Lab Exercises

Lab Exercise 1 — Time: Part 1

Name: _____ **Date:** _____

Section: _____

The following problem is intended to be solved in a closed-lab session with a teaching assistant or instructor present. The problem is divided into six parts:

1. Lab Objectives
2. Problem Description
3. Sample Output
4. Program Template (Fig. L 8.3–Fig. L 8.4)
5. Problem-Solving Tips
6. Follow-Up Questions and Activities

The program template represents a complete working Java program with one or more key lines of code replaced with comments. Read the problem description and examine the output, then study the template code. Using the problem-solving tips as a guide, replace the `/* */` comments with Java code. Compile and execute the program. Compare your output with the sample output provided. Then answer the follow-up questions. The source code for the template is available at `www.pearsonhighered.com/deitel`.

Lab Objectives

This lab was designed to reinforce programming concepts from Chapter 8 of *Java How To Program: 8/e*. In this lab, you will practice:

- Modifying methods of a class.
- Accessing instance variables.
- Using *set* and *get* methods.

The follow-up questions and activities also will give you practice:

- Understanding the difference between access specifiers `public` and `private`.

Problem Description

Modify the *set* methods in class `Time2` of Fig. L 8.1 to return appropriate error values if an attempt is made to set one of the instance variables `hour`, `minute` or `second` of an object of class `Time` to an invalid value. [*Hint:* Use `boolean` return types on each method.] Write a program that tests these new *set* methods and outputs error messages when incorrect values are supplied.

Lab Exercises

Name:

Lab Exercise 1 — Time: Part 1

Sample Output

```
1. Set Hour
2. Set Minute
3. Set Second
4. Add 1 second
5. Exit
Choice: 1
Enter Hours: 10
Hour: 10  Minute: 0  Second: 0
Universal time: 10:00:00   Standard time: 10:00:00 AM
1. Set Hour
2. Set Minute
3. Set Second
4. Add 1 second
5. Exit
Choice: 2
Enter Minutes: 10
Hour: 10  Minute: 10  Second: 0
Universal time: 10:10:00   Standard time: 10:10:00 AM
1. Set Hour
2. Set Minute
3. Set Second
4. Add 1 second
5. Exit
Choice: 3
Enter Seconds: 10
Hour: 10  Minute: 10  Second: 10
Universal time: 10:10:10   Standard time: 10:10:10 AM
1. Set Hour
2. Set Minute
3. Set Second
4. Add 1 second
5. Exit
Choice: 3
Enter Seconds: 99
Invalid seconds.
Hour: 10  Minute: 10  Second: 0
Universal time: 10:10:00   Standard time: 10:10:00 AM
1. Set Hour
2. Set Minute
3. Set Second
4. Add 1 second
5. Exit
Choice: 5
```

Lab Exercises

Name:

Lab Exercise 1 — Time: Part 1

Template

```
1  // Lab 1: Time2.java
2  // Time2 class definition with methods tick,
3  // incrementMinute and incrementHour.
4
5  public class Time2
6  {
7     private int hour; // 0 - 23
8     private int minute; // 0 - 59
9     private int second; // 0 - 59
10
11    // Time2 no-argument constructor: initializes each instance variable
12    // to zero; ensures that Time2 objects start in a consistent state
13    public Time2()
14    {
15       this( 0, 0, 0 ); // invoke Time2 constructor with three arguments
16    } // end Time2 no-argument constructor
17
18    // Time2 constructor: hour supplied, minute and second defaulted to 0
19    public Time2( int h )
20    {
21       this( h, 0, 0 ); // invoke Time2 constructor with three arguments
22    } // end Time2 one-argument constructor
23
24    // Time2 constructor: hour and minute supplied, second defaulted to 0
25    public Time2( int h, int m )
26    {
27       this( h, m, 0 ); // invoke Time2 constructor with three arguments
28    } // end Time2 two-argument constructor
29
30    // Time2 constructor: hour, minute and second supplied
31    public Time2( int h, int m, int s )
32    {
33       setTime( h, m, s ); // invoke setTime to validate time
34    } // end Time2 three-argument constructor
35
36    // Time2 constructor: another Time2 object supplied
37    public Time2( Time2 time )
38    {
39       // invoke Time2 constructor with three arguments
40       this( time.getHour(), time.getMinute(), time.getSecond() );
41    } // end Time2 constructor with Time2 argument
42
43    // Set a new time value using universal time. Perform
44    // validity checks on data. Set invalid values to zero.
45    /* Write header for setTime. */
46    {
47       /* Write code here that declares three boolean variables which are
48          initialized to the return values of setHour, setMinute and setSecond.
49          These lines of code should also set the three member variables. */
50
51       /* Return true if all three variables are true; otherwise, return false. */
52    }
53
```

Fig. L 8.3 | `Time2.java`. (Part 1 of 3.)

