**Scenario:**

“Tarangini Digital Services” is a DTH operator. They are looking at developing an application to let their subscribers manage their subscriptions.

As the application is under development, as an immediate requirement this organization is looking at receiving subscriber details from the local vendors in the form of a text file as attached below.



**Task**

You have been assigned with a high priority task of implementing **THREE** functions to the existing **Subscriber Management System.** The below attached Java file (SubscriberManagementService.java) has following classes definitions. Use this Java file to implement the functions.

1. SubscriberManagementService– that tracks the subscriber details.
2. SubscriberVO – Object representation of an subscriber record.
3. InvalidSubscriberException – User defined unchecked exception to be thrown in case of validation failure of subscriber record.
4. SubscriberParseException – User defined exception representing abnormalities during subscriber detail processing.



***Points to remember***

1. *You can use STS/Eclipse IDE with JDK1.8 to develop the code for the implementation. Ensure that you follow best Java coding practice while coding.*
2. *Do not modify any attribute names and method signatures provided as part of the given Java file.*
3. *Code the implementation in the respective methods of the given class ONLY.*
4. *Do not hard code the return values of methods, if found your code will not be considered for processing.*
5. *Use Java SE 8.0 API only for the function implementation.*
6. *Upload/Submit the updated ActivityManager.java file for evaluation.*

Following are the THREE functions to be implemented.

1. **Function 1 – Loading Value objects with subscriber details**

***Description***

**Read the file “subscriber**.txt” from a file path (record by record) and populate the field values in value objects. One value object for one record. Store all the value objects in a List and return the list. Handle all possible exceptions with respective handlers.

***Implementation***

Class: **SubscriberManagementService**

Method: **public List<SubscriberVO> loadAllSubscribers(String fileName) throws SubscriberParseException**

You can write additional utility methods wherever needed.

***Input***

Absolute path of the file “subscribers.txt”.

***Output:***

List of value objects representing activities.

***Validations***

While reading the subscriber records from the file ensure that the following validations are performed, if violated raise InvalidSubscriberException and skip that record.

|  |  |
| --- | --- |
| **Field** | **Constraints** |
| Subscriber Id | Cannot be negative or Zero |
| First name | Cannot be blank.  Must be of 4 to 25 characters in length  Should not have any characters other than space and alphabet(both lower and capital). |
| Last name | Can be blank.  Must be of 4 to 25 characters in length  Should not have any characters other than space and alphabet (both lower and capital). |
| Packages | A list of non-repetitive package names |
| Date of registration | Must be an older or present date. |
| BillingTerm | Annual/Monthly |

***Constraints:***

Use java.time.format.DateTimeFormatter while formatting the Date entries.

Skip the first line in the input text file “subscribers.txt” as it contains headings.

1. **Function 2 – Retrieving sorted set of distinct package names.**

***Description***

**Read the file “subscribers.**txt” from a file path (record by record) and return a SortedSet of distinct package names.

***Implementation***

Class: **SubscriberManagementService**

Method: **public SortedSet<String> getDistinctPackages(String fileName) throws SubscriberParseException**

You can write additional utility methods wherever needed.

***Input***

Absolute path of the file “subscribers.txt”.

***Output***:

SortedSet of distinct package names.

*Validations*

Throw SubscriberParseException if anything goes wrong during the output construction.

*Constraints*:

Use loadAllSubscribers () method to read the subscriber records from the input file.

Use ONLY java.util.stream package and lambda expression.

*Sample Input:*

**subscribers.txt**

SubscriberId,firstName,lastName,Packages,MobileNumber,registrationDate,billingTerm

