Comparing Sleep and Lunar Cycles with LunarSleep

Final Project Report
EMP 5117

Prepared for:
Professor Guy-Vincent Jourdan

Prepared by:
Marie-Chantal Ross: 8004624

April 13, 2019

Comparing Sleep and Lunar Cycles with LunarSleep
Prepared for Professor Guy-Vincent Jourdan

A program was built to see if there is any correlation between nightly Fitbit sleep cycles and moon phases. The initial intent was to categorize sleep into eight moon phases, but only four moon phase databases were freely available for download. The system requirements are mostly met; the the ability to read in a json formatted file, and the graphical user interface (GUI) for the output being the two that are yet to be satisfied.

Functionality:

LunarSleep calculates the standard standard deviation for each sleep cycle across four moon phases: First Quarter, Full Moon, Last Quarter and New Moon. It checks to see if the percent differences for the transition between sleep cycles is larger than a preset variable (variationCheck). variationCheck is set to 10%, but can be changed if necessary. If the percent difference is less than variationCheck, the code will print: "Sorry, not change in sleep patterns noticed."

If the difference is greater than variationCheck, then the code calls the SleepObject class and creates a series of sleep objects containing arrays of calculated Deep, Wake, Light, and REM sleep cycles for each of the moon phases using different calculation methods: Average, Root Mean Square, Natural Log and raised to the Power of e. Each of these objects pairs up with another object that calculates and stores the percent difference of Deep, Wake, Light and REM from one phase to the next.

Finally, once all the differences are calculated, the code will compare the percent difference for the sleep cycles across the different calculation methods, and tell the user which calculation method produces the larges differences, and between which phases these changes are the largest.

Functionality versus Requirements:

The following table shows how the requirements were met, or not met. The full requirements document can be found in Annex A.

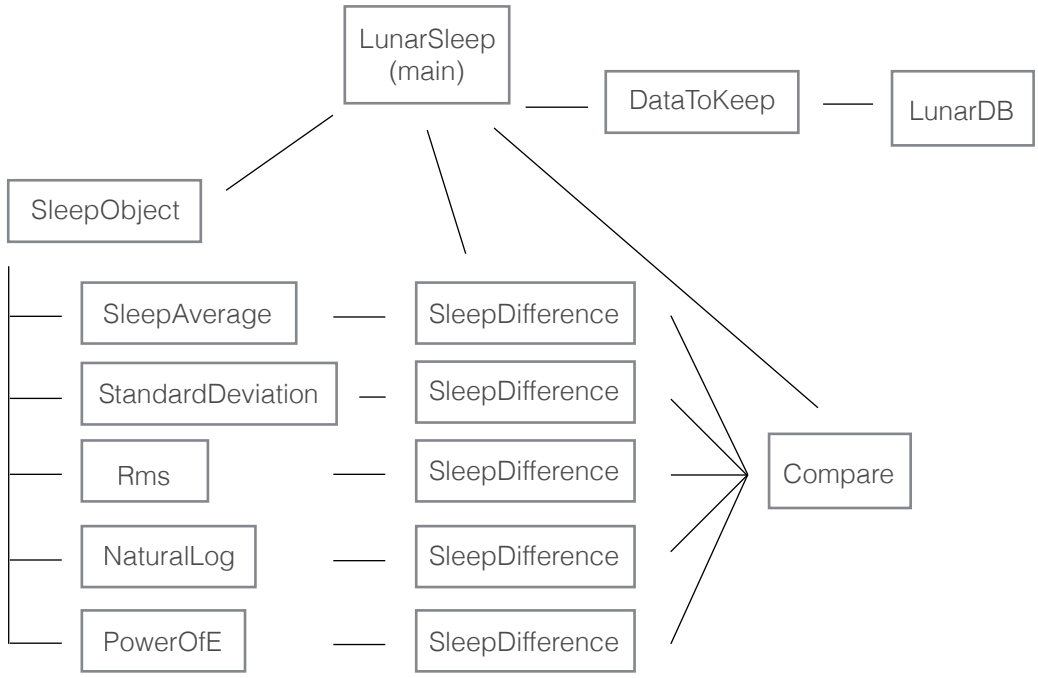
Table 1: Check functionality against requirements

Requirement	Met: Yes/No	Notes
Have a built in time stamp based on moon phases.	Yes & No	
• Match eight moon phase database created from json files downloaded from Nasa starting in March 2017 and ending in February 2019 (note that data is provided in primary phases (quarter waxing, full, quarter waning, new))		A database was created with dates that begin in March of 2017 and go on until March 2037, but for only 4 moon phases based on data availablity.

