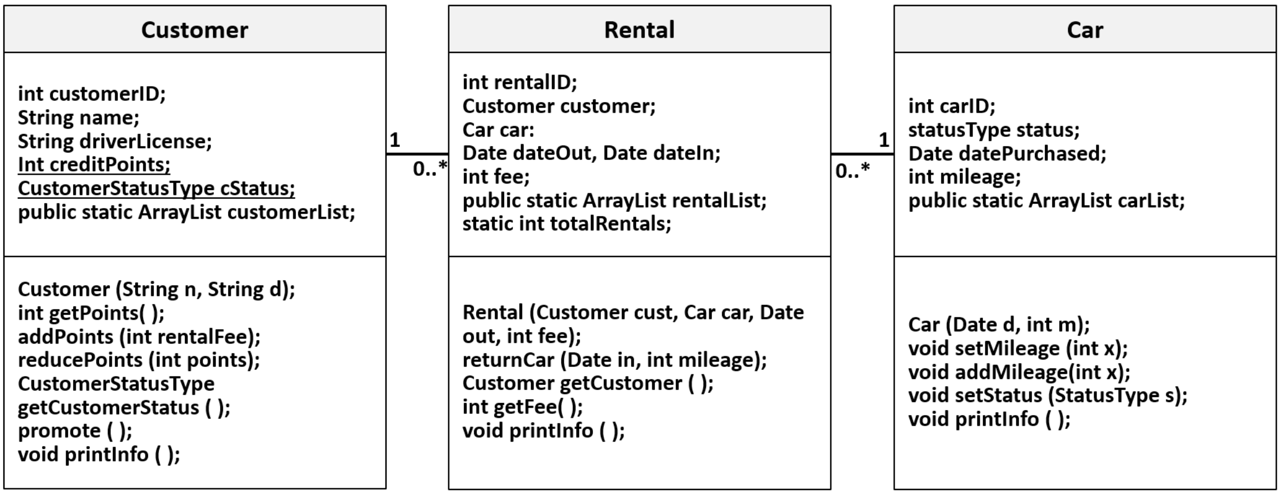
Laboratory 5. Inter-class Relationship

(Due: Specified on Smart Campus)

Spring 2022
Soo Dong Kim

1. Write a Java program which implements the following 3 classes and their relationships.



Customer

```
int customerID;
String name;
String driverLicense;
Int creditPoints;
CustomerStatusType cStatus;
public static ArrayList customerList;
```

```
Customer (String n, String d);
int getPoints( );
addPoints (int rentalFee);
reducePoints (int points);
CustomerStatusType
getCustomerStatus ( );
promote ( );
void printInfo ( );
```

Rental

```
int rentalID;
Customer customer;
Car car;
Date dateOut, Date dateIn;
int fee;
public static ArrayList rentalList;
static int totalRentals;
```

```
Rental (Customer cust, Car car, Date
out, int fee);
returnCar (Date in, int mileage);
Customer getCustomer ( );
int getFee( );
void printInfo ( );
```

Car

```
int carID;
statusType status;
Date datePurchased;
int mileage;
public static ArrayList carList;
```

```
Car (Date d, int m);
void setMileage (int x);
void addMileage(int x);
void setStatus (StatusType s);
void printInfo ( );
```

- ❑ Customer class
  - ❍ *CustomerID* is a unique 5-digit identification number for customers. The system generates this ID automatically using a random number generator inside the constructor. Check for a potential conflict with existing ID numbers.
  - ❍ *Name* is the name of the customer.
  - ❍ *Driverlicense* is the driver license of the customer.
  - ❍ *creditPoints* is the credit points earned from the customer' rental history. That is, customers earn credit points when returning cars. The point earning rates are slightly different for different levels of *cStatus*, as shown in the following table.

	Point Earning Rate
Silver	0%
Gold	10%
Diamond	20%

  - ❍ *CustomerStatusType* is an enumerated datatype, {*Silver, Gold*, *Diamond*}, representing the status of each customer.
  - ❍ *cStatus* represents the current status of the customer. initialize it as *silver* in the constructor.'
  - ❍ *customerList* is the list of all the customer instances created. Inside the constructor, each newly generated *Customer* instance is stored in this static array list.
  - ❍ *Customer( )* is the constructor that takes the parameters of the customer name and the driver license number. It assigns a customer ID and initializes the *creditPoints* with 0 and *cStatus* as *Silver*.

- *promote( )* method is to evaluate the rental history of the customer and change their *cStatus* the *total amount of rental fees paid*. For example, a customer with 300,000 wons for the total amount of rentals paid will be promoted to 'Gold' status.

