BZU-HUB

صُمِّمَ هَذا الموقع ليخدم طلبة جامعة بيرزيت، وهو موقع غير ربحي

```java
package myPackage;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
import java.util.InputMismatchException;
import java.util.Scanner;

public class Hotel {

    private String hotelName;
    private String hotelAddress;
    private static int MAX_ROOM_NUMBER = 20;
    private Room[] rooms = new Room[MAX_ROOM_NUMBER];
    private Scanner scan = new Scanner(System.in);
    private String path;

    public Hotel() {

        for (int i = 0; i < rooms.length; i++) {
            rooms[i] = new Room(i);
        }
    }

    public Hotel(String name, String adress, int maxRoomNumber, String path) {

        this.hotelName = name;
        this.hotelAddress = adress;
        Hotel.MAX_ROOM_NUMBER = maxRoomNumber;
        rooms = new Room[MAX_ROOM_NUMBER];
        for (int i = 0; i < rooms.length; i++) {
            rooms[i] = new Room(i);
        }
        this.path = path;
        readDataFromFile();
    }

```

```
37
38      // Here, data are being read from the file and stored in the the rooms' array.
39      private void readDataFromFile() {
40          File file = new File(path);
41          try {
42              Scanner sc = new Scanner(file);
43              while (sc.hasNext()) {
44                  String[] u = sc.nextLine().split(",");
45                  int q = Integer.parseInt(u[1]);
46                  rooms[q].reserve();
47                  rooms[q].setCustomerName(u[0]);
48                  rooms[q].setRoomType(u[2]);
49              }
50              sc.close();
51          } catch (FileNotFoundException e) {
52              System.out.println(e);
53          } catch (ArrayIndexOutOfBoundsException y) {
54              System.out.println(y);
55          } catch (NumberFormatException y) {
56              System.out.println(y);
57          } catch (Exception j) {
58              System.out.println(j);
59          }
60      }
61
```

```java
    // In this method, the existing file will be deleted and then recreated -will
    // have the same path-and be refilled with new data.
    public void updateCustomersFile() {
        try {
            File file = new File(path);
            PrintWriter out = new PrintWriter(path);
            for (int i = 0; i < rooms.length; i++) {
                if (!rooms[i].isRoomAvailable()) {
                    out.println(
                            rooms[i].getCustomerName() + "," + rooms[i].getRoomNumber() + "," + rooms[i].getRoomType());
                }
            }
            out.close();
        } catch (FileNotFoundException e) {
            System.out.println(e);
        }
    }

    // Reserve a room and choose its type.
    public void reserveRoom() {
        try {
            System.out.println("Please enter customer's name:");
            String name = scan.next();
            listAvaliableRooms();
            int n = askForRoomNumber();
            if (rooms[n].isRoomAvailable()) {
                rooms[n].reserve();
                rooms[n].setCustomerName(name);
                System.out.println("Thank you!, The room number: " + n + " has been reserved for you");
                System.out.println();
                System.out.println("Now, lets select the room type");
                selectRoomType(n);
            } else {
                System.out.println("Sorry, the room number: " + n + " is already reserved");
            }
        } catch (InputMismatchException y) {
            System.out.println(y);
        }
    }
```

```
101
102     // This private method is used in the previous one, selects the room type for
103     // the chosen one.
104     private void selectRoomType(int roomNumber) {
105         while (true) {
106             System.out.println("Please select one of the following room types: \r\n" + "Single \r\n" + "Double \r\n"
107                     + "King \r\n" + "Deluxe\n");
108             String type = scan.next();
109             if (checkRoomType(type)) {
110                 rooms[roomNumber].setRoomType(type);
111                 System.out.println("Thank you, the room type: " + type + " has been selected for you");
112                 break;
113             } else {
114                 System.out.println("Sorry, there is no room type: " + type);
115             }
116         }
117     }
118
```

```
118
119     // This is to change a previous reserved room.
120     public void changeRoomType() {
121         try {
122             int n = askForRoomNumber();
123             if (rooms[n].isRoomAvailable()) {
124                 System.out.println("Sorry, the room number: " + n + " is not booked yet");
125             } else {
126                 while (true) {
127                     System.out.println("Please select one of the following room types: \r\n" + "Single \r\n"
128                             + "Double \r\n" + "King \r\n" + "Deluxe");
129                     String type = scan.next();
130                     if (checkRoomType(type)) {
131                         if (rooms[n].getRoomType().equals(type)) {
132                             System.out.println("This room type is already selected.");
133                         } else {
134                             System.out.println("Thank you, your room type has been changed from: "
135                                     + rooms[n].getRoomType() + " to: " + type);
136                             rooms[n].setRoomType(type);
137                         }
138                         break;
139                     } else {
140                         System.out.println("Sorry, there is no room type: " + type);
141                     }
142                 }
143             }
144         } catch (InputMismatchException y) {
145             System.out.println(y);
146         }
147     }
148
```

```
148
149     // This is to delete a reserved room, the room then will be available for a new
150     // reservation.
