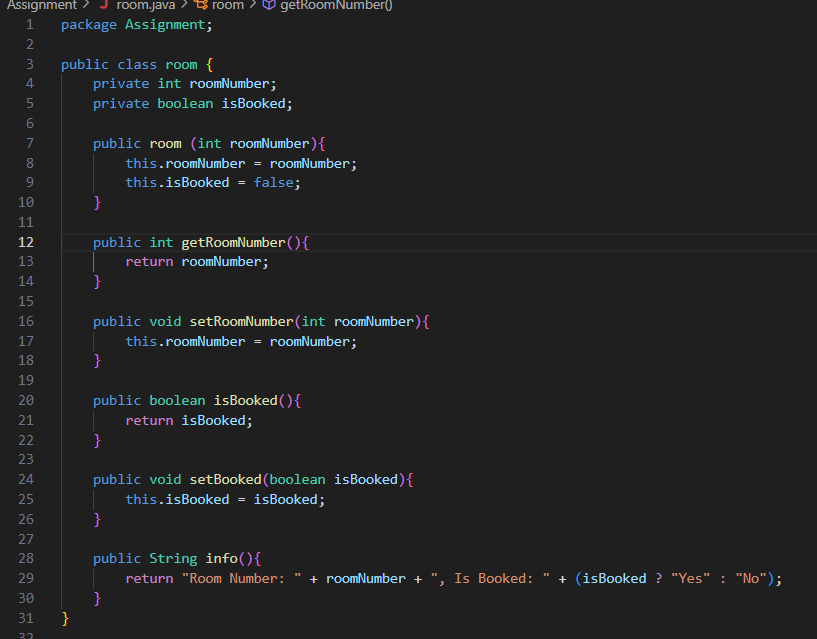
Output:

Customer:

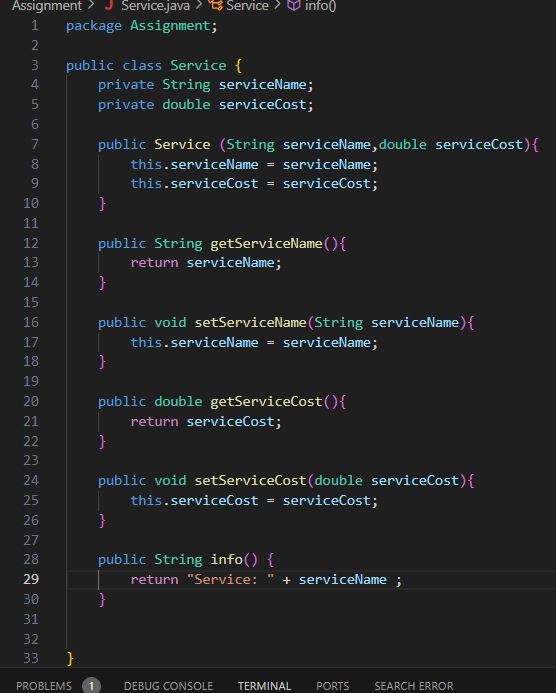


Hotel:

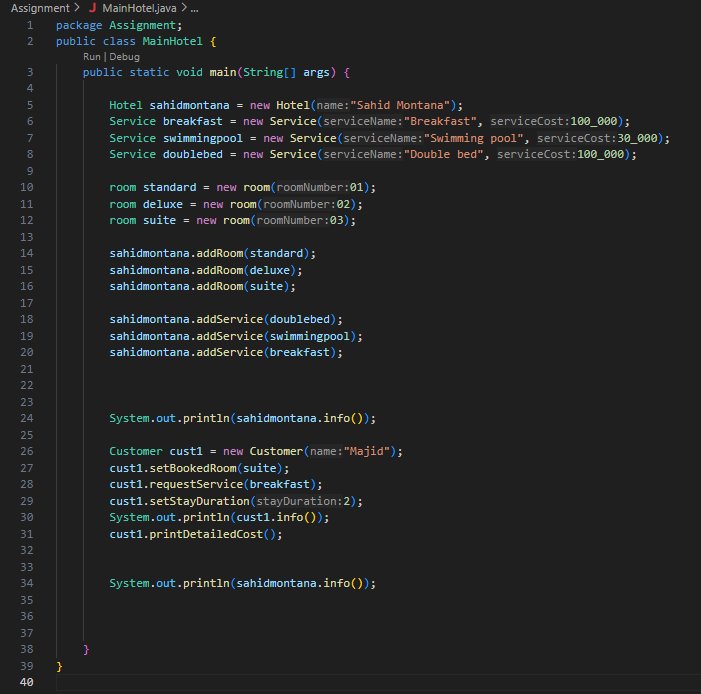


Room:

Service:



Main:



-full code on github