101,Srinivas,Dachepalli,SouthPlus;News,9247175830,10-Jan-21,Monthly

102,Lakshmi,Manchu,SouthPlus;Kids;Sports,9247212345,10-Jan-21,Annually

103,Sudheer,Yalamanchili,NorthPlus;Sports;Movies,9848012345,11-Jan-21,Annually

104,Anasuya,Ramalingam,NorthPlus;Kids;Movies,9848023456,11-Jan-21,Monthly

105,Rashmi,Goutham,SouthPlus;Movies;News,8093466666,12-Jan-21,Monthly

106,Adi,Hyper,NorthPlus;News,8093477777,12-Jan-21,Monthly

107,Rakesh,Rocking,SouthPlus;Kids;Sports,8093456565,10-Jan-21,Monthly

108,Abhi,Attili,SouthPlus;Movies;Kids,8093478789,11-Jan-21,Annually

109,Ramesh,Rachakonda,NorthPlus;Movies,9247212541,10-Jan-21,Monthly

110,Dora,Babu,SouthPlus;Sports;Movies;News,9247112350,11-Jan-21,Annually

*Sample Output:*

**SortedSet of distinct package names.**

**"SouthPlus","NorthPlus","Kids","Movies","News","Sports"**

1. **Function 3 – Retrieving set of fullNames of subscribers package wise.**

***Description***

**Read the file “subscribers.**txt” from a file path (record by record) and return a Map with the keys as package names and the values as Set of Full names of subscribers who subscribe for that package name.

***Implementation***

Class: **SubscriberManagementService**

Method: **public Map<String,List<String>> getPackageWiseSubscribers(String fileName) throws SubscriberParseException**

You can write additional utility methods wherever needed.

***Input***

Absolute path of the file “subscribers.txt”.

***Output***:

Map<String,List<SubscriberVO>> having packageName as key and value as set of subscriber fullNames.

*Validations*

Throw SubscriberParseException if anything goes wrong during the output construction.

*Constraints*:

Use loadAllSubscriberDetails() method to read the subscriber records from the input file.

Use ONLY java.util.stream package and lambda expression while creating the result Map.

Full name of a subscriber is concatenation of firstName and a space and lastName in that order only.

*Sample Input:*

**subscribers.txt**

SubscriberId,firstName,lastName,Packages,MobileNumber,registrationDate,billingTerm

101,Srinivas,Dachepalli,SouthPlus;News,9247175830,10-Jan-21,Monthly

102,Lakshmi,Manchu,SouthPlus;Kids;Sports,9247212345,10-Jan-21,Annually

103,Sudheer,Yalamanchili,NorthPlus;Sports;Movies,9848012345,11-Jan-21,Annually

104,Anasuya,Ramalingam,NorthPlus;Kids;Movies,9848023456,11-Jan-21,Monthly

105,Rashmi,Goutham,SouthPlus;Movies;News,8093466666,12-Jan-21,Monthly

106,Adi,Hyper,NorthPlus;News,8093477777,12-Jan-21,Monthly

107,Rakesh,Rocking,SouthPlus;Kids;Sports,8093456565,10-Jan-21,Monthly

108,Abhi,Attili,SouthPlus;Movies;Kids,8093478789,11-Jan-21,Annually

109,Ramesh,Rachakonda,NorthPlus;Movies,9247212541,10-Jan-21,Monthly

110,Dora,Babu,SouthPlus;Sports;Movies;News,9247112350,11-Jan-21,Annually

*Sample Output:*

**Map<String,List<SubscriberVO>> having packageName as key and value as set of subscriber fullNames.**

|  |  |
| --- | --- |
| **Key** | **Value** |
| Kids | ["Lakshmi Manchu","Anasuya Ramalingam","Rakesh Rocking","Abhi Attili"] |
| Movies | ["Sudheer Yalamanchili","Anasuya Ramalingam","Rashmi Goutham","Abhi Attili", "Ramesh Rachakonda", "Dora Babu"] |
| News | ["Srinivas Dachepalli","Rashmi Goutham", "Aditya Hyper", "Dora Babu"] |
| NorthPlus | ["Sudheer Yalamanchili","Anasuya Ramalingam", "Aditya Hyper", "Ramesh Rachakonda"] |
| SouthPlus | ["Srinivas Dachepalli","Lakshmi Manchu", "Rashmi Goutham", "Rakesh Rocking", "Abhi Attili", "Dora Babu"] |
| Sports | ["Lakshmi Manchu", "Sudheer Yalamanchili","Rakesh Rocking", "Dora Babu"] |