CS116 Final Exam

(35 Points + 3 Bonus Points)

Boris Glavic

May 8th, 2019

1 Instructions

- All tasks require you to write Java code. After you are done upload all `.java` files to Blackboard to the assignment `finalexam`.
- The exam consists of three parts
  - Design and implement classes to model a restaurant domain
  - Design a class for managing a collection of restaurants and searching for restaurants
  - Write a test class for the restaurant guide

Hints:

- start with the first part since the second parts depends on it

2 Restaurants - designing and implementing classes (20 points)

Design and implement a set of classes for storing information about restaurants:

- **Restaurant** - a restaurant has a `name`, a `cuisine` (e.g., *french*), an `owner`, an `address` (simply use a `String` to store the address of a restaurant), and a `menu`.
- **FastFoodRestaurant** - a fast food restaurant is a specific type of restaurant for which we are also storing the name of the `franchise` the restaurant belongs to (e.g., *Burger King*)
- **Cafeteria** - a cafeteria is a restaurant that serves employees/students of a company, school, or university. For a cafeteria we store the `sponsor`, i.e., the company/school/university that runs the cafeteria
- **Menu** - a menu consists of menu items for which we record a `name`, `price`, and `category` (e.g., `appetizer` or `main course`). The constructor of `Menu` should create an empty menu. Implement a method that allows menu items to be added to the menu. Throw an exception if a menu item is added the has a price less than or equal to `0.0`.

Keep the following in mind:

- Constructors of classes should allow us to provide values for the fields of the class
- Use `private` fields and provide access to the fields through **getters** and **setters**
- Inheritance is used to factor out common parts and avoid code repetition. The relationship between a class and one of its subclasses is typically an *isA* relationship. For example, *a Car* ***is a*** *Vehicle*.

3 Restaurant Guide - class design and searching (10 points)

Design and implement a class `RestaurantGuide` which manages a collection of restaurants (choose the appropriate type of collection to use).

- The constructor of `RestaurantGuide` should create an empty guide (no restaurants)
- Write a method that allows new restaurants to be added to the guide
- Write a method that returns the list of restaurants covered by the guide
- Write a method `search` that takes as input the `name` of a menu item and returns a list of restaurants that have an item with this name on their menu

4 Restaurant + Guide - test cases (5 points)

As a test case write a class `RestaurantTest` whose `main` method performs the following tests:

1. It creates a new restaurant guide with two restaurants:
   (a) A *FastFoodRestaurant* with `name`: "Mac", `cuisine`: "junkfood", `owner`: "Peter", `address`: "10 W 33 st, Chicago, IL", `francise`: "MacDonalds" with two menu entries:
      i. `name`: "Fries" , `price`: 2.50 , `category`: "MainCourse"
      ii. `name`: "Burger" , `price`: 3.99 , `category`: "MainCourse"
   (b) A *Cafeteria* with `name`: "MTCC commons", `cuisine`: "junkfood", `owner`: "Alice", `address`: "3555 S State, Chicago, IL", `sponsor`: "IIT" with two menu entries:
      i. `name`: "Fries" , `price`: 3.50, `category`: "MainCourse"
      ii. `name`: "Burrito", `price`: 5.99, `category`: "MainCourse"
2. Use the search method of `RestaurantGuide` to find all restaurants that serve burritos (have a menu item named *"Burrito"*) and print the result to `System.out`.

5 Bonus (3 points)

Extend the `RestaurantGuide` to store a numerical rating (`1` to `5`) for each restaurant.

- write methods for setting and retrieving the rating for a restaurant
- write a method that returns the restaurant with the highest rating

