**Week 3 Grade: 82/100 (6.6/8 of the final grade)**

|  |  |
| --- | --- |
| **Points** | **Item** |
| 25/30 | Feedback from previous assignment is incorporated correctly into the class diagram.  Manager   * updateReservation: where is the method going to get the values to change the reservation? You are missing the parameter with Reservation object * complete\*, cancel\*, calculate\*: missing parameter to indicate which reservation to do the action on * missing method to retrieve Account object given account id * createAccount: incorrect method name - the account object already exists as it is being passed in so the method should be called addAccount   Account   * IDNUMBER - this should not be a constant and so it should not be all upper case. It will receive its value from the UI on creation of the instance so each instant will have a different value. A constant is a hardcoded static final value that is the same memory location for all instances and an attribute is a separate value for each instance. An attribute may only be allowed to get a single value and never change (final) or it may be allowed to change * Missing toString() method   Reservation   * noOfBeds - this attribute is storing the number of beds - per requirements there is no default value * missing attribute for number of bedrooms with default of 1 * caclulateBasePice: invalid method - there is no separate “ base price” - there is a calculation for reservation for the night and for the total reservation that requires code in the corresponding methods. And then children may or may not have additional feeds for the price per night. * Missing toString() methods (children will need their own methods too unless the code is identical as parent) |
| 5/5 | Class diagram meets the system specification and does not have anything extraneous (you should only be adding constructors and making some classes/methods abstract) |
| 18/20 | Diagram includes correct constructors for all classes. Every class must have at least one constructor with or without parameters as needed and is appropriate for that class.   * Reservation(): invalid constructor - cannot create an object without values * Missing constructors to take JSON or value with XML data to parse, validate, and assign attributes |
| 6/10 | Constructors include correct names, access modifiers, and parameters as needed; parameters have data types and variable names for each parameter   * Reservation(Integer accountID,...String status): accountID cannot be passed in since that value is not set until the reservation is added to the account object; status cannot be passed since it needs to be draft on creation by default; missing parameter of number of bedrooms * Reservation(Integer accountID,...startDate): same as above * Reservation children: constructors of children classes must have all the values that must be set on creation and not just their locally defined attributes |
| 8/10 | Classes and/or methods have been changed to abstract as appropriate  Reservation: calculate\*: per requirements there are values these method should calculate for a generic calculation so they cannot be abstract |
| 15/20 | There are Java files from previous assignment with added pseudo code for the constructors and methods  Address   * toString(): incorrect pseudo code - the purpose of toString() is not to print anything but rather to format the attributes values into some format and return it as String. That string then can be used to write the data to file or display to user or write to log, etc. * Address(String city,...): incomplete pseudo code - it needs to validate the parameters first and then assign to the attributes (similar comment across all constructors with parameters)   Manager   * Manager(): incomplete pseudo code - it must also load the accounts data along with its reservations from the hardcoded path * addReservation(Integer accountNumber, Reservation reservation): incorrect pseudo code - it needs to find the account object that matches the passed in id and then call its add method (separation of concerns principle); same comment for all other methods that do something with reservation (reservations are managed by account and not manager) * updateReservation(Integer accountNo): where is it getting these details from and which reservation is it updating? * createAccountDirectory(Integer accountNo): incorrect pseudo code - the method name indicates it will create dir so that is all it should be doing * createAccount(Account account): incorrect pseudo code - what UI is doing can be a comment above your method but is not part of your pseudo code (same comment for other methods) * updateAccount(Account account): manager cannot update values as that would break separation of concerns principle so it needs to call Account’s method to do this - so in order for manager to have update account method, you need to add such update method to Account class * completeReservation: per requirements this system does not automatically update any files but rather there is requirement UI can request for this update to happen and that is why you have saveToFile(Integer accountID) method (same comment for all methods that update files) * saveToFile: it needs to update account file and its reservation files   Account   * addReservation(Reservation reservation): unless your design is to create a clone to add, the reservation exists as it is being passed in so there is nothing to create; it should not be saving anything (same comment for any other methods saving except save method) * completeReservation(String reservationNo): method cannot change the reservation status but rather call its method to do the change (reservation object controls is status data) - same comment for other methods trying to change status directly instead of calling appropriate method * getReservation(String reservationNo): incomplete pseudo code - it needs to find the reservation object that matches the parameter’s id and return that object otherwise return null * updateReservation(Reservation reservation): similar to update account in manager - reservation needs to do its updates so to have this update you need update methods to call in Reservation and children * calculate\*: incorrect code - need to call reservation methods   Reservation   * cancelReservation(): need to do validation before cancelling and should not be changing files (similar comment for complete\*) * calculateBasePrice(): incorrect method - a parent class cannot determine something only child class knows - calculatePricePerNight should calculate and return what is correct for the price and similar for calculateTotalPrice   ApartmentReservation   * calculatePricePerNight(): should call parent’s method to get the basic price and not redo calculation and then should add its additional fees (same for other children) * calculateTotalPrice(): should call method for price per night and not recalculate again duplicating the code |
| 5/5 | Diagram is exported and submitted as an image (in ArgoUML go to File | ExportGraphics and it converts workspace to image file PNG by default) |