Middle East Technical University
Northern Cyprus Campus

CNG 443: Intr. to Object-Oriented Programming Languages and Systems
Assignment 2: BASIC (*B*ed *A*nd breakfa*S*t *I*n *C*yprus)

Date handed-out: 10 November 2023, Friday
Date submission due: 24 November 2023, Friday 23:55 (Cyprus time)

Learning Outcomes

On successful completion of this assignment, a student will:

- Have used an UML class diagram to implement an application.
- Have practiced class hierarchy and the relevant design and implementation decisions.
- Have learnt how to maintain different types of objects.
- Have practiced and used abstract classes, and interfaces.
- Have learnt how to create a package for an application.
- Have also practiced Exception handling in Java.

Requirements

This assignment is about creating a small Java application for a Bed&Breakfast booking and management system which is called BASIC. This application will maintain users and also properties, and their hosts, and will manage bookings done on properties by users. The figure below shows a summary class diagram for this application.

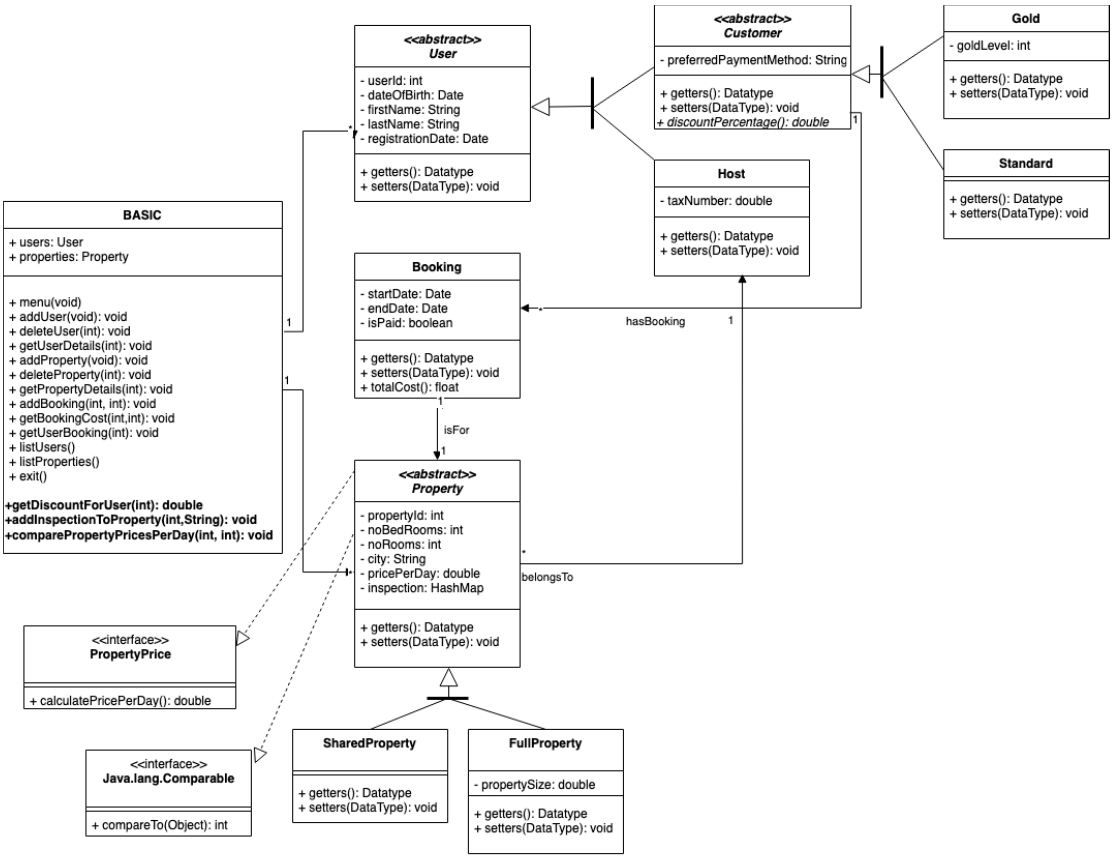

Figure 1 BASIC – Updated Class Diagram

The overall requirements are based on this class diagram, which is also summarised below:

