**Question 1:**  
  
a) Create a Class Contact with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| name | String |
| company | String |
| title | String |
| mobile | String |
| alternateMobile | String |
| email | String |
| dateCreated | java.util.Date |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters, Add a default constructor and a parameterized constructor to take in all attributes in the given order: Contact(String name, String company, String title, String mobile, String alternateMobile, String email, Date dateCreated)  
  
b) Create the following static methods in the Contact class,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| static Contact createContact(String detail) | This method accepts a String. The contact detail separated by commas is passed as the argument. This method will split the details and creates a contact object and return it. |

The contact details should be given as a comma-separated value in the below order,  
name,company,title, mobile, alternateMobile, email, dateCreated  
  
c) The Contact class should implement the Comparable interface which sorts the contact list based on names. While comparing all the names in the list are unique.  
  
d) Create a class **DateComparator** which implements Comparator interface and sort the contact list based on dateCreated. While comparing all the date created attributes in the list are unique.  
  
e) Create a class **DomainComparator** which implements Comparator interface and sort the contact list based on the domain name in the email. While comparing all the email domain in the list are unique.  
  
For email id  **john@micro.com**, the domain name is **micro**.  
  
Get the number of contacts and contact details and create a contact list. Sort the contact according to the given option and display the list.  
  
When the “contact” object is printed, it should display the following details  
Print format:  
**System.out.format("%-15s %-15s %-20s %-15s %-20s %-15s %s\n", "Name","Company","Title","Mobile","Alternate Mobile","Email","Date Created");**  
  
**Sample Input and Output 1:**  
  
Enter the number of the contacts:  
**4  
John,Microsoft,Product Engineer,9876543210,7894651320,john@micro.com,12-01-2018  
Peter,Apple,Team Lead,9654873210,9873216540,peter@apple.com,15-01-2018  
Starc,IBM,HR,9654373210,8989216540,starc@ibm.com,14-01-2018  
Tony,Intel,HR,9694873210,8973212540,tony@intel.com,13-01-2018**  
Enter a type to sort:  
1.Sort by name  
2.Sort by email domain  
3.Sort by date created  
**1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Company |  | Mobile | Alternate Mobile | Email | Date Created |
| John | Microsoft | Product Engineer | 9876543210 | 7894651320 | john@micro.com | 12-01-2018 |
| Peter | Apple | Team Lead | 9654873210 | 9873216540 | peter@apple.com | 15-01-2018 |
| Starc | IBM | HR | 9654373210 | 8989216540 | starc@ibm.com | 14-01-2018 |
| Tony | Intel | HR | 9694873210 | 8973212540 | tony@intel.com | 13-01-2018 |

**Sample Input and Output 2:**  
  
Enter the number of the contacts:  
**4  
John,Microsoft,Product Engineer,9876543210,7894651320,john@micro.com,12-01-2018  
Peter,Apple,Team Lead,9654873210,9873216540,peter@apple.com,15-01-2018  
Starc,IBM,HR,9654373210,8989216540,starc@ibm.com,14-01-2018  
Mark,Intel,Manager,7994873210,9873212340,mark@intel.com,13-01-2018**  
Enter a type to sort:  
1.Sort by name  
2.Sort by email domain  
3.Sort by date created  
**3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Company |  | Mobile | Alternate Mobile | Email | Date Created |
| John | Microsoft | Product Engineer | 9876543210 | 7894651320 | john@micro.com | 12-01-2018 |
| Mark | Intel | Manager | 7994873210 | 9873212340 | mark@intel.com | 13-01-2018 |
| Starc | IBM | HR | 9654373210 | 8989216540 | starc@ibm.com | 14-01-2018 |
| Peter | Apple | Team Lead | 9654873210 | 9873216540 | peter@apple.com | 15-01-2018 |

