
Due Date:	By 11:55pm Wednesday May 22, 2019
Evaluation:	4% of final mark (see marking rubric at the end of handout)
Late Submission:	none accepted
Purpose:	The purpose of this assignment is to help you learn Java identifiers, assignments, input/output and if and if/else statements.
CEAB/CIPS Attributes:	Design/Problem analysis/Communication Skills

General Guidelines When Writing Programs:

Include the following comments at the top of your source codes

```
// -----  
// Assignment (include number)  
// Written by: (include your name and student id)  
// For COMP 248 Section (your section) - Summer 2019  
// -----
```

- In a comment, give a general explanation of what your program does. As the programming questions get more complex, the explanations will get lengthier.
- Include comments in your program describing the main steps in your program. Focus in your comments rather on the why than the how.
- Display a welcome message.
- Display clear prompts for users when you are expecting the user to enter data from the keyboard.
- All output should be displayed with clear messages and in an easy to read format.
- End your program with a closing message so that the user knows that the program has ended.

Question 1 - Display a message (2 pts)

Write a complete Java program that displays the following output exactly as in the sample output in Figure 1 below. You are to use a single `System.out.println()` statement to output the message. There is no need to declare any variables for this question and this question does not require a header before the message

```
Welcome to my First Java program!  
The statement System.out.println("Hello"); will display  
Hello then move the cursor to the next line while the statement  
System.out.print("Hello"); will display  
Hello and the cursor will stay on the same line.  
  
All Done!
```

Figure 1 - Sample output screen for Question 1

Question 2 – String variables and use of String methods. (4 points)

Write a complete Java program which prompts the user for a string with 3 words each separated by a single space and reads the line into one String variable using `nextline()`. Extract each word and add them to a new String variable in reverse order with a space between the words. Have the 1st letter of the new string in upper case and all other strings in lower case letter. Display the new string.

Figure 2 is a sample output to illustrate the expected behaviour of your program. Your program should work for any input, not just the one in the sample below. The text in **green** is user input

```
\\-----  
\\ String Splitter Program  
\\-----  
  
Enter 3 words separated by a blank:  
Programming orientED OBJECT  
  
Words in reverse order:  
Object oriented programming  
  
All done!!!
```

Figure 2 - Sample output for Question 2

Restrictions: The use of loops of any kind is not allowed for this question.

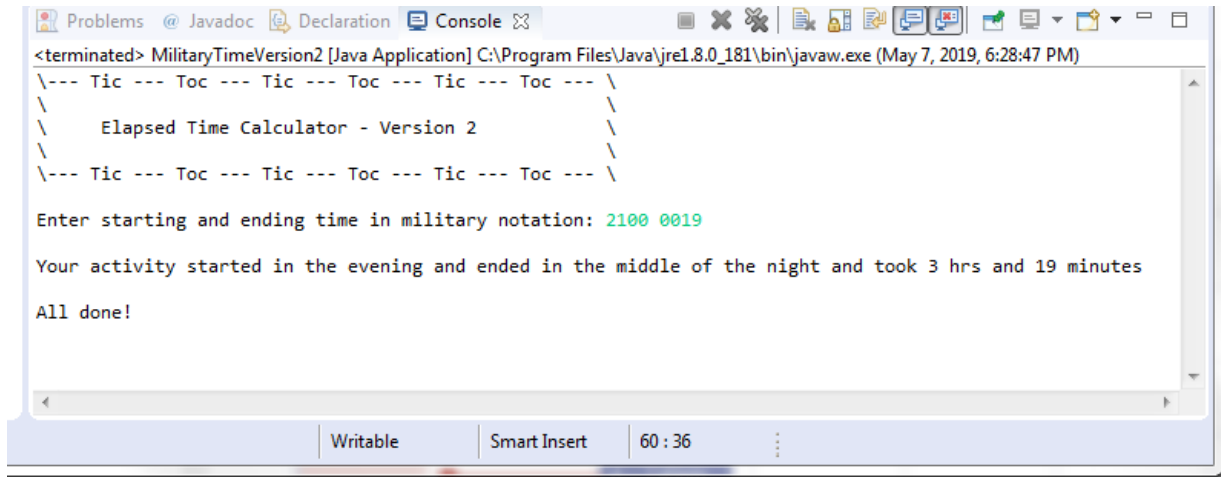
Question 3 - User input & selection (8 points)

- a) You are given a Java program (without comments and non descriptive variable names) which reads two integer times in military format (0900, 1750) and prints the number of hours and minutes between the two times. The program in file *MilitaryTime.java* assumes that the times are valid times and that the first time entered (starting time) is less than the second time (ending time). Your task is to understand what this program does, to change the variable names to names that will indicate what they contain, to prompt the user for inputs, to add comments to the program and to indent it properly. You are not to change the logic of the code.
- b) Copy your modified program from part a) of this question into a second file and modify the second program so that:
 - If the starting time is less than or equal the ending time, it will calculate the elapsed time and display it. (This is the code from part a) of this questions)
 - If the starting time is greater than the ending time it will perform the required calculation. Note this means that the event is starting on one day and ended on the next day. For example, if the event started at 2355 and ended at 0001, then the elapsed time is 0 hours and 6 minutes.
 - In both cases there should also be a message indicating what period of the day the event started and ended. Here is the classification to use:

- if the time is before 0600, then it is the middle of the night
- if the time is ≥ 6000 and less than 1200 it is in the morning
- if the time is ≥ 1200 and less than 1800 it is during the day
- if the time is ≥ 1800 then it is in the evening.
- It ends by displaying a closing message.