- The main application called BASIC will be used to maintain information about users and the properties maintained by this application. BASIC will also have the main method and will provide the overall interaction with the application. The UML diagram given above includes arrows which shows in which way the classes interact. For example, the BASIC application which will have the main will maintain users and properties but not bookings. For instance, Booking is associated to a user and a property. Therefore, BASIC class should include the static main method where an instance of this class is constructed and the menu of commands is displayed to the user. Since we have not yet covered Graphical User Interfaces (GUI) in this course, you need to implement it as a command-line application. The required methods are as follows:
    - ***void menu()***: This method will display the interaction menu to the user.
    - ***void addUser()***: This method will add new user to the list of users maintained. Each user needs to have unique ID number. You need to also record if it is a host, standard customer, or gold customer. **Gold customers have a level which will allow them to have special discounts. The level can be between 1-3.**
    - ***void deleteUser(int userId)***: This method will read an ID number of a user, and delete the corresponding user object. If the user ID number does not exist, the program should provide an appropriate error message.
    - ***void getUserDetails(int userId)***: Given a user ID number, this method will display the user details. If the ID number does not exist, the program should provide an appropriate error message. **Otherwise, if the userID exists, then you will display all the relevant details of the user depending on the user type. Therefore, you are strongly recommended to implement toString() method and override it. There is no need to show the booking details of the user.**
    - ***void addProperty()***: This method will add a new property. **The system can also have two types of properties now: Shared room or a full house. If it is a full house then the system will also record the size of the house in terms of square meters. This is for tax purposes. For property, you need to also add the host details. You need to get the host's userID and check if the host already exists or not. If it exists then you should add the relevant reference to the property object.**
    - ***void deleteProperty(int propertyId)***: Given a property ID, this method will delete the given property. If the propertyID does not exist, the program should provide an appropriate error message.
    - ***void getPropertyDetails(int propertyID)***: Given a propertyID, this method will display the property details. If the propertyID does not exist, the method should provide an appropriate error message. **Otherwise, if the propertyID exists, then you will display all the relevant details of the property depending on the property type. Therefore, you are strongly recommended to implement toString() method and override it.**

- ***void addBooking(int userId, int propertyId)***: This method will record a booking for a given user and property. If the property with the given ID or the user with the given ID does not exist then the relevant error messages should be given. Please note that when a booking is added, all the relevant information should be asked to be recorded including the start date and endDate.
- ***Void getUserBooking(int userId)***: This method will display the bookings done by a user. It will mainly display the booking start and end dates, if there are no bookings, then it should give the relevant feedback to the user.
- ***void getBookingCost(int userId, int propertyId)***: This method will call to totalCost method on the Booking object. The method should first identify the booking object for a given user for a given property ID. If there are multiple bookings on a property by the same user, then you will need to display the total cost of each of those bookings. totalCost method on Booking will calculate the total number of days of the booking and **will multiply it with the price returned by calculatePricePerDay()** of the property. For example, if the booking is for 1/06/2023 to 11/06/2023 then this booking is for 10 days. If the per day cost of the property is 50 then it will show 500.

  Please note that now that we have different types of property instead of using the pricePerDay field, the calculatePricePerDay() method should be used. This is because the pricePerDay shows the full house price per day but depending on the type of the property it needs to be computed further for booking:

  - **SharedProperty: If it is a shared property then the calculatePricePerDay() will divide the pricePerDay field by the number of bedrooms (noBedRooms field) of the property. For example, if the pricePerDay is 100 USD and it has 4 rooms. Each room will cost 25 USD.**
  - **FullProperty: If it is a full property then depending on the size of the house there would be additional tax fee which is computed as follows:**
    - **Up to 200 m^2, it will be additional of 1% tax.**
    - **Between 200-300 (inclusive), it will be additional of 3% tax.**
    - **Above 300 m^2, it will be additional 4% tax.**

    Therefore, if the property is 200 m^2 and pricePerDay is 100 USD then calculatePricePerDay() will return 103 USD.

    This method will also make use of the getDiscountForUser() method explained below to also check if the user also has a special discount. For example, if the user is a gold customer with level 1 then they will get 1% discount which means the property costing 103 USD will now cost ~101.97 USD.

