

Lab 9 Requirements

Create a new Eclipse workspace named "**Lab9_1234567890**" on the desktop of your computer (replace **1234567890** with your student ID number). For each question below, create a new project in that workspace. Call each project by its question number: "**Question1**", "**Question2**", etc. If you do not remember how to create a workspace or projects, read the "*Introduction to Eclipse*" document which is on iSpace. Answer all the questions below. At the end of the lab, create a ZIP archive of the whole workspace folder. The resulting ZIP file must be called "**Lab9_1234567890.zip**" (replace **1234567890** with your student ID number). Upload the ZIP file on iSpace.

Question 1

Create a class **Door** with the following UML specification:

```
+-----+
|           Door           |
+-----+
| - isOpen: boolean       |
+-----+
| + Door()                |
| + isOpen(): boolean     |
| + open(): void          |
| + close(): void         |
| + testDoor(): void      |
+-----+
```

where **isOpen** is an instance variable indicating whether the door is currently open or not. The default state of a door when it is created is to be closed.

Then create a class **Car** with the following UML specification:

```
+-----+
|           Car           |
+-----+
| - name: String          |
| - doors: Door[]         |
+-----+
| + Car(String name, int numberOfDoors) |
| + listDoors(): void      |
| + countOpenDoors(): int  |
| + openOneDoor(int doorNumber): void   |
| + testCar(): void       |
+-----+
```

Trying to create a car with less than one door must throw a **BadCarException** with the message "**A car must have at least one door!**". When a car is created, all its doors are closed.

The method **listDoors** prints one line of output for each door of the car. Each line starts with the name of the car followed by a message indicating whether the door is currently open or closed. For example, if a car has the name "**Tiny**" and has two doors, one which is currently open and one which is currently closed, then calling the **listDoors** method of the car object should produce the following output:

```
Tiny: door is open
```

Tiny: door is closed

The method **countOpenDoors** counts the number of doors of the car which are currently open.

The method **openOneDoor** opens a specific door, as indicated by the door's number. Doors are numbered starting from 1 (not 0!) Trying to open a nonexistent door should throw a **BadDoorException** with the message "**Door XXX does not exist!**", where **XXX** is replaced with the number of the nonexistent door. Trying to open a door which is already open does nothing.

Use enhanced **for** loops as much as possible instead of using normal **for** loops.

Also add to your program a **Start** class with a **main** method to test your program.

Question 2

Add a method **changeAllDoors** to the **Car** class that opens all the closed doors of the car and closes all the open doors of the car.

Question 3

Add a method **replaceDoor** to the **Car** class that takes as argument a door number, and replaces the existing car door (as indicated by the door number) with a completely new door.

Remember that doors are numbered starting from 1 (not 0!) Trying to replace a nonexistent door should throw a **BadDoorException** with the message "**Door XXX does not exist!**", where **XXX** is replaced with the number of the nonexistent door.

Question 4

Add a method **replaceAllDoors** to the **Car** class that takes no argument and replaces all the existing car doors with completely new doors. Please don't call **replaceDoor** in this question.

What happens if you try to use an enhanced **for** loop?

Question 5

Add a method **replaceManyDoors** to the **Car** class that takes as argument an integer **numOfDoorsToReplace** that indicates the number of existing car doors that should be replaced with completely new doors (starting from the first door in the array). Please don't call **replaceDoor** in this question.

What happens if you try to replace more doors than the car has? Add a test to the **testCar** method to test this case. Make sure the test does not kill the program. (You do not need to do anything else, only test the case).

Question 6

Add a method **expandCar** that adds two new doors to the car. The existing doors should not be modified (the same **Door** objects must be used as before the car was expanded).

Question 7

Change the **Car** class to use an **ArrayList** instead of using an array. Use generics for the arraylist. Make sure all your tests still work, including the tests for exceptions.