**Sample Input and Output 3:**  
  
Enter the number of the contacts:  
**4  
John,Microsoft,Product Engineer,9876543210,7894651320,john@micro.com,12-01-2018  
Peter,Apple,Team Lead,9654873210,9873216540,peter@apple.com,15-01-2018  
Starc,IBM,HR,9654373210,8989216540,starc@ibm.com,14-01-2018  
Tony,Intel,HR,9694873210,8973212540,tony@intel.com,13-01-2018**  
Enter a type to sort:  
1.Sort by name  
2.Sort by email domain  
3.Sort by date created  
**2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Company |  | Mobile | Alternate Mobile | Email | Date Created |
| Peter | Apple | Team Lead | 9654873210 | 9873216540 | peter@apple.com | 15-01-2018 |
| Starc | IBM | HR | 9654373210 | 8989216540 | starc@ibm.com | 14-01-2018 |
| Tony | Intel | HR | 9694873210 | 8973212540 | tony@intel.com | 13-01-2018 |
| John | Microsoft | Product Engineer | 9876543210 | 7894651320 | john@micro.com | 12-01-2018 |

**Question 2:**  
a) Create a Class Mail with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| id | Long |
| to | String |
| from | String |
| subject | String |
| content | String |
| receivedDate | java.util.Date |
| size | Double |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters, Add a default constructor and a parameterized constructor to take in all attributes in the given order: Mail(Long id, String to, String from, String subject, String content,DatereceivedDate,Double size)  
  
b) Create a class **MailBO**with the following methods,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| public List<Mail>findMail(List<Mail>mailList,String to) | This method accepts a list of mails and to address as arguments and returns a list of mails that matches with given to address. |
| public List<Mail>findMail(List<Mail>mailList,DatereceivedDate) | This method accepts a list of mails and received date as arguments and returns a list of mails that were received on the given specified date. |
| public List<Mail>findMail(List<Mail>mailList,Double size) | This method accepts a list of mails and size as arguments, then find all the mails with the given size from the mail list and return the list of mails with the specified size. |

The mail details should be given as a comma-separated value in the below order,  
id,to,from, subject, content, receivedDate,size  
  
When the “mail” object is printed, it should display the following details  
Print format:  
**System.out.format("%-10s %-20s %-25s %-20s %-20s %-15s %s\n", "Id","To","From","Subject","Content","ReceivedDate","Size");**  
  
**Note:** The mail lists are displayed in the main method.  
              If any other choice is selected, display "**Invalid Choice**"  
              Display one digit after the decimal point for Double Datatype.  
  
**Sample Input and Output 1:**  
  
Enter the number of Mails:  
**4  
1001,meyyappan@gmail.com,satish@gmail.com,Master Copy,Attached doc,05-05-2017,10.0  
1010,morsh@gmail.com,meyyappan@hotmail.com,TaskList,Attached doc,25-01-2018,15.1  
1016,sami@rediff.com,morsh@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4  
1020,praveens@yahoo.com,roshan@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4**  
Enter a search type:  
1.By To Address  
2.By Received Date  
3.By Size  
**1**  
Enter the To Address:  
**praveens@yahoo.com**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id | To | From | Subject | Content | Received Date | Size |
| 1020 | praveens@yahoo.com | roshan@gmail.com | Remiander Mail | Today's schedule | 24-01-2018 | 5.4 |

**Sample Input and Output 2:**  
  
Enter the number of Mails:  
**4  
1001,meyyappan@gmail.com,satish@gmail.com,Master Copy,Attached doc,05-05-2017,10.0  
1010,morsh@gmail.com,meyyappan@hotmail.com,TaskList,Attached doc,25-01-2018,15.1  
1016,sami@rediff.com,morsh@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4  
1020,praveens@yahoo.com,roshan@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4**  
Enter a search type:  
1.By To Address  
2.By Received Date  
3.By Size  
**2**  
Enter the Received Date:  
**24-01-2018**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id | To | From | Subject | Content | Received Date | Size |
| 1016 | sami@rediff.com | morsh@gmail.com | Remainder Mail | Today's schedule | 24-01-2018 | 5.4 |
| 1020 | praveens@yahoo.com | roshan@gmail.com | Remainder Mail | Today's schedule | 24-01-2018 | 5.4 |