6 Solutions

6.1 Restaurant

```java
/**
 *
 */
package solutions.finalexam;

/**
 * @author lord_pretzel
 *
 */
public class Restaurant {

	private String name;
	private String cuisine;
	private String owner;
	private String address;
	private Menu menu;

	public Restaurant (String name, String cuisine, String owner, String
	↪  address) {
		this.name = name;
		this.cuisine = cuisine;
		this.owner = owner;
		this.address = address;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getCuisine() {
		return cuisine;
	}

	public void setCuisine(String cuisine) {
		this.cuisine = cuisine;
	}

	public String getOwner() {
```

```
42                  return owner;
43          }
44
45          public void setOwner(String owner) {
46                  this.owner = owner;
47          }
48
49          public String getAddress() {
50                  return address;
51          }
52
53          public void setAddress(String address) {
54                  this.address = address;
55          }
56
57          public Menu getMenu() {
58                  return menu;
59          }
60
61          public void setMenu(Menu menu) {
62                  this.menu = menu;
63          }
64
65          public boolean equals (Object o) {
66                  Restaurant r;
67                  if (o == null || !(o instanceof Restaurant))
68                          return false;
69                  r = (Restaurant) o;
70
71                  return this.name.equals(r.name) &&
                    ↪  this.address.equals(r.address);
72          }
73
74          public int hashCode () {
75                  return name.hashCode() ^ address.hashCode();
76          }
77
78          public String toString() {
79                  return name + " (" + cuisine + ") owned by " + owner + " located
                    ↪  at " + address + " serves: \n\n" + menu.toString();
80          }
81
82  }
```

6.2 FastFoodRestaurant and Cafeteria

```java
/**
 *
 */
package solutions.finalexam;

/**
 * @author lord_pretzel
 *
 */
public class FastFoodRestaunt extends Restaurant {

	private String franchise;

	/**
	 * @param name
	 * @param cuisine
	 * @param owner
	 * @param address
	 */
	public FastFoodRestaunt(String name, String cuisine, String owner,
			String address) {
		super(name, cuisine, owner, address);
	}

	public FastFoodRestaunt(String name, String cuisine, String owner,
			String address, String franchise) {
		super(name, cuisine, owner, address);
		this.setFranchise(franchise);
	}

	public String getFranchise() {
		return franchise;
	}

	public void setFranchise(String franchise) {
		this.franchise = franchise;
	}

	public String toString() {
		return super.toString() + " from franchise " + franchise;
	}
}
```

```java
/**
```

```java
 *
 */
package solutions.finalexam;

/**
 * @author lord_pretzel
 *
 */
public class Cafeteria extends Restaurant {

	private String sponsor;

	/**
	 * @param name
	 * @param cuisine
	 * @param owner
	 * @param address
	 */
	public Cafeteria(String name, String cuisine, String owner,
			String address) {
		super(name, cuisine, owner, address);
	}

	public Cafeteria(String name, String cuisine, String owner,
			String address, String sponsor) {
		super(name, cuisine, owner, address);
		this.sponsor = sponsor;
	}

	public String getSponsor() {
		return sponsor;
	}

	public void setSponsor(String sponsor) {
		this.sponsor = sponsor;
	}

	public String toString() {
		return super.toString() + " sponsored by " + sponsor;
	}

}
```

6.3 Menu

```java
/**
 *
 */
package solutions.finalexam;

import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.ArrayList;
import java.util.List;

import javax.swing.text.NumberFormatter;

/**
 * @author lord_pretzel
 *
 */
public class Menu {

	private List<MenuItem> items;

	public enum Category {
		Appetizer,
		Dessert,
		MainCourse,
		Beverage
	}

	public static class MenuItem {

		private static NumberFormat formatter =
		↪  NumberFormat.getCurrencyInstance();

		private String name;
		private double price;
		private Category cat;

		/**
		 * @param name
		 * @param price
		 * @param cat
		 * @throws Exception
		 */
```

```
42                  public MenuItem(String name, double price, Category cat) throws
                    ↪  Exception {
43                          super();
44                          this.name = name;
45                          setPrice(price);
46                          this.cat = cat;
47                  }
48                  public String getName() {
49                          return name;
50                  }
51                  public void setName(String name) {
52                          this.name = name;
53                  }
54                  public double getPrice() {
55                          return price;
56                  }
57                  public void setPrice(double price) throws Exception {
58                          if (price <= 0.0)
59                                  throw new Exception("price of items has to be
                                    ↪  positive");
60                          this.price = price;
61                  }
62                  public Category getCat() {
63                          return cat;
64                  }
65                  public void setCat(Category cat) {
66                          this.cat = cat;
67                  }
68
69                  public String toString() {
70                          return cat.toString() + ": " + name + "\t" +
                            ↪  formatter.format(price);
71                  }
72          }
73
74          public Menu() {
75                  items = new ArrayList<> ();
76          }
77
78          public void addItem (MenuItem i) {
79                  items.add(i);
80          }
81
82          public List<MenuItem> getItems () {
83                  return items;
```

```
84          }
85
86          public String toString() {
87                  StringBuilder b = new StringBuilder();
88
89                  for(MenuItem i: items) {
90                          b.append(i.toString() + "\n");
91                  }
92
93                  return b.toString();
94          }
95
96  }
```