Requirement	Met: Yes/No	Notes
Accept a dataset of unspecified length	Yes & No	
• Fitbit allows the download of all information stored in json format. The length of the data string is unknown, so the software must be able to accept any amount of data.		I did not have the ability to create a json_parser, which is necessary for this functionality. So I "hired" a sub-contractor to create the json parser. I commented it out because it is not mine, and should not be evaluated as such. Activiting the parser requires that the program be run using javac -cp .:json-simple-1.1.1.jar LunarSleep.java java -cp .:json-simple-1.1.1.jar LunarSleep However, the way the code works for my ability (putting the data directly in the main) is sloppy. I should also note that I attempted to create a file reader to read in a text file, but my knowledge of Exceptions is too limited.
Parse out relevant data	Yes	
• In this case sets of continuous data streams with time spent in REM, Deep, Light and Awake phases (note there are nights where only Sleep and Restless are recorded, or not data is available)		
Match sleep dates to moon phases	Yes	
Assess data based on percent sleep per phase, and if no changes, assess based on logarithmic scale (both base 10 and base e). If no changes, assume no correlation.	Yes & No	
• Report sleep averages per sleep cycles in scatterplot of bar chart • Report % deviation if applicable		Rather than calculate based on percent error, all calculations are done immediately. Though only the maximum is stored, all data is saved for analysis should that be desired. The UI functionality was not done as it exceeded my abilities at this time.

Program Structure:

The structure of the program is as follows:



LunarSleep: Reads in the json file for sleep data and then calls the DataToKeepClass, SleepObject and SleepDifference in that order.

DataToKeep: Compares the fitbit sleep data to a database (LunarDB) of moon phases per date.

SleepObject: Parent object that stores sleep data from children, and contains method for printing that data.

Child Objects (SleepAverage, StandardDeviation, Rms, NaturalLog, PowerOfE): Use different calculation methods to assess the quantity of Deep, Wake, Light and REM sleep cycles per moon phase.

SleepDifference: Calculates the difference in sleep cycles between each of the moon phases and calculates the largest difference in between.

Compare: Takes the SleepDifference values and compares them to the calculation method used to find the largest difference.

System Improvements:

The following improvements are necessary to make this a viable program for true analysis of sleep patterns.

- Create a proper user interface:
    - Properly read in json files of unspecified length
    - Provide a window of results that are meaningful to the user, and allows all sleep object data to be assessed for all calculation methods performed .
- Improve the calculation methods and impose proper data analysis techniques such as regression analysis.
- Ensure proper exceptions are introduced so that program will not crash if formats of data being input do not meet the codes ability to calculate.

ANNEX A - Product Spec

Product:

The final product will be a tool that can be used by anyone looking to analyze if there are differences in sleep patterns in the four main sleep categories (Deep, Light, REM and awake) as the moon shifts between the 8 accepted phases as listed in Figure 1.

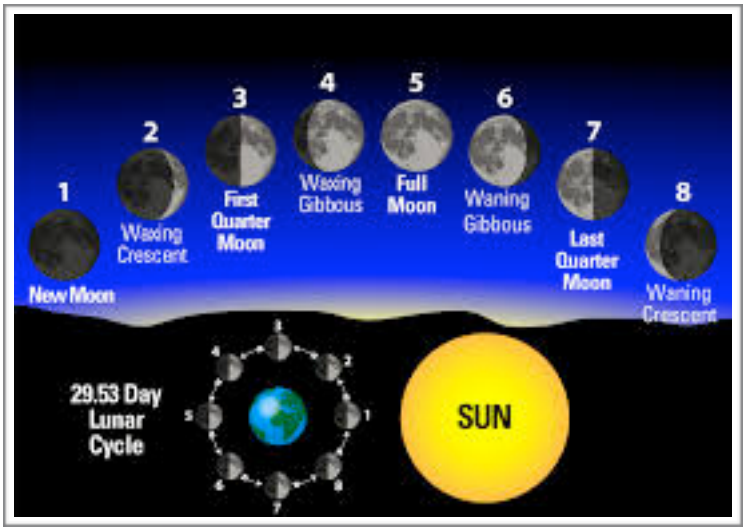


Figure 1: Description of the lunar cycle (image from www.farmersalmanac.com)

The tool will look at the nightly percentage of each sleep stage and categorize these per moon phase. The output will be a visual representation of each sleep stage either in a bar or scatter plot with the lunar cycle along the x-axis and the sleep stages represented by colour.

System must:

1. Have a built in time stamp based on moon phases.
    - Match eight moon phase database created from json files downloaded from Nasa starting in March 2017 and ending in February 2019 (note that data is provided in primary phases (quarter waxing, full, quarter waning, new))
2. Accept a dataset of unspecified length
    - Fitbit allows the download of all information stored in json format. The length of the data string is unknown, so the software must be able to accept any amount of data.
3. Parse out relevant data
    - In this case sets of continuous data streams with time spent in REM, Deep, Light and Awake phases (note there are nights where only Sleep and Restless are recorded, or not data is available)
4. Match sleep dates to moon phases
5. Assess data based on percent sleep per phase, and if no changes, assess based on logarithmic scale (both base 10 and base e). If no changes, assume no correlation.
    1. Report sleep averages per sleep cycles in scatterplot of bar chart
    2. Report % deviation if applicable