**Sample Input and Output 3:**  
  
Enter the number of Mails:  
**4  
1001,meyyappan@gmail.com,satish@gmail.com,Master Copy,Attached doc,05-05-2017,10.0  
1010,morsh@gmail.com,meyyappan@hotmail.com,TaskList,Attached doc,25-01-2018,15.1  
1016,sami@rediff.com,morsh@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4  
1020,praveens@yahoo.com,roshan@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4**  
Enter a search type:  
1.By To Address  
2.By Received Date  
3.By Size  
**3**  
Enter the Size:  
**5.4**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id | To | From | Subject | Content | Received Date | Size |
| 1016 | sami@rediff.com | morsh@gmail.com | Remainder Mail | Today's schedule | 24-01-2018 | 5.4 |
| 1020 | praveens@yahoo.com | roshan@gmail.com | Remainder Mail | Today's schedule | 24-01-2018 | 5.4 |

**Sample Input and Output 4:**  
  
Enter the number of Mails:  
**4  
1001,meyyappan@gmail.com,satish@gmail.com,Master Copy,Attached doc,05-05-2017,10.0  
1010,morsh@gmail.com,meyyappan@hotmail.com,TaskList,Attached doc,25-01-2018,15.1  
1016,sami@rediff.com,morsh@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4  
1020,praveens@yahoo.com,roshan@gmail.com,Remainder Mail,Today'schedule,24-01-2018,5.4**  
Enter a search type:  
1.By To Address  
2.By Received Date  
3.By Size  
**4**  
Invalid choice

**Question 3:**  
  
a) Create a Class Mail with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| id | Long |
| from | String |
| to | String |
| subject | String |
| content | String |
| receivedDate | java.util.Date |
| size | Double |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters, Add a default constructor and a parameterized constructor to take in all attributes in the given order: Mail(Long id, String from, String to, String subject, String content, Date receivedDate, Double size)  
  
b) Create the following static methods in the Mail class,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| static Mail createMail(String detail) | This method accepts a String. The mail detail separated by commas is passed as the argument. Split the details and create a mail object and returns it. |

The mail details should be given as a comma-separated value in the below order,  
id,from,to,subject,content,receivedDate,size  
  
c) The Mail class should implement the **Comparable** interface which sorts the Mail list based on from address. While comparing, all the from address in the list are unique.  
  
d) Create a class **DateComparator** which implements Comparator interface and sort the Mail list based on receivedDate. While comparing, all the receivedDate attributes in the list are unique.  
  
e) Create a class **SizeComparator** which implements Comparator interface and sort the Mail list based on the size. While comparing, all the size attributes in the list are unique.  
  
Get the number of Mail and mail details and create a mail list. Sort the mail according to the given option and display the list.  
  
When the “mail” object is printed, it should display the following details  
Print format:  
**System.out.format("%-15s %-15s %-15s %-20s %-20s %-15s %s\n", "Id","From","To","Subject","Content","Receiveddate","Size");**  
Display one digit after decimal point for Double datatype.  
  
**Sample Input and Output 1:**  
  
Enter the number of mails:  
**4  
1,raj@abc.in,bala@abc.in,Freshers' list,PFA the Freshers db,05-01-2018,256  
2,amir@abc.in,chris@abc.in,M.O.M,PFA the M.O.M,29-01-2018,678  
3,abdul@abc.in,antony@abc.in,project requirement,PFA the requirements,20-01-2018,1658  
4,karim@abc.in,krishna@abc.in,Accounts,PFA the accounts,27-01-2018,2048**  
Enter a type to sort:  
1.Sort by from address  
2.Sort by date received  
3.Sort by size  
**1**  
Id       From                      To                               Subject                              Content                                  Received date     Size  
3        abdul@abc.in    antony@abc.in   project requirement    PFA the requirements    20-01-2018            1658.0  
2        amir@abc.in       chris@abc.in        M.O.M                                 PFA the M.O.M                   29-01-2018            678.0  
4        karim@abc.in     krishna@abc.in  Accounts                          PFA the accounts             27-01-2018            2048.0  
1        raj@abc.in            bala@abc.in       Freshers' list                    PFA the Freshersdb       05-01-2018            256.0