	Minimum Amount of Total Rental Fees paid
Silver	0
Gold	100,000
Diamond	500,000

- printInfo( )

  This method is to print the information of the customer.

- Car class

  - *CarID* is a unique 4-digit number identification number for cars, i.e., the range of 1,000 ~ 9,999." The system generates this ID automatically using a random number generator inside the constructor. Check for a potential conflict with existing ID numbers.
  - *StatusType* is an enumerated datatype representing the status of each car. The datatype is defined with the values of (*available, checkedOut*, *inService*, *discarded*, and *sold*)." Initialize the value of *StatusType* with '*available'* inside the constructor.
  - *Mileage* is to represent the miles (or kilometers) that the car has been driven so far.
  - *carList* is the list of all the car instances created. Inside the constructor, each newly generated *Car* instance is stored in this static array list.
  - The 'd' and 'm' for the constructor are the initial values of *datePurchased* and *mileage* attributes respectively. Utilize *Date* and SimpleDate classes of Java library.
  - *Car( )* is the constructor that takes the parameters of the date purchased and the current mileage on the car. It assigns a Car ID and initializes the s*tatus* with 'available'.
  - printInfo( )

    This method is to print the information of the car.

- Rental class

  Rental class is to represent rentals made by customers. Each *Rantal* instance has a reference to a customer and a reference to a rental car.

  - *rentalID is* a unique 6-digit identification number for rentals. The system generates this ID automatically using a random number generator inside the constructor. Check for a potential conflict with existing ID numbers.
  - *customer* is an object reference to the customer for the current rental.
  - *car* is an object reference to the rental car for the current rental.
  - *dateOut* represents the date that the rental car is checkedout.
  - *dateIn* represents the date that the rental is returned.
  - *fee* reprenets the rental fee charged for the rental.
  - *rentalList* is the list of all the rental instances created. Inside the constructor, each newly generated *Rental* instance is stored in this static array list.

- *Rental( )* is the constructor that takes the parameters of the customer ID, the car ID, the checkout date, and the rental fee.
  A customer may use the accumulated points to pay for the rental fee. Inside this consturctor, the system asks how many points the customer will use for a rental, by displaying the accumulated points. Then, it uses the points specified by the customer for the rental fee and invokes *reducePoints( )* of *Customer* class to deduct the points.
- *returnCar( )* is to calculate the points to earn using the earning table and add the earned points with *addPoints( )* of *Customer* class.
- *getCustomer( )* returns the reference of the customer.
- *getFee( )* returns the rental fee for the rental.
- printInfo( )

  This method is to print the information of the rental.

2. Define a main( ) method in *Rental* class that performs the followings;

- ❑ Creates 2 *Customer* instances using appropriate input parameter values.
- ❑ Print the information of all the customers.
- ❑ Creates 3 *Car* instances using appropriate input parameter values.
- ❑ Print the information of all the cars.
- ❑ The customer #1 makes a rental with the rental fee of 500,000.
  The customer #2 makes a rental with the rental fee of 2,000,000.
- ❑ Print the information of all rentals.
- ❑ Customers return their rental cars by invoking *returnCar( )*.
- ❑ Run *promote( )* for all the customers, which determines the status of the customers.
- ❑ Print the information of all customers.
- ❑ The customer #1 makes a new rental with the rental fee of 5,000,000.
- ❑ The customer #1 returns the car by invoking *returnCar( )*.
- ❑ Print the information of the customer #1.

3. Submission Guidelines

- ❑ Submit your solution on the web site; *class.ssu.ac.kr*
- ❑ Submit just **1 PDF file** containing the followings;
  - ❍ Java Source Code, *.java* file
  - ❍ Screenshot showing the program output
- ❑ Use this filename convention for your submission; **OOP.LAB.##.홍길동.pdf.** where ## is the laboratory number in 2 digits.
- ❑ No Plagiarism
  - ❍ The laboratory is an individual exercise. Do not copy others.
  - ❍ Submit your original work.

4. Grading Criteria (Total of 10 Points)

- ❑ Quality of Program (6)
  - ❍ Program Structure (4)
  - ❍ Exception Handling (1)
  - ❍ Header in the Source Program and Comments on Code (1)
- ❑ Accuracy of Output (4)
  - ❍ Correctness of Output Values (3)

    The output must be correct according to the problem specification.
  - ❍ Comprehensive Output Format (1)

    The output should be readable and comprehensive.
    Copy only the output part from the screen. (No Entire Screen)