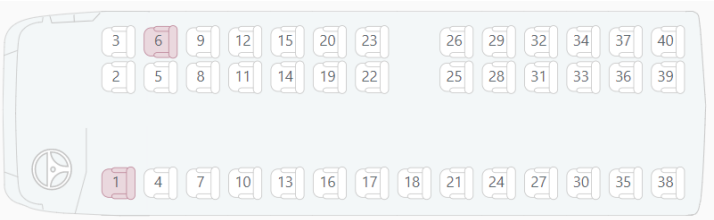
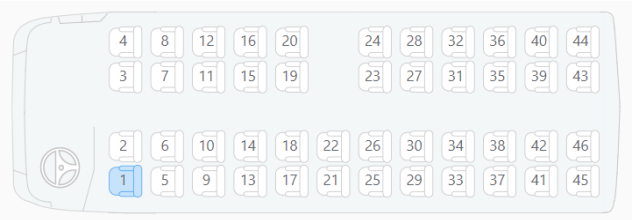
Homework 3 & 4

You are working for a bus company as a software developer who is in eager in object-oriented development. Your task is to develop flexible software for the ticket management system. Your design must allow creating a bus class with different seat layouts. You can use arrays (1D or 2D or 3D) to store seat objects such as left seats, right seats, and rear seats. For the sake of simplicity, I removed the last row from the bus layout (see below figures) so now you only have left and right rows. During the construction of the bus object, you must also consider the position of the middle door. You are free to use any numbering schema for the seats, but of course front seats will have lowest seat number and back seats will have highest seat numbers. Note that seat number 1 will always be placed behind the driver's seat. The Bus class will be abstract, and you will extend it to below 2 bus types. But a new bus type can be added as well. Implement bus (seat) layout mechanism in the abstract Bus class and extended class will simply call the proper constructor of the abstract Bus class with proper parameters.





Each seat object can be reserved for a ticket. In your system, passengers cannot decide which seat to buy but they can prefer the row placement, the system will automatically decide the seat number. Row placement is 0.0 for front rows and 1.0 for back rows. If ticket reference in a seat object is *null,* then the seat is not sold yet. Keep the price on the ticket object. A single passenger or multiple passengers (in array) can buy a ticket. If a single passenger tries to buy a ticket, then his or her neighbor's seat must be sold to a passenger of the same gender. If multiple passengers try to buy tickets, then they can buy a row of seats freely. So, you will have to create a method with the same name which accepts a passenger or array of passengers. But, for example, if 3 passengers try to buy a ticket and 2 of them sit in a row with 2 seats, then same-gender rule will still apply for the third passenger.

The Bus class must provide certain functionalities such as bus-type, plate-number, number-of-seats, number-of-free-seats, sell-seat (with number), and make-seat-free for single or multiple passengers. Some of these functionalities (or maybe all of them) must be abstract in Bus class and therefore must be implemented in subclasses. Also, Bus class and/or its subclasses must return the bus layout (as shown in the above figure) as a string (override `toString()` method).

Bus, Ticket, Passenger classes and their subclasses must be placed in BusTicketSystem package. You can also use interfaces if you find it useful or necessary. Try to use encapsulation and other object-oriented design practices. Your test class, which includes the `main()` method, must be in the default package. In this main method, write necessary test cases to show capabilities of your design and implementation.

Please also read and follow below guidelines:
https://www.geeksforgeeks.org/java-naming-conventions/
https://blog.alexdevero.com/6-simple-tips-writing-clean-code/

You will work as 2-students team (only one student will submit the code).
https://en.wikipedia.org/wiki/Pair_programming
Your submission will be Name1_Surname1,Name2_Surname2.zip which only contains Java files.