|  |
| --- |
| Problem Statement |
| Create a Class SavingAccount with field’s acc\_balance, acc\_ID, accountHoldername, isSalaryAccount. Also add setter and getter methods with business method like withdraw and deposit.   a.Create class BankAccountList which will maintain SavingAccount objects. Ensure that this class should not allow duplicates as well as data should be displayed in sorted order. (as per acc\_ID) |
| Create class Movie\_Details with field’s mov\_Name, lead\_actor, lead\_actories, and genre add setter and getter method in that class.     a. After creating this class create class Movie\_DetailsList class who will maintain all the objects.  b. Movie\_DetailsList class should have method add\_movie(), remove\_movie(), remove\_AllMovies(), find\_movie\_By\_mov\_Name(), find\_movie\_By\_Genre()  c. Movie\_DetailsList should have method which will take an argument that will be use to determine on which to sort |
| Create two classes Employee\_information and MMASaving\_Account   a. Employee\_information class will have fields empID, Employee\_name, employee\_designation , Employee\_salary, employee\_comm with that they will have setter getter methods    b. MMASaving Account class will have fields accountID, accountholder\_name, account\_balance, isSalaryAccount with that they will have setter and getter methods   c. Employee\_information object and MMASaving\_Account object belongs to a same entity assuming there are more than 5 entities how will you store the objects preserving the relation between them. |
| Create two classes Employee\_information and MMASaving\_Account   a. Employee\_information class will have fields empID, Employee\_name, employee\_designation , Employee\_salary, employee\_comm with that they will have setter getter methods    b. MMASaving Account class will have fields accountID, accountholder\_name, account\_balance, isSalaryAccount with that they will have setter and getter methods   c. Employee\_information object and MMASaving\_Account object belongs to a same entity assuming there are more than 5 entities how will you store the objects preserving the relation between them. |
| Create class Movie\_Details with field’s mov\_Name, lead\_actor, lead\_actress, and genre add setter and getter method in that class.  a. After creating this class create class Movie\_DetailsList class who will maintain all the objects. b. Movie\_DetailsList class should have method add\_movie(), remove\_movie(), remove\_AllMovies(), find\_movie\_By\_mov\_Name(), find\_movie\_By\_Genre() c. Movie\_DetailsList should have method which will take an argument that will be use to determine on which to sort |
| There is parking slot available in R-Mall with 3 floors; each floor has 4 sections and each section can maximum park 20 cars. You need to design one application which will maintain all car details in such way when ever car owner arrives to collect his car your application should provide details including where it is located. a. Create class Parked\_CarOwner\_Details which will have field’s owerName, carModel, carNO, owerMobileNo, owerAddress with setter and getter methods. b. Create class Parked\_CarOwenerList which will have method’s int add\_new\_car(Parked\_CarOwner\_Details obj), remove\_car(), get\_parked\_car\_location(token) |
| Create two classes Employee\_information and MMASaving\_Account  a. Employee\_information class will have fields empID, Employee\_name, employee\_designation , Employee\_salary, employee\_comm with that they will have setter getter methods  b. MMASaving Account class will have fields accountID, accountholder\_name, account\_balance, isSalaryAccount with that they will have setter and getter methods c. Employee\_information object and MMASaving\_Account object belongs to a same entity assuming there are more than 5 entities how will you store the objects preserving the relation between them. |
| Create classes for the below objects with the fields specified below. Create three different instances for each type and add them to a List. Now, read from these List and print each attribute in console.  Laptop : company, model, operatingSystem, processor Car : make, model, year, price Television : company, type (LCD, LED, Plasma), 3D enabled, price CellPhone : company, model, description, operatingSstem, price School : name, city, school district, greatSchoolRanking  Set: Observe that Set doesn’t accept duplicate objects. For each above object, override equals() and hashCode() methods and define the equality. Please see the criteria below. Create few equal objects (based on the below criteria) and add them to HashSet and observe the fact that Set doesn’t accept duplicates (through equals() and hashCode()).  Company and model together define the equality for Laptop. If two laptops have same company and model then they are equal. Car : make and model together define equality Television : company, type (LCD, LED, Plasma) and price together define equality CellPhone : company, model and operatingSstem together define equality School : name, city and school district together define equality  LinkedHashSet : Add all the names of our class students randomly to a LinkedHashSet, iterate through them and print. Observe that LinkedHashSet is maintaining the order that you add.  TreeSet : Add all the names of our class students randomly to a TreeSet, iterate through them and print. Observe that TreeSet keeps these elements in a natural order.  TreeSet : Create few instances of Car and add them to a TreeSet. Have Car implements Comparable interface and override compareTo() to order them by ascending order of make of the Car. Observe the fact that you need to provide the order criteria in compareTo().  Map : Create a simple HashMap to store all of our class students’ names and their favorite fruits. Try to get their favorite fruit by giving their name. Also, list all the names and their favorite fruits using keySet() and entrySet() approaches. |
