Exercise 12.00

Imagine a system in which you (as a feature) are going to log all actions, e.g. into a file. Example:
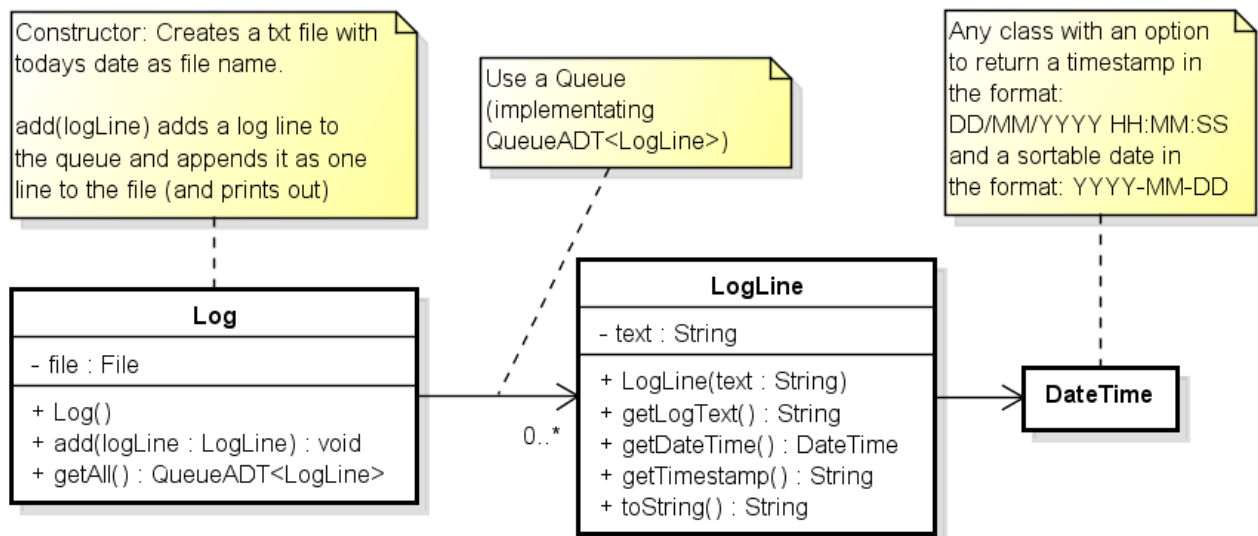
"01/03/16 12:11:16 removing a cd has been pressed"

"01/03/16 12:11:23 title for cd to remove has been entered"

"01/03/16 12:11:24 cd found and has been removed in the model"

"01/03/16 12:11:26 cd has been removed in the database"

Implement the classes shown (you are allowed to add extra instance variables and/or private methods):

Constructor: Creates a txt file with todays date as file name.

add(logLine) adds a log line to the queue and appends it as one line to the file (and prints out)

Use a Queue
(implementating
QueueADT<LogLine>)

Any class with an option to return a timestamp in the format:
DD/MM/YYYY HH:MM:SS
and a sortable date in the format: YYYY-MM-DD

Log

- file : File

+ Log()
+ add(logLine : LogLine) : void
+ getAll() : QueueADT<LogLine>

0..*

LogLine

- text : String

+ LogLine(text : String)
+ getLogText() : String
+ getDateTime() : DateTime
+ getTimestamp() : String
+ toString() : String

DateTime

Use e.g. the following class for class `DateTime`:

```java
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;

public class DateTime
{
    private Date date;

    public DateTime()
    {
        date = Calendar.getInstance().getTime(); // store current time
    }
    public String getTimestamp()
    {
        SimpleDateFormat sdf = new SimpleDateFormat(
                                    "dd/MM/yyyy HH:mm:ss");
        return sdf.format(date);
    }
    public String getSortableDate()
    {
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        return sdf.format(date);
    }
}
```

Hint: This is how to write to a file:

```java
private void addToFile(LogLine log)
{
   if (log == null)
   {
      return;
   }
   BufferedWriter out = null;
   try
   {
      out = new BufferedWriter(new FileWriter(logFile, true));
      out.write(log + "\n");
   }
   catch (Exception e)
   {
      e.printStackTrace();
   }
   finally
   {
      try
      {
         out.close();
      }
      catch (Exception e)
      {
         e.printStackTrace();
      }
   }
}
```

Exercise 12.01 – Singleton

Change the Log class from the previous exercise such that it is a Singleton
- Draw a UML Class diagram (in Astah) of the class Log
- Test your singleton from a test program (with a main method).