**Question 4:**  
  
a) Create a Class Mail with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| id | String |
| from | String |
| to | String |
| subject | String |
| content | String |
| receivedDate | java.util.Date |
| size | Double |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters. Add a default constructor and a parameterized constructor to take in all attributes in the given order: Mail(Long id, String from, String to, String subject, String content, Date receivedDate, Double size)  
  
b) Create the following static methods in the Mail class,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| static Map<Date,Integer>calculateDateCount(List<Mail> list) | This method accepts a list of Mail as arguments and returns a TreeMap with the receivedDate as key and number of mails received on that date as value and returns the map. |

In the TreeMap have the receivedDate as key and Count the number of mails received on the date and keep the number of mails as value. Print the value sorted by Date.  
  
The mail details should be given as a comma separated value in the below order,  
id,from,to,subject,content,receivedDate,size  
  
Print format:  
**System.out.format("%-15s %s\n","Date","Count");**  
  
**Sample Input and Output 1:**  
  
Enter the number of mails:  
**5  
218,krish@abc.com,ram@abc.com,Freshers' list,PFA the Freshers db,05-10-2016,1256  
569,ani@abc.com,gautham@abc.com,M.O.M,PFA the M.O.M,05-10-2016,1896  
538,ram12@abc.com,ravi@abc.com,project requirement,PFA the requirements,20-11-2011,1058  
102,ganesh@abc.com,chris@abc.com,Accounts,PFA the accounts,20-11-2011,48  
997,sam@abc.com,rahim@abc.com,Appraisal,PFA appraisal db,27-01-2007,2148**  
Date                      Count  
27-01-2007      1  
20-11-2011      2  
05-10-2016      2

**Question 5:**  
  
a) Create a Class Player with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| name | String |
| dateOfBirth | java.util.Date |
| skill | String |
| numberOfMatches | Integer |
| runs | Integer |
| wickets | Integer |
| nationality | String |
| powerRating | Double |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters, Add a default constructor and a parameterized constructor to take in all attributes in the given order: Player(String name, Date dateOfBirth, String skill, Integer numberOfMatches, Integer runs,Integer wickets,String nationality,Double powerRating)  
  
b) Create a class **PlayerBO** with the following methods,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| public List<Player> findPlayer(List<Player> playerList,String nationality) | This method accepts a list of players and Nationality as arguments and returns a list of players that match with given Nationality. |
| public List<Player> findPlayer(List<Player> playerList,Date dateOfBirth) | This method accepts a list of players and date of birth as arguments and returns a list of players who were born on the given date. |
| public List<Player> findPlayer(List<Player> playerList,Double powerRating) | This method accepts a list of players and power rating as arguments, then find all the players with the given power rating from the player list and return the list of players with the specified power rating. |

The player details should be given as a comma-separated value in the below order,  
name,dateOfBirth,skill,numberOfMatches,runs,wickets,nationality,powerRating  
  
Get the number of player and the player details, build a player list and perform the search by nationality,dateOfBirth or powerRating.  
  
When the “player” object is printed, it should display the following details  
Print format:  
**System.out.format("%-15s %-15s %-15s %-20s %-15s %-15s %-15s %s\n","Name","Date of Birth","Skill","Number of Wickets","Runs","Wickets","Nationality","Power Rating");**  
  
**Note:** The player lists are displayed in the main method.  
              If any other choice is selected, display "**Invalid choice**"  
              Display one digit after the decimal point for Double Datatype.  
  
