The City College of New York, CUNY
Department of Compute Science
North Academic Center, Room 8/206
160 Convent Avenue, New York, NY 10031

Assignment 3 – Fall 2020

Due Date: *by Sunday October 18, 2020 11:59PM*
How to submit: *upload JAVA files to Blackboard*

Network Log Utility.

Log files contain information about the running of a specific software. Network files, for instance, contain network-related communication that occur between devices. The log may include information about the connection's that include timestamp, source and destination IP addresses, packet sizes, and comments. Due to their size and structure, software is need to analyze and extract patterns from a log file's content. In this assignment, you are asked to create a simple Java program which provides a set of methods to parse, filter, and query a network log file. Your program must consist of two Java classes, a read-only class container for each log entry and a utility class. A test class is provided for this assignment.

Note:
- ✓ ***This is an individual assignment; please do your own work, sharing and/or copying code and/or solution ideas with/from others will result in a grade of 0 and disciplinary actions for all involved parties. If you run into problems and have done your best to solve them, please contact me before/after class or by e-mail.***
- ✓ ***A 20% grade deduction for every day the assignment is late.***

Class' Description:

I. $LogEntry$:

- ✓ Non-default constructor, initializes the class' fields after validating the information. If any of the values do not conform to the restrictions shown in Figure 1, the constructor throws an exception of type $InstantiationException$ with the message "$One\ or\ more\ values\ are\ invalid$"
  Hint:
  - A $Strings$ maybe converted to an int using the method $Integer.parseInt(\ )$. e.g. $int\ i = Integer.parseInt("55");$
- ✓ 7 getters methods for each of the class' fields.
- ✓ $getDateTimeAsDate$, returns the value of $dateTime$ field as a $Date$ object. Use $SimpleDateFormat$ with pattern "$MM/dd/yyy\ HH:mm:ss$". Note that Java has multiple $Date$ classes, make sure to use $java.util.Date$
- ✓ $toString$: returns a comma delimited String consisting of the class' 7 fields.

$$\ll value\ of\ sequence \gg, \ll value\ of\ dateTime \gg, \ldots$$

II. $LogUtility$:

- ✓ $default\ constructor$: initializes the $final$ field $listLogEntry$.
- ✓ $parseFile$: reads all lines from the provided text file ($Network.log$). The method extracts a line from the text file, parses the line by the comma delimiter, and creates a $LogEntry$ instance. If the instance is created successfully, the instance is added to the $listLogEntry$ field. If an exception is raised from the $LogEntry$ constructor, print the message: $Skipping\ line: \ll the\ line\ being\ skipped \gg$. See Figure 3

  Hints:
  - The $FileNotFoundException$ should not be handled within this method.
  - Use $Scanner$ class to read the input file's lines. This is similar to reading from the standard command line.
  - Use the String class' split method to parse each line. The split method returns an array:

$$String\ line = "a, b, c, d";$$
$$String\ arr[\ \ ] = line.split(",");$$

- ✓ $toString$: returns the following String. Replace ☺ with the number of entries in $listLogEntries$. Figure 3

$$LogUtility: there\ are\ ☺\ records$$

- ✓ $getBetween$: returns a list of records between the date string parameters (inclusive). The two parameters are the start and end dates respectively. First, convert the parameters from $String$ to $Date$ in order to perform the search. $ParseException$ should be handled by this method.

  Hint:

  ☞ The $Date$ class provides the $compareTo$ method for date-to-date object comparison. The methods returns:
  1. 0, if the two date objects are equal
  2. A number < 0, if the $left$ object is chronologically smaller
  3. A number > 0, if the $right$ object is chronologically smaller
- ✓ $findFirst*$: the remaining 7 required methods allow searching by a specific field. For simplicity, these methods return only the first occurrence that matches the search criteria (i.e. parameter). Please note that it is highly inefficient to implement each method independently, therefore you must create a common method which contains the search subroutine. This method should select the proper field and perform the search.

  Hint:

  ☞ One way to distinguish between the fields in the common method is to create an enumerator and use a switch structure to select the proper field.

