Advanced Programming
Assignment 2
(30 Marks)

You must demonstrate*/explain* your work to the tutor, if you are absent/unavailable or fail to demonstrate properly, zero marks will be awarded.

Please note, this is an individual assignment, and it will be checked for plagiarism. All the involved parties will be penalized if any plagiarism is found.

Please visit https://goo.gl/hQ87zg for more details.

*Demonstration/explain – can include

- explaining the code used in your submission.
- rewriting whole or part of the code
- changing part of the code

Instructions

1. This assignment is based on a business scenario, with 4 stages of solutions.
   a. Stage – A – Maximum 59%
   b. Stage – B – Maximum 69%
   c. Stage – C – Maximum 79%
   d. Stage – D – Maximum 100%
2. **Each stage elaborates on the previous stage, so you need to submit only one stage.**
3. Submit one MS Word document and the java source code/s (or Eclipse project). Use the following format to prepare the word document (use the report template available in assignment page).
   a. Java program - copy and paste your Java program (not the screenshot of the code)
   b. Enough screenshots of the output that show all possible outcome.
4. Marks will be given for proper indentation and comments.
5. **Assignment Demonstration is mandatory**.

Other requirements:

- This assignment must be written in Java.
- Your code must have appropriate header (multiline/block) comments including your name and student number, the name of the .java file, the purpose of the program, brief explanations of variables and explanations of any code, which is not obvious to another programmer, summarising the input, output and local variables as well as expressions used in your program and test data.
- Include inline (single line) comments throughout the program describing important statements.
- Use appropriate and descriptive variable following the naming rules and conventions.
- Marks will be allocated depending on the amount of original work submitted. Marks will be deducted for

plagiarised and/or unattributed work.

Business Scenario:

Your task is to produce, using standard Java, a prototype resort management system for ‘Resort by the Sea (RBS)’, a (fictitious) resort located in Cairns, Queensland.

RBS is a medium-sized luxury seaside resort. The resort consists of 30 accommodations, out of which, 10 can accommodate two guests, another 10 can accommodate four guests and the remaining 10 have space for six guests.

The management of RBS is interested in modernizing the entire resort management system. The system includes all aspects of the business; namely, reservations, check-in, check-out, food and beverage services and airport pick up and drop.

The first phase is to develop a prototype system for room booking, check-in and check-outs. Notes from interviews with RBS staff and instructions from your supervisor are given below. You should analyse the interviews and instructions to determine the functional requirements for your program. There are four stages in this assignment (A, B, C and D). Since each stage elaborates on the previous one, you only need to submit one program – the one implementing the highest stage you were able to complete.

Stage A – Modelling and implementing a Reservation System

An edited summary of client interviews and additional requirements from your supervisor is given below: Kanchan – RBS Manager

“At Resort By The Sea we have three types of accommodations, 10 numbers of single room that can fit two guest, then 10 units with two rooms which can fit four guest and our larger 10 units have three rooms which can fit six guests.

All the rooms have a unique room number (incrementing from a starting number of 101 for the first room), a room name, a single line description and a cost per day. For this stage, I’d like the system to be able to store up to six rooms, two room from each type. So, we can easily test the system. My guest officer will need to be able to process a guest booking, check-in, and check-outs.

For this prototype we won’t computerise guest records but be aware that each guest is identified with a (string) ID code. and the computerization will not accept advance room reservations. We computerize booking for the same day check-in. Both of us (Manager & Guest Officer) need to be able to display all information stored about a specified accommodation (including if it has been occupied), and also to list a summary of room ID’s, and availability status of all rooms together with a final count of the number of checked-in rooms and the total income from all the checked-in rooms. All displays should be neatly formatted."

Nathan – RBS Guest Officer “When a guest wishes to book an accommodation, I collect the guest’s ID, and find out which type (single, double, or triple) of room, and for how many days they wish to book. If the required room is available, I then record that the room is booked by that guest for that many days. I then charge the guest and give them a receipt listing their guest ID, the room number, the number of days, and the total charge. As the system is being computerised, I’d like the system to let me know if reservation isn’t possible because the required rooms are already booked out and the system to show what other rooms are available with larger than the required capacity. When a guest checks-out, I look up the record for that room number and mark the room as being available for booking.”