Lab Exercises

Name:

Lab Exercise 1 — Time: Part 1

```
54      // validate and set hour
55      /* Write header for the setHour method. */
56      {
57         /* Write code here that determines whether the hour is valid.
58            If so, set the hour and return true. */
59
60         /* If the hour is not valid, set the hour to 0 and return false. */
61      }
62
63      // validate and set minute
64      /* Write the header for the setMinute method. */
65      {
66         /* Write code here that determines whether the minute is valid.
67            If so, set the minute and return true. */
68
69         /* If the minute is not valid, set the minute to 0 and return false. */
70      }
71
72      // validate and set second
73      /* Write the header for the setSecond method. */
74      {
75         /* Write code here that determines whether the second is valid.
76            If so, set the second and return true. */
77
78         /* If the second is not valid, set the second to 0 and return false. */
79      }
80
81      // Get Methods
82      // get hour value
83      public int getHour()
84      {
85         return hour;
86      } // end method getHour
87
88      // get minute value
89      public int getMinute()
90      {
91         return minute;
92      } // end method getMinute
93
94      // get second value
95      public int getSecond()
96      {
97         return second;
98      } // end method getSecond
99
100     // Tick the time by one second
101     public void tick()
102     {
103        setSecond( second + 1 );
104
105        if ( second == 0 )
106           incrementMinute();
107     } // end method tick
108
```

Fig. L 8.3 | `Time2.java`. (Part 2 of 3.)

Lab Exercises

Name:

Lab Exercise 1 — Time: Part 1

```
109    // Increment the minute
110    public void incrementMinute()
111    {
112       setMinute( minute + 1 );
113
114       if ( minute == 0 )
115          incrementHour();
116    } // end method incrementMinute
117
118    // Increment the hour
119    public void incrementHour()
120    {
121       setHour( hour + 1 );
122    } // end method incrementHour
123
124    // convert to String in universal-time format (HH:MM:SS)
125    public String toUniversalString()
126    {
127       return String.format(
128          "%02d:%02d:%02d", getHour(), getMinute(), getSecond() );
129    } // end method toUniversalString
130
131    // convert to String in standard-time format (H:MM:SS AM or PM)
132    public String toString()
133    {
134       return String.format( "%d:%02d:%02d %s",
135          ( ( getHour() == 0 || getHour() == 12 ) ? 12 : getHour() % 12 ),
136          getMinute(), getSecond(), ( getHour() < 12 ? "AM" : "PM" ) );
137    } // end method toStandardString
138 } // end class Time2
```

Fig. L 8.3 | Time2.java. (Part 3 of 3.)

```
1  // Lab 1: Time2Test.java
2  // Program adds validation to Fig. 8.7 example
3  import java.util.Scanner;
4
5  public class Time2Test
6  {
7     public static void main( String args[] )
8     {
9        Scanner input = new Scanner( System.in );
10
11       Time2 time = new Time2(); // the Time2 object
12
13       int choice = getMenuChoice();
14
15       while ( choice != 5 )
16       {
17          switch ( choice )
18          {
19             case 1: // set hour
20                System.out.print( "Enter Hours: " );
21                int hours = input.nextInt();
22
```

Fig. L 8.4 | Time2Test.java. (Part 1 of 2.)

Lab Exercises

Name:

Lab Exercise 1 — Time: Part 1

```
23                /* Write code here that sets the hour. If the hour is invalid,
24                   display an error message. */
25
26                break;
27             case 2: // set minute
28                System.out.print( "Enter Minutes: " );
29                int minutes = input.nextInt();
30
31                /* Write code here that sets the minute. If the minute is invalid,
32                   display an error message. */
33
34                break;
35             case 3: // set seconds
36                System.out.print( "Enter Seconds: " );
37                int seconds = input.nextInt();
38
39                /* Write code here that sets the second. If the second is invalid,
40                   display an error message. */
41
42                break;
43             case 4: // add 1 second
44                time.tick();
45                break;
46          } // end switch
47
48          System.out.printf( "Hour: %d  Minute: %d  Second: %d\n",
49             time.getHour(), time.getMinute(), time.getSecond() );
50          System.out.printf( "Universal time: %s   Standard time: %s\n",
51             time.toUniversalString(), time.toString() );
52
53          choice = getMenuChoice();
54       } // end while
55    } // end main
56
57    // prints a menu and returns a value corresponding to the menu choice
58    private static int getMenuChoice()
59    {
60       Scanner input = new Scanner( System.in );
61
62       System.out.println( "1. Set Hour" );
63       System.out.println( "2. Set Minute" );
64       System.out.println( "3. Set Second" );
65       System.out.println( "4. Add 1 second" );
66       System.out.println( "5. Exit" );
67       System.out.print( "Choice: " );
68
69       return input.nextInt();
70    } // end method getMenuChoice
71 } // end class Time2Test
```

Fig. L 8.4 | `Time2Test.java`. (Part 2 of 2.)

Problem-Solving Tips

1. Use `boolean` return types for the *set* methods.
2. Each *set* method should `return true` if the value is valid and `false` if it is not.
3. If you have any questions as you proceed, ask your lab instructor for assistance.