III. $TestUtility$:

- ✓ This is the provided test class. Your code should work with this class AS IS. Please DO NOT modify this class and adhere to the names provided in the class' UML diagram (Figure 1). You may, however, comment lines of code until you are ready to test the methods they invoke.
- ✓ Your code's output should match the output shown in Figure 3

What to submit:

1) $LogEntry.java$
2) $LogUtility.java$

Grading:

Criteria	Points
Class LogEntry	
Constructor – validation and exception handling	10
Getters	5
getDateTimeAsDate()	5
toString	5
Class LogUtility	0
listLogEntries	2
Constructor	3
parseFile – read file, parse, add to listLogEntries, handle exception, and close resources	10
toString	5
getBetween	15
7 $findFirst*$ methods	10
Common method for 7 $findFirst*$ methods	20
Correct output	10
	100

Figures:

Column #	Description	Data Type	Restrictions
1	Sequence Number	Integral	Must be between 1 and *Integer.MAX_VALUE*
2	Date and Time	Sting	Exactly 19 characters
3	Source IP Address	Sting	Cannot be empty or null
4	Destination IP Address	Sting	Cannot be empty or null
5	Protocol	Sting	Either TCP or UDP
6	Packet Size	Integral	Between 1 and 1,500
7	Comment	Sting	No restrictions

Figure 1: Log File Description

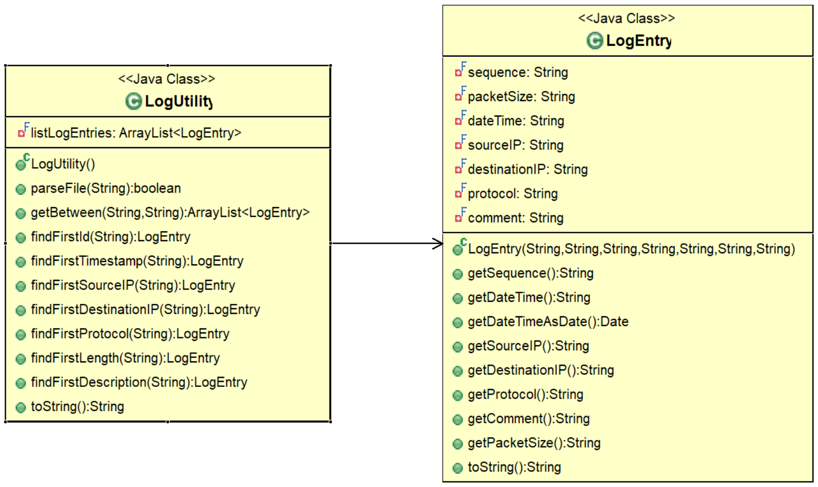
<<Java Class>>
LogUtility
listLogEntries: ArrayList<LogEntry>
LogUtility()
parseFile(String):boolean
getBetween(String,String):ArrayList<LogEntry>
findFirstId(String):LogEntry
findFirstTimestamp(String):LogEntry
findFirstSourceIP(String):LogEntry
findFirstDestinationIP(String):LogEntry
findFirstProtocol(String):LogEntry
findFirstLength(String):LogEntry
findFirstDescription(String):LogEntry
toString():String
<<Java Class>>
LogEntry
sequence: String
packetSize: String
dateTime: String
sourceIP: String
destinationIP: String
protocol: String
comment: String
LogEntry(String,String,String,String,String,String,String)
getSequence():String
getDateTime():String
getDateTimeAsDate():Date
getSourceIP():String
getDestinationIP():String
getProtocol():String
getComment():String
getPacketSize():String
toString():String

