SYSC 2004
Winter 2018
Lab 3

Objectives

1. Programming Java arrays.
2. Copying

Submission Requirements: (Exact names are required by the submit program)

- **Without a submission you will not get a mark**
  Seat.java, Car.java and Train.java

A passenger train is made up of one or more business-class cars and economy-class cars. Each car contains several seats. When you purchase a ticket for a train trip, a specific seat is booked for you. Purchasing a ticket to ride in a business-class car costs more than a ticket for a seat in an economy-class car.

Part 1 - Exploring class `Seat`

1. You are provided with a mostly-complete implementation of the `Seat` class, along with its JUnit test class called `SeatTest`.java. Download these files from CULearn.
2. Create a project *train*.
   - Do NOT click the checkbox to create a `main`() method.
   - If you do, don't worry. We just won't use that file.
3. Drag-and-drop the provided files to your project's **Source Package.**
4. Run the test – **Right-click** on `SeatTest` and select **Run**.
5. Edit the `Seat` class to now properly implement its `toString`() method. Keep making changes until the `SeatTest` passes all tests.
6. Within the `Seat` class, add a `main`() method. We will use this as practice in manipulating Seat objects.

```
public static void main(String[] args) { // TBD }
```

7. One by one, complete the following tasks. **Please** don't do them all and then run. **Please learn how to code incrementally**. Write the code, run. Write more code, run again.

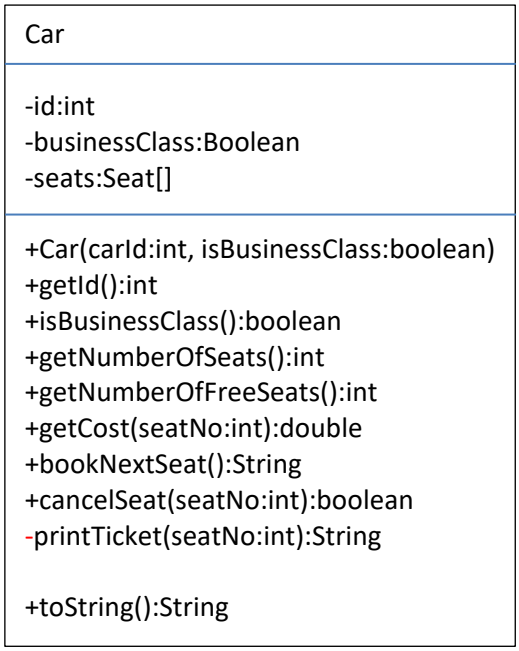
   - Create a `seat` object that represents Seat #5, with a ticket price of $25.00.
   - Invoke `number()`, `price()` and `isBooked()` on the `seat` object. Print out their values.
   - Invoke `book()` to reserve the seat. Remember to assign this instance method's return value to a variable. What is the meaning of the value returned by this instance method?

- Invoke `isBooked()`. Explain the value returned by this instance method. Print out the return value.
- Even though the seat has been booked, invoke `book()` a second time. Again, print and examine the return value. What is the meaning of the value returned by this instance method?
- Invoke `cancelBooking()` to cancel the `seat`'s booking. What is the meaning of the value returned by this instance method?
- Invoke `isBooked()`. Explain the value returned by this method.
- Even though the booking has been cancelled, invoke `cancelBooking()` a second time. What is the meaning of the value returned by this method?

Note: The previous list of tasks is an example of how to test an object. Notice how methods are called both to succeed and to fail! Watch-and-learn. On a midterm, you may be asked to demonstrate how to test a class.

Part 2 - Developing Class `Car`

The *train* project contains an incomplete implementation of a class that models passenger cars in a train.



Note: printTicket() is a private method!

Not shown in the UML are four constants that define the price of tickets and the number of seats in economy-class cars and business-class cars

In this part, **your job is to complete the implementation of `Car`.** You cannot define additional constructors or methods in either `Seat` or `Car`, or change the specification of any of the constructors and methods. All fields must be private; i.e., it is not permitted for objects of one class to directly access the fields of objects of another class.

You will test your `Car` class using the provided JUnit `CarTest` class. At this point in the course, you are not expected to understand all of the code in `CarTest` (in particular, the use of methods `assertEquals()`, `assertTrue()` and `assertFalse()`), but you are welcome to try to figure out what each of the test methods does. Please build incrementally, passing one test at a time. You will save yourself time and pain.

Special notes about your implementation (given only when a method is not self-evident):

1. When working with classes that have an instance variable that is an array, the constructor normally initializes the array. In this case, initializes the `seats` field with a new list (array) of seats. There are 30 seats in a business-class car, and 40 seats in an economy-class car. Remember that each element in the list must then be initialized with a new `Seat` object. The seats in each car are numbered consecutively, starting from 1. The cost of a ticket for a seat in a business-class car is $125, while the cost of a ticket for a seat in an economy-class car is $50. If your constructor works, `testCreateBusinessCar()` and `testCreateEconomyCar()` should pass.

2. Method `bookNextSeat()`searches the list of seats in the car and reserves the first available seat that it can find. After finding a free seat and booking it, this method should print and return a ticket by invoking `printTicket()`. If no seats in the car are available, this method returns `null`.

3. Method `cancelSeat()` cancels the booking for the given `seat`. It returns `true` if this seat number is valid and if this `seat` has been previously reserved. It returns false if the seat number is not valid or if this `seat` has not been previously reserved.

Part 3 - Developing Class `Train`

Students and TAs: A mark of SATISFACTORY for this lab does NOT require completion of Part 3. This part is included for good students during the lab, and for everyone for study purposes.

As one further practice in initializing arrays, you will complete the implementation of a client class called `Train`. An empty template is given for you. Follow the instructions given as comments in this file.