Your supervisor advises that your program should be able to use object-oriented techniques to model a room in the class Room. The Room class shouldn't collect data from the user. Another class, StageA, should be a console-driven class that houses the main method, obtains relevant room details from the manager as required, creates and manages objects of class Room in order to implement the functionality from the above interviews. The Room class must contain a static variable indicating the value of the next id and use this when determining the id for a new room object. The Room class constructor should accept user data from StageA and set the room attributes accordingly.

In stage A, your program MUST utilise a single array (called Accommodation) for storing information about all the rooms, in class StageA. Collections such as ArrayLists must not be used. Not meeting this requirement will result in a mark of 0.

Your Room class must include and use a Boolean method with signature checkinRoom(String guestID, int numDays) which sets room instance variables to indicate the room is being checked-in by guestID and booked for numDays of days. It returns false if the room is already booked out, true otherwise.

Similarly, Room must implement a Boolean method called checkoutRoom() to checkout a room. This method returns false if the room wasn't chedcked-in and true if it had been checked-in. Your StageA should be designed to use the return values of the Room checkinRoom and checkoutRoom methods for checking in/checking out rooms and determining if an attempt is being made to book an already checked-in room or check-out a room that wasn't checked-in.

The StageA class should provide a user-friendly menu to allow the user to perform tasks relevant to the needs expressed in interviews, above, calling methods in Room objects as relevant. Your program should be able to handle conditions such as searching for rooms that don't exist, attempting to add more rooms than the (user specified) maximum number of rooms, a guest attempting to check-in a room that another guest has already checked-in etc.

There are fixed costs for each of the rooms:

Room Type	**Price**
Single Room	$230
Double Room	$350
Triple Room	$500

You are expected to adhere to all relevant object-oriented programming guidelines, including:

- Visibility of instance variables and methods set appropriately.
- No unnecessary accessors (setters) or mutators (getters) - only provide methods which will be needed when implementing the application class in this stage.
- Where object instance variable values can be reasonably determined before object creation, initialisation of these variables should be carried out via a constructor.
- Methods should work with instance variables when performing their required task.
- Parameter lists in methods should be appropriate to the task the method is performing - only accept parameters where a method requires one or more values from the caller to perform its assigned task that it does not already have access to.
- Methods which need to communicate a value or result back to the caller should do by returning the value in question, not by storing it in an instance variable or printing it to the screen.
- Data classes (i.e. Room) should not prompt for input from the user

Stage B - Extending the Room class.

In this stage you rename class StageA as StageB, modify and extend the Room class and modify the StageB class to meet the requirements presented in the interviews and supervisor requirements below.

Kanchan – RBS Manager "We have several categories of rooms, and don't have 'generic' rooms. We have standard, deluxe, and premium categories of rooms. When adding a room, I need the system to ask which category of room is being added and collect details relevant to that room. It shouldn't be possible to add a 'generic' room as in stage A.

These categories are – Standard – which has basic room amenities, Deluxe – rooms are more spacious and has bathtub, and Premium – rooms contain spa and kitchenette.

There are fixed costs for each of the rooms based on the category:

Category	Single Room	Double Room	Triple Room
Standard	$230	$350	$500
Deluxe	$280	$430	$600
Premium	$350	$500	$690

This cost should be stored when adding a room into the system. The display of a specified room should produce neatly formatted output of data relevant for that type of room."

Important: Your supervisor advises that Room class should be extended to categories Standard, Deluxe and Premium rooms to implement the requirements described in the above interviews. Your Stage B program must utilise the same array (Accommodation) used in Stage A for storing information about all the rooms, in class StageB.

Your supervisor also specifies that the Room class should be modified so only subclasses of Room can be instantiated and that those subclasses cannot be further extended. In addition, the Room class must be modified so as to force all subclasses to implement a method determinePrice() which returns a double (the cost) relevant for the subclass, calculated as per Kanchan's interview above. You should assume when adding a new room to the system, the manager will enter, as a string, a valid category and this information should be stored by the Room object and used by it to determine the room price. You are expected to adhere to all relevant object-oriented programming guidelines as described in stage A.

Stage C – Adding Special Services & Exception Handling

Kanchan – RBS Manager added "our resort has two special rooms to support wheelchair-bound guests. These rooms are fitted with ramps, one at the entrance to the room and one at the rear which has direct access to the beach. These two rooms also include an emergency calling facility in the washroom/toilet. One standard double room and one standard triple room have this facility. We allocate these rooms when guests request or when we get wheelchair-bound guests".

Kanchan – further added, "its our Resort policy that we don't let these rooms to be occupied by a single guest, even if they're wheelchair-bound. For emergency and medical reasons, we welcome only accompanied by at least one more guest".