**Sample Input and Output 1:**  
  
Enter the number of Players:  
**4  
MSD,07-07-1981,WK&BAT,300,10000,2,Indian,4.9  
ABD,17-02-1984,WK&BAT,300,10000,3,South African,4.9  
SRaina,27-11-1986,BAT,100,5000,50,Indian,4.5  
Maxwell,14-10-1988,ALLROUNDER,200,4000,50,Australian,4.5**  
Enter a search type:  
1.By Nationality  
2.By Date of Birth  
3.By Power Rating  
**1**  
Enter the Nationality:  
**Australian**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Name | Date of Birth | Skill | Number of Matches | Runs | Wickets | Nationality | Power Rating |
| Maxwell | 14-10-1988 | ALLROUNDER | 200 | 4000 | 50 | Australian | 4.5 |

**Question 6:**  
  
In this requirement, given a list of players you need to find the number of players playing for a country using Map.  
  
a) Create a Class Player with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| name | String |
| dateOfBirth | java.util.Date |
| skill | String |
| numberOfMatches | Integer |
| runs | Integer |
| wickets | Integer |
| nationality | String |
| powerRating | Double |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters. Add a default constructor and a parameterized constructor to take in all attributes in the given order:  
**Player(String name, java.util.Date dateOfBirth, String skill, Integer numberOfMatches, Integer runs, Integer wickets, String nationality, DoublepowerRating)**  
  
b) Create the following static methods in the **Player** class,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| static Map<String,Integer> calculateNationalityCount(List<Player> list) | This method accepts a list of Player as arguments and returns a TreeMap with the nationality as key and number of players playing for the country as value and returns the map. |

In the TreeMap have the nationality as key and Count the number of players playing for the country and keep the number of players as value. Print the value sorted by country name.  
  
The player details should be given as a comma separated value in the below order,  
name, dateOfBirth, skill, numberOfMatches, runs, wickets, nationality, powerRating  
  
Print format:  
**System.out.format("%-15s %s\n","Country","Count");**  
  
**Sample Input and Output 1:**  
  
Enter the number of players:  
**5  
MS Dhoni,07-07-1981,Batsman,159,3561,0,India,4.4  
Virat Kohli,05-11-1988,Batsman,149,4418,4,India,4.7  
Bhuvneshwar,05-02-1990,Bowler,90,158,111,India,4.1  
Mike hussey,27-05-1975,Batsman,59,1977,0,Australia,4.5  
Mitchell Stark,30-01-1990,Bowler,27,96,34,Australia,4**  
Country         Count  
Australia       2  
India              3

**Question 7:**

Usually, As part of batch processing jobs, a CSV or TXT file is read, relevant objects are created and database is populated with CSV contents.  
Create the list of customer objects with the CSV content provided in the sample IO.

Create a class named as **Customer**, which contains following private variable/ attributes,   

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Member Field name** | **Type** | | id | Long | | name | String | | gender | Character (M/F) | | email | String | | contactNumber | String | | createdOn | Date (time in 24 hrs clock) dd/MM/yyyy HH:mm:ss | |  |

Mark all the attributes as private

Create / Generate appropriate Getters & Setters.

Add a default constructor and a parameterized constructor to take in all attributes.

In the Customer class, implement the following methods.

|  |  |  |
| --- | --- | --- |
| **No** | **Method Name** | **Method Description** |
| 1 | public static List<Customer> populateCustomers(List<String> csvList) | In this method, given parameter is the list of customer details in a string format where each data is separated by a comma. Parse the string and create a customer arrayList. |
| 2 | public static List<Customer> findCustomerNameFromList(List<Customer> customers, String subString) | In this method, given part of customer name, search the customer list based on name and return the customer list with matching names. |

**Input format:**  
The first input consists of an integer that corresponds  to the number of customer  n.  
The next n line of the input consists of a string that corresponds to the customer details, which is separated  by a comma.  
Input sequence:  
**id, name, gender, email, contactNumber, createdOn.**  
The last input is the substring that used to search the specified customers.  
  
