# CSE 1322L - Lab 8

## Introduction

In this lab, you will write a simple calculator to determine the difference, in seconds, between two timestamps that the user will enter.

The program should prompt the user for two timestamps in military time, which has the following format:

```
HH:MM:SS
```

The hour component must be a number between 0 and 23, and both the minute and the second components must be numbers between 0 and 59. These numbers are all inclusive. Your program must be able to recover from the user entering invalid information.

## Requirements

The features described below must be in your program:

- A class called InvalidTimeException, which is a subclass of Exception
  - **InvalidTimeException(String)**: passes the argument to the super class constructor
- In your driver class, create the following methods:
  - **static int differenceInSeconds(String, String)**: Converts the arguments from military time into seconds, and then returns the difference between the second one and the first one. <u>You can always assume the first argument is the start time and the second the end time</u>. This method must be able to throw InvalidTimeExceptions. The conditions under which the exception is thrown, as well as their associated messages, can be found on the table below:

| Condition | Message |
|---|---|
| Input does not have hours and minutes and seconds components | "Timestamp must be in format HH:MM:SS" |
| The hour component of the timestamp is out of range | "Hours must be greater than or equal to 0" **OR** "Hours must be less than 24" |
| The minute component of the timestamp is out of range | "Minutes must be greater than or equal to 0" **OR** "Minutes must be less than 60" |
| The second component of the timestamp is out of range | "Seconds must be greater than or equal to 0" **OR** "Seconds must be less than 60" |

        - **Note: differenceInSeconds() must not have any TRY-CATCH blocks**. It must also not read any input from the user: the user's input must be supplied through the parameters.
        - **Note: The conditions on the table above must be checked in the order they appear in the table**. A timestamp of "27:59:200" would throw an exception with the message "Hours must be less than 24", even though the number of seconds is also incorrect.
    - **static void main(String[])**: Implements the menu options below:
        - **Calculate difference in seconds**: prompts the user for two timestamps, then passes them to differenceInSeconds(), printing out the results. You can assume the first timestamp always comes before the second one.

          **Note that this menu option must be able to recover from either the user entering non-numbers or from the user entering invalid numbers**. The recovering must be as follows:

            - If the user enters non-numbers (e.g.: 'e' instead of 3), print "You must enter integers for the hours, minutes, and seconds". You must catch the exact exception type thrown (i.e.: catching a general Exception will lose you points)
            - If the user enters an invalid time (e.g.: "27:00"), print out the message in the exception
        - **Exit**: terminates the program
    -

## Deliverables

- InvalidTimeException.java
- Lab8.java (driver)
    - differenceInSeconds()
    - main()

## Considerations

- The easiest way to figure out if the timestamp is in the correct format is by using the String's split() method. If the method's output doesn't have the correct length, we know that the user either entered too little or too much information. You can find the documentation for said method here:
  https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html#split(java.lang.String)
- It bears repeating: there should be no TRY-CATCH blocks inside of differenceInSeconds(). The method is simply concerned with converting the arguments into integers, taking their difference, and returning the result. If the method runs into any exceptions, those must be thrown back to main().

- When trying to convert a timestamp's components to a number, the conversion might fail because the user entered a non-number. Your main() (not differenceInSeconds()) must be able to recover from this by catching the exact exception type that gets thrown. You can find the exact type of exception thrown when Integer.parseInt() or Scanner.nextInt() fails in the documentations below:

  https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/lang/Integer.html#parseInt(java.lang.String)

  https://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html#nextInt--

- The easiest way to calculate the difference between the timestamps is by converting both of them to seconds, and then taking the difference between them. Recall that each hour has 3600 seconds, and each minute has 60 seconds.

## Sample Output (user input in red)

```
[Time Calculator]
1. Calculate difference in seconds
2. Exit
Enter your option: 1

Enter the start timestamp: 23:59:12
Enter the end timestamp: 23:70:12
Minutes must be less than 60

1. Calculate difference in seconds
2. Exit
Enter your option: 1

Enter the start timestamp: 23:00:-12
Enter the end timestamp: 27:55:00
Hours must be less than 24

1. Calculate difference in seconds
2. Exit
Enter your option: 1

Enter the start timestamp: 12:-12:12
Enter the end timestamp: 12:00:12
Minutes must be greater than or equal to 0

1. Calculate difference in seconds
2. Exit
Enter your option: 1
Enter the start timestamp: 7:15:15
Enter the end timestamp: 9:89
Timestamp must be in the format HH:MM:SS

1. Calculate difference in seconds
2. Exit
Enter your option: 1

Enter the start timestamp: 3:35:15
Enter the end timestamp: 17:05:00
The difference between 3:35:15 and 17:05:00 is 48585 seconds

1. Calculate difference in seconds
2. Exit
Enter your option: 2

Shutting off...
```