In this stage you rename class StageB as StageC, modify and extend the Room class, add a new Interface to implement Special Services and implement the Interface to two new extended classes of room. The two new extended classes; one standard double and one triple rooms, will include additional room properties (ramp length and width and number of emergency calling facilities) and methods based on the details given in the interview above. Also modify the StageC class to meet the requirements presented in the supervisor requirements below.

Your supervisor specifies that the room constructor must ensure the room category is one of those specified in Kanchan’s interview as you can no longer assume the user will enter a valid category. The room constructor must reject an invalid room category parameter and throw an IllegalArgumentException with a suitable message. This exception must be caught in StageC and the message displayed. Your supervisor has also asked you extend class Exception to define your own exception type, to be called AccommodationException. This AccommodationException class should be in its own separate source code (.java) file.

The Room checkinRoom(String guestID, int numDays) and checkoutRoom() methods must be rewritten to be 'void' and to generate and throw a AccommodationException containing a suitable error message if an attempt to check-in or check-out the room fails.

Note that any AccommodationException that is thrown should be caught in StageC and the message displayed.

Stage D – Adding Object Persistence and re-factoring bookings to an ArrayList

In this stage you rename class StageC as StageD, modify the Room, and StageD classes to meet the requirements presented in the interview and supervisor requirements below.

Kanchan – RBS Manager “I’d like the system to save all information before exiting, and then restore that information again when it is next run.”

Your supervisor instructs you to refactor array bookings to a single ArrayList of type Room called Accommodations. There is no longer a limit on the number of rooms that can be stored.

Your supervisor also instructs you to design an approach to writing out to file information for all rooms currently stored in the system in such a way that they can be reconstructed in full and then implement functionality to:
extract details for all rooms currently in the system and write the information out to a text file (in a format which will allow the same set of objects to be restored in full later) when the user chooses to exit the program, and read the information back in from the text file and use it to restore the existing set of objects in full and store them back in the Accommodation ArrayList when the program starts up.

The design should take advantage of inheritance. Your text file location should be relative to your project workspace – it should not be necessary to specify a full path and your solution will lose marks if it opens a file with a path that exists on your local machine, rather than one relative to the project workspace. If there is no data file then the program should proceed to run in an "empty" state, asking the user what the maximum number of items to store is, and then displaying a menu for providing the stage A/B/C functionality. You are not permitted to use automatic object serialization in your implementation.

Assignment submission information:

Submissions through **Canvas** must be made on or before the due date/time.
Each submission should have two files.
1. A report (name of the report should be with your student number, eg: 1012546_assignment1.docx) – use template provided with this assignment.

This report will be used for plagiarism check using Turnitin software. **20% of marks will be deducted if this report is missing for plagiarism check.** Report must (***.doc/docx)*** contain:

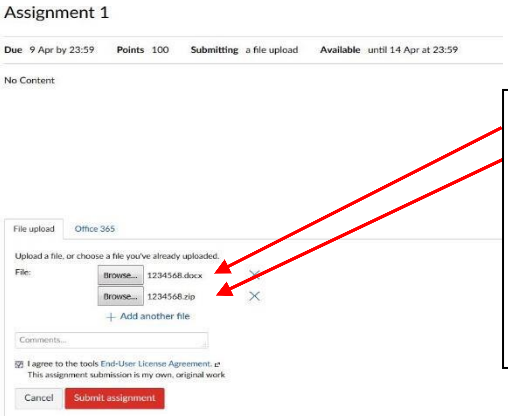
- Description of the problem
- A copy of the contents of the **.java** file for all tasks (copy and paste the code not

the screenshot of the code).

- Pasted **text output** or **screen shots** of the working program resulting from the testing of the program.

2. A ***.zip*** file (name of the zip file should be your student number, eg: 1012546_assignment_1.zip) containing:

a) The actual java project (Eclipse project file) with comments.



Assignment 1

Due 9 Apr by 23:59 **Points** 100 **Submitting** a file upload **Available** until 14 Apr at 23:59

No Content

Upload a file, or choose a file you've already uploaded.

File: 1234568.docx

1234568.zip

☑ I agree to the tools End-User License Agreement.
This assignment submission is my own, original work

Make sure two files are selected separately for submission as shown in this figure.
Give names:
studentid_Assignment_1.docx
studentid_Assignment_1.zip

Marking Criteria – Refer to rubric given in Assignment page.

End of Assignment