You can assume a perfect user who will enter valid times in military format.

Following are a couple of screen shots to illustrate the expected behavior of your program.



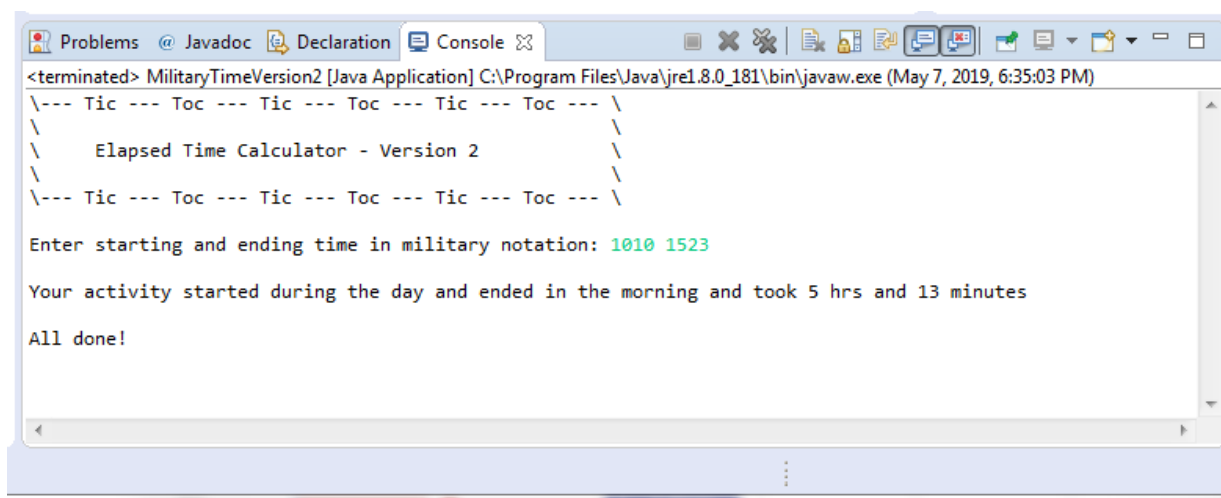
```
<terminated> MilitaryTimeVersion2 [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (May 7, 2019, 6:28:47 PM)
\--- Tic --- Toc --- Tic --- Toc --- Tic --- Toc --- \
\
\   Elapsed Time Calculator - Version 2   \
\
\--- Tic --- Toc --- Tic --- Toc --- Tic --- Toc --- \

Enter starting and ending time in military notation: 2100 0019

Your activity started in the evening and ended in the middle of the night and took 3 hrs and 19 minutes

All done!
```

Figure 3- Sample output where starting time is **greater** than ending time



```
<terminated> MilitaryTimeVersion2 [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (May 7, 2019, 6:35:03 PM)
\--- Tic --- Toc --- Tic --- Toc --- Tic --- Toc --- \
\
\   Elapsed Time Calculator - Version 2   \
\
\--- Tic --- Toc --- Tic --- Toc --- Tic --- Toc --- \

Enter starting and ending time in military notation: 1010 1523

Your activity started during the day and ended in the morning and took 5 hrs and 13 minutes

All done!
```

Figure 4- Sample output where starting time is **less** than ending time

Submitting Assignment 1

1. Compress the source codes (the .java file only please) of this assignment together into one file following the following naming convention:

The zip file should be called *a#_studentID*, where # is the number of the assignment and *studentID* is your student ID number. For example, for the first assignment, student 123456 would submit a zip file named a1_123456.zip.

You will have 4 .java files to submit for this assignment

2. **Upload** your compressed file using the appropriate assignment link in the Assessment page.
3. **NOTE:** The only compressed file format we accept is .ZIP.
No .RAR files are accepted.

Evaluation Criteria for Assignment 1 (20 points)

Source Code	
Comments for all 3 questions (3 pts.)	
Description of the program (authors, date, purpose)	1 pts.
Description of variables and constants	1 pt.
Description of the algorithm	1 pts.
Programming Style for all 3 questions (3 pts.)	
Use of significant names for identifiers	1 pt.
Indentation and readability	1 pt.
Welcome Banner or message/Closing message	1 pt.
Question 1 (2 pts.)	
Display text using a SINGLE System.out.println()	2 pts.
Question 2 (4 pts.)	
Read text into one variable of type String	1 pt.
Extract the 3 words	1.5 pts.
Display in reverse order in requested format	1.5 pts.
Question 3 (8 pts.)	
Part a) 3 pts	3 pts.
Part b) 5 pts	
Calculate elapsed time when starting time > ending time	2 pts.
Display period of the day for starting & ending time	3 pts.
TOTAL	20 pts.