  - ***void listUsers()***: This method will list all the users. All user details should be displayed.
  - ***void listProperties()***: This method will list all the properties. All property details should be displayed.
  - ***void exit()***: This method should terminate the program.
  - ***getDiscountForUser(int userId):*** **This method will return the discount percentage for a customer. Depending on the type of the customer discountPercentage is calculated differently as follows:**
    - **Gold Customer: If the customer is a gold customer then depending on the level of the customer it can be calculated differently. If it is a level 1, then the discount is 1%, if it is level 2 then the discount is 2% and if it is level 3 then the discount is 3%.**
    - **Standard Customer: If the customer is registered for more than 10 years (including 10) then they get 2% discount. You need to take the current system date and use the registrationDate field to calculate the number of years. If it is less than 10 they get no discount.**
  - ***addInspectionToProperty(int propertyId, String)*****: Each property will have regular inspections and their reports as a simple String will be recorded. To record the inspections you will be using a Hashmap. When an inspection is added then you will take the current system date and use it as hash key (There can only be one inspection per day), and you will record the inspection text (i.e., String). That means inspections will be as follows:**

    - **1 November 2023 – “It was in good condition”;**
    - **10 November 2023 – “Broken toilet that needs to be fixed”.**
  - ***comparePropertyPricesPerDay(int propertyId, int propertyId):*** **This method will take two property IDs and show which one is cheaper to rent. It should use compareTo() method on the property class which compares properties with calculatePricePerDay() method on each property. For example, if the user enters property ID 4 and 5, then it should show user a message such as “Property ID 4 is cheaper” or “Property ID 5 is cheaper” or “they have the same price!”.**
- The given class diagram has all the fields and methods needed, so please follow the diagram. If you need extra fields, you can but please make sure that you update your class diagram.
- In this assignment, you need to do exception handling. Please make sure that all the **checked exception types are handled** in your methods.
- Since you did not learn how to make your class persistent or use a database, you will lose data every time you run your application. Therefore, you need to create some objects before you start your application. Your application needs to start with 3 user objects, 3 property objects, with each user having one booking and each property having one host. To create this data, you need to create a class

which is called *PopulateData* that can be used to populate your application with these initial data. Please note that this is not given in the UML diagram but it is an extra class that needs to be created.

- Once you complete your implementation, fully update the UML class diagram, in case any changes are needed and submit it as well. Original UML diagram was created with Draw.io. You can use that or any other tool to create your updated UML diagram (e.g. Draw.io (www.draw.io), LucidChart (www.lucidchart.com/), Visio, etc.). This assignment also has an attachment that is the Visio version of this diagram so that you can import it to a tool and edit it.

Environment: As a development environment, you can use any IDE you like but you are strongly recommended to use **IntelliJ** (https://www.jetbrains.com/idea/).

Submission: Please organise your submission as a *single ZIP file* that includes the following:

- **[Jar file]**: A JAR file that can be executed on a command line. Mark sure that BASIC is the main class.
- **[doc folder]:** This should include the full Javadocs generated.
- **[source folder]:** This should include your full source code. Please note that if you do not include *.java files, we cannot grade your work and you will automatically receive zero.
- [**diagram folder**]: This should include the updated UML.

If you are not following this submission structure, you will not receive marks from the Package item in the grading policy.

Extra Requirements:

Some additional requirements are listed below:

- We have not yet covered how to use a Database or make objects persistent in this course. Therefore, this assignment maintains objects such as property and users in arrayLists.
- We have not yet covered Graphical User Interfaces (GUI) in this course. So please provide a command-line interaction (CLI).
- For each class, please decide what kind of constructors are required, the access types of methods and fields. If you use private fields, make sure that you provide accessor and mutators. For each class, you need to do constructor overloading and provide at least **two constructors**. Please note that the default constructor does not count so your code should include default plus two different versions.
- You should use the Date class provided in java.utils in order to read the date from the user, read a string with the “dd/mm/yyyy” format, in which dd, mm and yyyy represent the day, month and the year, respectively. Study the “Parsing Strings into Dates” section provided in:
    - https://www.tutorialspoint.com/java/java_date_time.htm
- **Regarding Date representation, you can use LocalDate and Calendar classes**.

- Pay attention to the overall design, layout and presentation of your code.
- You need to submit your Java code with proper JavaDoc comments. For each class, you need to have used at least @author and @version, and for each method, you need at least @param and @return.

Assessment Criteria

This assignment will be marked as follows:

Aspect	**Marks (Total 100)**
All classes are implemented	10
All class hierarchies are implemented	10
All interfaces are implemented and used	10
For all classes constructors are properly implemented	10
For all classes all required data fields are implemented	10
For all classes all required methods are implemented	10
All methods in the BASIC are implemented	30
Package Structure and Jar for Invoking the application	5
Exception Handling is done	5

For each of the items above, we will also use the following grading criteria:

Half working	%20
Fully working	%20
Appropriate reuse of other code	%10
Good Javadoc comments	%10
Good quality code[1]	%40

[1] 15 principles will be considered during grading: https://www.informit.com/articles/article.aspx?p=2223710