| Create classes for the below objects with the fields specified below. Create three different instances for each type and add them to a List. Now, read from these List and print each attribute in console.  Laptop : company, model, operatingSystem, processor Car : make, model, year, price Television : company, type (LCD, LED, Plasma), 3D enabled, price CellPhone : company, model, description, operatingSstem, price School : name, city, school district, greatSchoolRanking  Set: Observe that Set doesn’t accept duplicate objects. For each above object, override equals() and hashCode() methods and define the equality. Please see the criteria below. Create few equal objects (based on the below criteria) and add them to HashSet and observe the fact that Set doesn’t accept duplicates (through equals() and hashCode()).  Company and model together define the equality for Laptop. If two laptops have same company and model then they are equal. Car : make and model together define equality Television : company, type (LCD, LED, Plasma) and price together define equality CellPhone : company, model and operatingSstem together define equality School : name, city and school district together define equality  LinkedHashSet : Add all the names of our class students randomly to a LinkedHashSet, iterate through them and print. Observe that LinkedHashSet is maintaining the order that you add.  TreeSet : Add all the names of our class students randomly to a TreeSet, iterate through them and print. Observe that TreeSet keeps these elements in a natural order.  TreeSet : Create few instances of Car and add them to a TreeSet. Have Car implements Comparable interface and override compareTo() to order them by ascending order of make of the Car. Observe the fact that you need to provide the order criteria in compareTo().  Map : Create a simple HashMap to store all of our class students’ names and their favorite fruits. Try to get their favorite fruit by giving their name. Also, list all the names and their favorite fruits using keySet() and entrySet() approaches. |
| Create a class Product having following instance variables:-prodId,prodName Define class called Temp having main method. Accept 5 Product id and name from the user in main method.Store this information into product objects. Add these 5 product objects to Treemap of type product.Take product object as key and description of it as value. Eg: treeMap.put(p1,”TV”); Sort the products in Treemap in ascending order. |
| 7) Create class Order with following instance variables  private String orederId, private String orderDetails, private Date orderDate   Define the constructors to initialize all instance variables .  Perform following validations in constructor :- • orderId should be at least of 3 characters in length” • orderId should start with ‘O’.  If any validation fails, throw OrderNotValidException & display with proper message onto the console.  Create class Customer with following instance variables private int custNo private String custName  Define the constructors to initialize all instance variables .  Create class MyShopping with following members:-    private Hashtable < Customer, Order >  public void storeRecord(String filename) : To store hashtable details into file. public void getRecord(String filename) : to print record on console   Create class Shop having main method performing tasks:-  1] Accept values from user for 1 customer and 1 order objects respectively.  2] Store them into hashtable of MyShopping 3] Store objects into a file “records.dat” using storeRecord() 4] Retrieve and print records by getRecord() |
| 7) Create class Order with following instance variables  private String orderId, private String orderDetails, private Date orderDate  Define the constructors to initialize all instance variables .  Perform following validations in constructor:- • orderId should be at least of 3 characters in length” • orderId should start with ‘O’.  If any validation fails, throw OrderNotValidException & display with proper message onto the console.  Create class Customer with following instance variables private int custNo private String custName  Define the constructors to initialize all instance variables .  Create class MyShopping with following members:-    private Hashtable < Customer, Order >  public void storeRecord(String filename) : To store hashtable details into file. public void getRecord(String filename) : to print record on console   Create class Shop having main method performing tasks:-  1] Accept values from user for 1 customer and 1 order objects respectively.  2] Store them into hashtable of MyShopping 3] Store objects into a file “records.dat” using storeRecord() 4] Retrieve and print records by getRecord() |
|  |