CSCE 145: Algorithmic Design I

Homework - 6

Objective: Design a Java project to convert 24-hour time(military clock) into 12-hour time using the following Java classes:

- TimeConverter
- ClockTimeDemo
- TimeException

Requirements:

- Functionality (85 pts)
  - No Syntax, Major Run-Time, or Major Logic Errors. (85 pts*)
    - *Code that cannot be compiled due to syntax errors is non-functional code and will receive no points for this entire section.
    - *Code that cannot be executed or tested due to major run-time or logic errors is nonfunctional code and will receive no points for this entire section.
    - *You are NOT allowed to use any built-in packages/parsers/methods including but not limited to java.time, java.chrono, Clock, LocalTime, LocalDateTime, OffsetTime, ZonedDateTime, DateTimeFormatter, DateTimeParseException and the like.
    - Usage of any of the above listed built-in packages is strictly prohibited.
    - You may use String methods such as substring, indexOf, charAt
    - You may use Wrapper classes for parsing such as Integer.parseInt()
  - Clear and Easy-To-Use Interface (5 pts)
    - Users should easily understand what the program does and how to use it.
    - Users should be prompted for input and should be able to enter data easily.
    - Users should be presented with output after major functions, operations, or calculations.
    - All the above must apply for full credit
  - Create a class called **TimeException** that inherits from Exception (10pts)

- This class DOES NOT contain the main method
- A default constructor- calls the parent's constructor and prints the message "EXCEPTION: Invalid time entered"
- A parameterized constructor that takes in a parameter of type String
  - This constructor passes the message to the parent's constructor

◦ Create another class: **TimeConverter**

- This class DOES NOT contain the main method
- It has 4 instance variables namely: (4pts)
  - hours
  - minutes
  - seconds
  - notPM → a boolean to check if it is AM or PM
- Constructors (6pts)
  - Default - sets the values of all instance variables to default
    - default value for hours is 0 (assume that the 24th hour is 0)
    - default value for minutes is 0
    - default value for seconds is 0
    - default value for notPM is true
  - Parameterized Constructor that takes in 3 parameters for hours, minutes and seconds
- Methods
  - Accessors for all instance variables (10 pts)
    - Create accessors for each instance variable
    - Returns the value of the current instance
  - Mutators for all instance variables (10 pts)
    - Create mutators for each instance variable and check for valid values
      - For example: Minutes can only have a value between 0-59 (both inclusive)

- updateTime (10pts)
  - This method takes in 3 parameters – hours, minutes and seconds
  - For valid values:
    - Converts only valid values of hours, minutes and seconds into a 12-hour time
    - Converts AM to PM and vice-versa
    - assigns the new values of the 12-hour clock to the instance variables
  - For invalid values: throws the TimeException
- updateTime (20pts)
  - creates an overloaded method
  - This method takes in a parameter of type String in the following format
    - hours:minutes:seconds
    - First, separate the value of hours, minutes and seconds in the entered string
      - For example: if the string has a value of 20:59:05
        - the value of hours is 20
        - the value of minutes is 59
        - the value of seconds is 5
      - Hint: use substrings and wrapper classes to handle the values of hours, minutes and seconds.
      - Check for the validity of hours, minutes and seconds just like the other updateTime() method
      - While parsing the string parameter, if the user provides a non-numeric value such as letters instead of time, you need to handle the general Exception using a try-catch.
- displayTime (5pts)
  - This method prints the time using a 12-hour clock
    - Format: <<hours>>**:**<<minutes>>**:**<<seconds>> <<time of day>>

        - For example: 8:59:05 PM
    - Create a test class: **ClockTimeDemo** (15 pts)
        - This class contains a main method
        1. Create the first object of the type TimeConverter
            - Prompt the user to provide the hours, minutes and seconds using the military clock(24-hour)
            - Convert the time into a 12-hour clock and display the results on the console
        2. Create the second object of the type TimeConverter
            - Prompt the user to provide time using the 24-hour clock in the format "hours:minutes:seconds"
            - Convert the time entered into a 12-hour clock and display the results on the console
        - Ask the user if they want to do this again. If yes, repeat steps 1 and 2 *(See Example section)*
- Coding Style (9 points)
    - Readable Code
        - Meaningful identifiers for data and methods.
        - Proper indentation that clearly identifies statements within the body of a class, a method, a branching statement, a loop statement, etc.
        - All the above must apply for full credit.
- Comments (6 pts)
    - Your name at the beginning of the file as a single-line comment. (1 pt)
    - At least 5 meaningful comments in addition to your name. These must describe the function of the code it is near. (5 pts)

Example: Convert military time into 12-hour clock time!!! Enter the hours on the military clock:

21

Enter the minutes on the military clock:

30

Enter the seconds on the military clock:

14

12-hour clock time → 9:30:14 PM

Enter 24 hour clock time in the format “hours:minutes:seconds”

15:15:15

12-hour clock time → 3:15:15 PM

Would you like to do this again? Enter “Yes” or “No”:

Yes

Enter the hours on the military clock:

27

Enter the minutes on the military clock:

60

Enter the seconds on the military clock:

-9

EXCEPTION: Invalid time entered!

Enter 24 hour clock time in the format “hours: minutes:seconds”

10:59:59

12-hour clock time → 10:59:59 AM

Would you like to do this again? Enter “Yes” or “No”: *No* Exiting the program!

Submission:

- Submit all .java files on Dropbox