151     public void deleteReservedRoom() {
152         try {
153             int n = askForRoomNumber();
154             if (rooms[n].isRoomAvailable()) {
155                 System.out.println("Sorry, the room number: " + n + " is not booked yet.");
156             } else {
157                 rooms[n].delete();
158                 rooms[n].setRoomType(null);
159                 rooms[n].setCustomerName(null);
160                 System.out.println("Thank you, the room number: " + n + " has been deleted.");
161             }
162         } catch (InputMismatchException y) {
163             System.out.println(y);
164         }
165     }
166
167     // This is to delete all reserved rooms and make them all available for new
168     // reservations.
169     public void deleteAllReservedRooms() {
170         boolean exist = false;
171         for (int i = 0; i < rooms.length; i++) {
172             if (!rooms[i].isRoomAvailable()) {
173                 rooms[i].delete();
174                 rooms[i].setRoomType(null);
175                 rooms[i].setCustomerName(null);
176                 exist = true;
177             }
178         }
179         if (exist) {
180             System.out.println("Thank you, all reservred rooms have been deleted.");
181         } else {
182             System.out.println("there are no reserved rooms to be deleted.");
183         }
184     }
185
```

```
185
186     // This method shows all reserved room information: room number, room type and
187     // customer name.
188     public void showAllReservedRoomsInformation() {
189         boolean exist = false;
190         System.out.println("Room Number \tRoom Type \tCustomer Name");
191         for (int i = 0; i < rooms.length; i++) {
192             if (!rooms[i].isRoomAvailable()) {
193                 System.out.println(rooms[i].getRoomNumber() + " \t\t" + rooms[i].getRoomType() + " \t\t"
194                         + rooms[i].getCustomerName());
195                 exist = true;
196             }
197         }
198         if (!exist) {
199             System.out.println("There are no rooms to show.");
200         }
201     }
202
203     // This private method checks whether the room type a customer chooses is
204     // valid or not, and its used in selectRoomType and changeRoomType methods.
205     private boolean checkRoomType(String roomType) {
206         if (roomType.equals("Single") || roomType.equals("Double") || roomType.equals("King")
207                 || roomType.equals("Deluxe")) {
208             return true;
209         } else {
210             return false;
211         }
212     }
213
```

```
214     // Here, the user is asked to enter a valid room number and the program checks
215     // whether it is valid or not(room number should be between 0 and maximum room
216     // number).
217     private int askForRoomNumber() throws InputMismatchException {
218         try {
219             while (true) {
220                 System.out.println("please enter a valid room number between 0 and " + (rooms.length - 1));
221                 int n = scan.nextInt();
222                 if (n >= 0 && n < rooms.length) {
223                     return n;
224                 } else {
225                     System.out.println("Room number is not valid");
226                 }
227             }
228         } catch (InputMismatchException j) {
229             scan.next();
230             throw j;
231         }
232     }
233 
234     // This methods lists all available rooms for the user(i.e., rooms that are not
235     // reserved yet).
236     private void listAvaliableRooms() {
237         System.out.println("Avaialble Rooms:");
238         for (int i = 0; i < rooms.length; i++) {
239             if (rooms[i].isRoomAvailable()) {
240                 System.out.print(rooms[i].getRoomNumber() + "\t");
241                 if ((i + 1) % 10 == 0) {
242                     System.out.println();
243                 }
244             }
245         }
246         System.out.println();
247     }
248 
```

```
232     }
233
234     // This methods lists all available rooms for the user(i.e., rooms that are not
235     // reserved yet).
236     private void listAvaliableRooms() {
237         System.out.println("Avaialble Rooms:");
238         for (int i = 0; i < rooms.length; i++) {
239             if (rooms[i].isRoomAvailable()) {
240                 System.out.print(rooms[i].getRoomNumber() + "\t");
241                 if ((i + 1) % 10 == 0) {
242                     System.out.println();
243                 }
244             }
245         }
246         System.out.println();
247     }
248
249     // This method is to get the name of the hotel.
250     public String getName() {
251         return this.hotelName;
252     }
253
254     public void setName(String name) {
255         this.hotelName = name;
256     }
257
258     // This methods is to get the hotel address.
259     public String getAddress() {
260         return this.hotelAddress;
261     }
262
263     // This method takes the hotel address from the user and stores it.
264     public void setAddress(String address) {
265         this.hotelAddress = address;
266     }
267 }
```