Page of

**Batch Processing**

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,+85-7489-8596478596,12/12/2016 12:30:00**

Enter the customer 2 detail:

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

Enter the customer 3 detail:

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

Enter the customer 4 detail:

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

Enter the customer 5 detail:

**9,Ruford,M,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 | +85-7489-8596478596 | 12/12/2016 12:30:00 |
| 15 | Tedmond | M | tedmond@a.com | +45-9857-5266987485 | 14/01/2017 04:30:00 |
| 11 | Dalton | M | dalton@a.com | +48-8967-7485947558 | 12/02/2017 20:00:00 |
| 5 | Raymond | M | raymond@a.com | +88-8745-8554712569 | 28/01/2017 10:30:00 |
| 9 | Ruford | M | 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 | +45-9857-5266987485 | 14/01/2017 04:30:00 |
| 5 | Raymond | M | raymond@a.com | +88-8745-8554712569 | 28/01/2017 10:30:00 |