**Output format:**  
Refer Sample Input and Output.  
  
HINT:  
The implementation can either be done in the BO class or static method in the customer class.  
In real time projects, its done in the BO Class and a fallback is given the customer class as static method.  
Ensure the static methods in the Customer class is present.  
The implementation can be in the BO layer with the static methods calling the methods in BO layer.  
  
Main - Customer class static methods - CustomerBO methods.  
  
  
**Sample Input and Output:**

Enter the number of customer:

**5**

Enter the customer 1 detail:

**12,John Smith,M,**[**johnsmith@a.com**](mailto:johnsmith@a.com)**,+85-7489-8596478596,12/12/2016 12:30:00**

Enter the customer 2 detail:

**15,Tedmond,M,**[**tedmond@a.com**](mailto:tedmond@a.com)**,+45-9857-5266987485,14/01/2017 04:30:00**

Enter the customer 3 detail:

**11,Dalton,M,**[**dalton@a.com**](mailto:dalton@a.com)**,+48-8967-7485947558,12/02/2017 20:00:00**

Enter the customer 4 detail:

**5,Raymond,M,**[**raymond@a.com**](mailto:raymond@a.com)**,+88-8745-8554712569,28/01/2017 10:30:00**

Enter the customer 5 detail:

**9,Ruford,M,**[**ruford@a.com**](mailto:ruford@a.com)**,+88-4859-7714589633,01/04/2017 17:45:00**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Name | Gender | Email | Contact no | Created on |
| 12 | John Smith | M | [johnsmith@a.com](mailto:johnsmith@a.com) | +85-7489-8596478596 | 12/12/2016 12:30:00 |
| 15 | Tedmond | M | [tedmond@a.com](mailto:tedmond@a.com) | +45-9857-5266987485 | 14/01/2017 04:30:00 |
| 11 | Dalton | M | [dalton@a.com](mailto:dalton@a.com) | +48-8967-7485947558 | 12/02/2017 20:00:00 |
| 5 | Raymond | M | [raymond@a.com](mailto:raymond@a.com) | +88-8745-8554712569 | 28/01/2017 10:30:00 |
| 9 | Ruford | M | [ruford@a.com](mailto:ruford@a.com) | +88-4859-7714589633 | 01/04/2017 17:45:00 |

Enter the substring to search from customer list:

**mon**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Name | Gender | Email | Contact no | Created on |
| 15 | Tedmond | M | [tedmond@a.com](mailto:tedmond@a.com) | +45-9857-5266987485 | 14/01/2017 04:30:00 |
| 5 | Raymond | M | [raymond@a.com](mailto:raymond@a.com) | +88-8745-8554712569 | 28/01/2017 10:30:00 |

**Question 8:**

Create a class named as **Customer**, which contains following private variables/ attributes,

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Member Field name** | **Type** | | id | Long | | name | String | | gender | Character (M/F) | | email | String | | contactNumber | String | | createdOn | Date (time in 24 hrs clock) dd/MM/yyyy HH:mm:ss | | address | Address | |

Mark all the attributes as private

Create / Generate appropriate Getters & Setters.

Add a default constructor and a parameterized constructor to take in all attributes.

|  |  |  |
| --- | --- | --- |
| **No** | **Method Name** | **Method Description** |
| 1 | public static HashMap<String, Integer> convertCsvToMap(List<String> csvDetails) | This method returns a Hashmap <String state, Integer count>. The key would be the state name and count would be the number of customers in the particular state |

Create a class named as **Address**, which contains following private variable/ attributes,

|  |  |
| --- | --- |
| **Member Field name** | **Type** |
| street | String |
| city | String |
| state | String |
| country | String |
| zipCode | Integer |

Include appropriate getters and setters.  
Add a default constructor and a parameterized constructor to take in all attributes.

**Hint:** You can get all the keys from hashmap and create a sorted list.

State – sorted based on string ascending order.

**Input Format:**  
  
The first input consists of an integer that corresponds to the number of customers n.  
The next n input consists of a string that corresponds to the customer details, which is separated by the comma (,).  
Input sequence - id,name,email,state,country.  
  