6.4 RestaurantGuide

```
1  /**
2   *
3   */
4  package solutions.finalexam;
5
6  import java.util.ArrayList;
7  import java.util.HashMap;
8  import java.util.HashSet;
9  import java.util.List;
10 import java.util.Set;
11
12 import solutions.finalexam.Menu.MenuItem;
13
14 /**
15  * @author lord_pretzel
16  *
17  */
18 public class RestaurantGuide {
19
20         private Set<Restaurant> restaurants;
21         private HashMap<Restaurant, Float> rating;
22
23         public RestaurantGuide() {
24                 restaurants = new HashSet<> ();
25                 rating = new HashMap<>();
26         }
27
28         public void addRestaurant(Restaurant r) {
29                 restaurants.add(r);
30         }
31
32         public List<Restaurant> getAllRestaurants() {
33                 return new ArrayList<> (restaurants);
34         }
35
36         public List<Restaurant> search(String item) {
37                 List<Restaurant> res = new ArrayList<> ();
38                 for(Restaurant r: restaurants) {
39                         Menu m = r.getMenu();
40
41                         for(MenuItem i: m.getItems()) {
42                                 if (i.getName().equals(item))
43                                         res.add(r);
```

```
44                          }
45                  }
46                  return res;
47          }
48
49          public void addRating(Restaurant r, float rat) {
50                  rating.put(r, rat);
51          }
52
53          public float getRating(Restaurant r) {
54                  return rating.get(r);
55          }
56
57          public Restaurant getHighestRatedOne () {
58                  Restaurant best;
59
60                  if (restaurants.size() == 0)
61                          return null;
62                  best = restaurants.iterator().next();
63
64                  for(Restaurant r: restaurants) {
65                          if (rating.get(r) > rating.get(best))
66                                  best = r;
67                  }
68
69                  return best;
70          }
71
72  }
```

6.5 RestaurantTest

```
1  package solutions.finalexam;
2
3  import solutions.finalexam.Menu.Category;
4  import solutions.finalexam.Menu.MenuItem;
5
6  /**
7   * @author lord_pretzel
8   *
9   */
10 public class RestaurantTest {
11
12         public static void main(String[] args) throws Exception {
13                 RestaurantGuide g = new RestaurantGuide();
14                 Restaurant r1, r2;
15                 Menu m;
16
17                 r1 = new FastFoodRestaunt("Mac", "junkfood", "Peter", "10 W 33
                   ↪  st, Chicago, IL", "MacDonalds");
18                 m = new Menu();
19                 r1.setMenu(m);
20                 m.addItem(new MenuItem("Fries", 2.50, Category.MainCourse));
21                 m.addItem(new MenuItem("Burger", 3.99, Category.MainCourse));
22
23
24                 r2 = new Cafeteria("MTCC commons", "junkfood", "Alice", "3555 S
                   ↪  State, Chicago, IL", "IIT");
25                 m = new Menu();
26                 r2.setMenu(m);
27                 m.addItem(new MenuItem("Fries", 3.50, Category.MainCourse));
28                 m.addItem(new MenuItem("Burrito", 5.99, Category.MainCourse));
29
30                 g.addRestaurant(r1);
31                 g.addRestaurant(r2);
32
33                 System.out.println("Restaurants serving burritos:\n\n" +
                   ↪  g.search("Burrito").toString());
34
35                 g.addRating(r1, 3);
36                 g.addRating(r2, 4);
37
38                 System.out.println("\n\nHighest rated one is:\n\n" +
                   ↪  g.getHighestRatedOne());
39         }
```

```
40  }
```