Object-Oriented Programming Lab#8, Fall 23

Today's Topics

- Inheritance
- Encapsulation
- Polymorphism
- Abstraction

ArrayLIst:

Action	Code
Creating an ArrayList	ArrayList<T> list = new ArrayList<T>();
Adding element to arraylist	list.add(T t);
Adding multiple elements to arraylist	list.addAll(ArrayList<T> t);
Remove an element	list.remove(int index) list.remove(T t)
Remove multiple elements from an arraylist	list.removeAll(ArrayList<T> t);
Accessing an element	List.get(int index)
Size of arraylist	list.size();

Problems/Assignments – Online Reservation System

Create an Online Reservation System to help the customer and owner of the item to streamline the booking process and enhance customer experience. The system should allow customers to easily make reservations for a hotel, restaurant, vehicle through an intuitive and user-friendly interface. There will be 2 types of customers to this system; admin and customer who wants to reserve the item. Admin has full control over the system. Customer has to create account if he/she wants to reserve an item.

Here is the list of the classes to implement the Application

A. **Create a project name ReservationSystemLibrary and add the following class to this project**

1. **Item** Class - abstract class and under **uap** package
    a. **Attributes** (all private): id, ~~description~~, ~~category~~, ~~ArrayList<String> features,~~ rate, isAvailable, ~~owner~~, reservedBy
    b. **Constructor** – pass parameter for all except **isAvailable** and **reservedBy**. Inside the constructor, initialize the attributes with respective parameters and set the **isAvailable** to true.
    c. **Methods**:

Method Header	What the method should do?
add getter methods for all attributes	

add setter method for rate, isAvailable, and reservedBy	
public void reserveItem(String reservedBy)	set the **isAvailable** to false and **reservedBy** using the setter methods.
public void reservationOver()	set the **isAvailable** to true and **reservedBy** to **null** using the setter methods.
public void cancelReservation()	set the **isAvailable** to true and **reservedBy** to **null** using the setter methods.
public abstract double getPayment(int quantity)	An abstract method.
public String toString()	Return the attribute values as String.

2. **Vehicle** class – a subclass of **Item** class and under **uap** package
    a. **Attributes** (all private): model, capacity, enginePower
    b. **Constructor** – pass parameter for all except **isAvailable** and **reservedBy**. Inside the constructor, call parent's constructor and initialize the rest of the attributes with respective parameters.
    c. **Override** getPayment(int quantity) method – return the total payment which is the multiplication of rate attribute and quantity parameter

3. **HotelRoom** class – a subclass of **Item** class and under **uap** package
    a. **Attributes** (all private): hotelName, rankOfHotel, occupancy, hasAC
    b. **Constructor** – pass parameter for all except **isAvailable** and **reservedBy**. Inside the constructor, call parent's constructor and initialize the rest of the attributes with respective parameters.
    c. **Override** getPayment(int quantity) method – return the total payment which is the multiplication of rate attribute and quantity parameter.

4. **Restaurant** class – a subclass of **Item** class and under **uap** package
    a. **Attributes** (all private): restaurantName, capacity, occupied
    b. **Constructor** – pass parameter for all except **isAvailable, reservedBy,** and **occupied**. Inside the constructor, call parent's constructor and initialize the rest of the attributes with respective parameters.
    c. **Override** getPayment(int noOfGuests) method – return the total payment which is the multiplication of **rate** attribute and **noOfGuests** parameter.

5. **ReservationSystem** class (under **uap** package):
    a. **Attributes** (all private): name, ArrayList<Item> items
    b. **Constructor-** pass parameter for name. Inside the constructor, initialize the name attribute with the parameter and instantiate the items arraylist.
    c. **Methods**:

Method Header	What the method should do?
public void addItem(String id, double rate, String model, int capacity, float enginePower)	Create an object of **Vehicle** class using the parameters and add the object to ***items*** attribute/list

public void addItem(String id, double rate, String hotelName, int hotelRank, int occupancy, boolean hasAC)	Create an object of **HotelRoom** class using the parameters and add the object to ***items*** attribute/list
public void addItem(String id, double rate, String restaurantName, int capacity)	Create an object of **Restaurant** class using the parameters and add the object to ***items*** attribute/list
public ArrayList<Room> findRooms(String hotelName, int occupancy, boolean hasAC)	Loop through the ***items*** attribute and find the rooms that has matching attributes and return all those rooms as an arraylist. If no room found, an empty arraylist will return.
public ArrayList<Room> findRooms(int occupancy, boolean hasAC, int minRate, int maxRate)	Loop through the ***items*** attribute and find the rooms that has matching attributes and return all those rooms as an arraylist. If no room found, an empty arraylist will return.
public ArrayList<Vehicle> findVehicles(int capacity, int minRate, int maxRate)	Loop through the ***items*** attribute and find the vehicles that has matching attributes and return all those vehicles as an arraylist. If no vehicle found, an empty arraylist will return.
public ArrayList<Restaurant> findRestaurants(int noOfGuest, int minRate, int maxRate)	Loop through the ***items*** attribute and find the restaurants that has matching attributes and return all those restaurants as an arraylist. If no restaurants found, an empty arraylist will return.
public Restaurant findRestaurant(String restaurantName, int noOfGuest, int minRate, int maxRate)	Loop through the ***items*** attribute and find the restaurant that has matching attributes and return that restaurant. If the restaurant doesn't have enough free slot to accommodate the noOfGuests, return null.
public Item findItem(String id)	Loop through the ***items*** attribute and find the restaurant that has matching id. If no item found, return null
public void reservationComplete(String id)	Call ***findItem*** method. If the item is available, call ***reservationOver*** method of the **Item** class.
public void cancelReservation(String id)	Call ***findItem*** method. If the item is available, call ***cancelReservation*** method of the **Item** class.
public ArrayList<Item> getItems()	Getter method for **items** attribute.
public void viewAll()	Loop through the ***items*** attribute and print each item.
public void viewDetails(String id)	Call ***findItem*** method and print the item if the item is found.

B. **<u>Create another project name ReservationSystemApp and add the following class to this project</u>**

1. **App** class (under **uap.app** package):

a. Add main method, create an object of **ReservationSystem** class and provide menu for each method.
    i. Add Item (Vehicle/hotel room/restaurant) [It will be admin related functionality]
    ii. Search Item
    iii. View by category
    iv. Reserve
    v. Reservation Over
    vi. Cancel Reservation
    vii. Exit