**Output Format:**  
  
The first list format is,  
**"%-15s %s\n","State","No of customer(s)"**  
  
**Sample Input and Output:**

Enter the number of customer:

**10**

Enter the customer 1 details:

**1,John Smith,M,**[**johnsmith@a.com**](mailto:johnsmith@a.com)**,+89-7485-8578974885,15/01/2016 10:30:00,112th St,Utica,New York,USA,13455**

Enter the customer 2 details:

**2,Aekerman,M,**[**aekerman@a.com**](mailto:aekerman@a.com)**,+99-7489-8857945569,14/02/2016 16:30:00,Avenue,Austin,Texas,USA,88596**

Enter the customer 3 details:

**3,Madeleine,F,**[**madeleine@a.com**](mailto:madeleine@a.com)**,+88-7859-7748599989,25/01/2016 10:00:00,155th St,Plano,Texas,USA,56684**

Enter the customer 4 details:

**4,Edrick,M,**[**edrick@a.com**](mailto:edrick@a.com)**,+99-7482-4115233987,18/12/2016 07:30:00,111th St,Sitka,Alaska,USA,66584**

Enter the customer 5 details:

**5,Tedmond,M,**[**tedmond@a.com**](mailto:tedmond@a.com)**,+77-8599-4225610074,05/01/2016 08:30:00,Parc St,Olympia,Washington,USA,85574**

Enter the customer 6 details:

**6,Nelson,M,**[**nelson@a.com**](mailto:nelson@a.com)**,+78-7488-4221258447,02/01/2017 10:30:00,5th St,Dayton,Ohio,USA,84587**

Enter the customer 7 details:

**7,Dalton,M,**[**dalton@a.com**](mailto:dalton@a.com)**,+78-8547-8555479512,15/01/2017 20:30:00,North St,Buffallo,New York,USA,25664**

Enter the customer 8 details:

**8,Raymond,M,**[**raymond@a.com**](mailto:raymond@a.com)**,+89-7484-8577458895,25/02/2017 10:30:00,15th St,Waco,Texas,USA,7858**

Enter the customer 9 details:

**9,Rosemary,F,**[**rosemary@a.com**](mailto:rosemary@a.com)**,+88-4888-7485998741,20/02/2017 12:30:00,15th St,Tacoma,Washington,USA,87458**

Enter the customer 10 details:

**10,Ruford,M,**[**ruford@a.com**](mailto:ruford@a.com)**,+84-422-9887485995,22/03/2017 9:30:00,9th St,Butte,Montana,USA,22458**

|  |  |
| --- | --- |
| State | No of customer(s) |
| Alaska | 1 |
| Montana | 1 |
| New York | 2 |
| Ohio | 1 |
| Texas | 3 |
| Washington | 2 |

**Question 8:**

a) Create a Class Event with the following attributes:

Member Field Name Type

name String

organizer String

category String

startDate Date

endDate Date

hallName String

Mark all the attributes as private, Create / Generate appropriate Getters & Setters, Add a parameterized constructor to take in all attributes in the given order: Event(String name, String organizer, String category, Date startDate, Date endDate, String hallName)

b) Create the following static methods in the Transaction class,

Method Name

static Map<String,Integer> categoryWiseCount(List<Event> list)

This method accepts a List of Event objects.

It returns a Map with category as key and the number of events on that category as value.

The user details should be given as a comma-separated value in the below order,

name, organizer, category, startDate, endDate, hallName

Note:

Use System.out.format("%-20s %s\n","Category","Count") for formatting the output.

Sample Input and Output:

Enter the number of events

4

Oktoberfest,Oliver,Dance,12-10-2018,13-10-2018,Opera Hall

Music&Arts Event,Harry,Music,10-10-2018,11-10-2018,Symphony Hall

SuperBowl,Joe,Game,03-10-2018,05-10-2018,Concert Hall

Baloon Event,Danny,Game,01-10-2018,02-10-2018,Chamber Hall

Category Count

Dance 1

Game 2

Music 1