Figure 2: Class UML Diagram

```
Instantiating LogUtility
LogUtility: there are 0 records
----------------------------------------------------------------------------------------------------------------------------------
Skipping line: #,Time Stamp,Source,Destination,Protocol,Packet Size,Comment
Skipping line: 9412,02/19/2018 15:15:17,192.168.5.241,.55.163.53,TCP,0,TCP communication with 192.168.5.152
Skipping line: 2686,08/06/2019 11:15:16,192.168.5.147,51.101.65.140,TCP,2662,TCP communication with quora.com
Skipping line: 4366,08/18/2019 15:40:53,192.168.5.173,51.101.65.140,TCP,0,TCP communication with amazon.com
Skipping line: 4928,02/19/2015 07:34:57,192.168.5.108,172.217.11.14,-,803, communication with 192.168.5.152
Skipping line: ,05/03/2020 16:56:21,192.168.5.135,51.101.66.49,TCP,1099, communication with etsy.com
Skipping line: 4733,09/15/2017 03:28:19,192.168.5.233,52.6.240.48,-,8, communication with homedepot.com
Skipping line: 5804,01/11/2019 20:47:24,192.168.5.65,.101.1.224,-,1424, communication with nytimes.com
Skipping line: 1172,08/22/2020 21:40:35,192.168.5.194,51.101.130.187,TCP,1859,TCP communication with forbes.com
Skipping line: 7960,08/06/2016 20:18:38,192.168.5.245,192.229.189.15,-,119, communication with etsy.com
Skipping line: 7694,08/28/2016 06:55:01,192.168.5.221,34.194.103.209,-,688, communication with 192.168.5.120
Skipping line: 2909,07/08/2017 04:34:25,192.168.5.203,192.168.5.177,-,871, communication with ebay.com
Skipping line: 6268,04/11/2015 09:52:55,192.168.5.150,51.101.65.140,-,31, communication with indeed.com
Skipping line: 6464,07/26/2019 16:23:59,192.168.5.55,172.217.10.238,TCP,2528,TCP communication with facebook.com
Skipping line: 7193,11/10/2016 07:34:04,192.168.5.141,192.168.5.152,-,134, communication with facebook.com
Skipping line: 5828,12/15/2017 05:58:04,192.168.5.89,192.168.5.120,-,1192, communication with linkedin.com
Skipping line: 7318,11/13/2017 09:17:55,192.168.5.254,31.13.66.35,TCP,1561,TCP communication with 192.168.1.1
Skipping line: 1492,12/17/2017 08:26:05,192.168.5.143,192.168.5.152,-,974, communication with tripadvisor.com
Skipping line: 2362,04/08/2020 18:46:55,192.168.5.223,192.168.5.120,-,708, communication with 192.168.5.177
Skipping line: 7946,07/21/2016 04:41:52,192.168.5.238,151.101.128.84,-,1161, communication with apple.com
Skipping line: 9561,04/17/2017 06:08:08,192.168.5.128,.101.1.224,-,1, communication with 192.168.5.244
Skipping line: 1965,02/26/2020 19:45:29,192.168.5.229,34.194.103.209,-,1176, communication with 192.168.5.152
Skipping line: 4833,11/21/2017 03:29:12,192.168.5.208,52.6.240.48,-,845, communication with 11
Skipping line: 5625,08/04/2018 08:09:11,192.168.5.110,192.168.5.120,-,574, communication with 192.168.1.1
Skipping line: 7327,09/11/2017 16:29:36,192.168.5.229,192.168.5.148,UDP,1922,UDP communication with indeed.com
Skipping line: 1651,02/13/2018 02:21:03,192.168.5.109,151.101.128.84,TCP,1549,TCP communication with youtube.com
Skipping line: 7058,01/22/2019 21:35:34,192.168.5.233,172.217.10.238,TCP,0,TCP communication with reddit.com
Skipping line: 7637,04/17/2018 20:57:11,192.168.5.214,34.194.103.209,TCP,2266,TCP communication with 192.168.5.197
Skipping line: 5936,11/09/2015 03:19:15,192.168.5.209,192.168.5.152,,822, communication with nytimes.com
Skipping line: 5083,01/01/2020 12:11:11,192.168.5.79,13.77.161.179,TCP,2118, communication with ebay.com
----------------------------------------------------------------------------------------------------------------------------------
LogUtility: there are 471 records
----------------------------------------------------------------------------------------------------------------------------------
First record with id 1: 4276,09/25/2016 13:38:54,192.168.5.93,51.101.130.187,TCP,TCP communication with youtube.com,1399
First record with id 9: 5965,10/19/2019 07:18:39,192.168.5.163,17.172.224.47,TCP,TCP communication with youtube.com,1196
----------------------------------------------------------------------------------------------------------------------------------
There are 88 entries from 2019
```

Figure 3: Test Class' Output

The City College of New York, CUNY
Department of Compute Science
North Academic Center, Room 8/206
160 Convent Avenue, New York, NY 10031

UML Diagram Legend

Symbol	Description
Underlined	Indicates a static member
🟥	A private member (i.e. variable or method)
🟥F	A private final member (i.e. variable)
🟢	A public field (i.e. variable or method)
🟢A	A public abstract member (i.e. variable or method)
🟢C	A public constructor
🟢S	A static public member
Ⓘ	An interface
Ⓒ	A public class
ⒸA	A public abstract class
—▷	A hollowed arrow indicates inheritance
→	An open-ended arrow indicates composition
┈▷	A dotted line and hollowed arrow indicate class implementation