Programming with C#

# Decision Structures Practice

1. Open Visual Studio if it is not already open.
2. Locate the folder where you extracted the companion content.
3. In the Chapter 4 folder, locate either the Windows 7 or Windows 8 folder.
4. Open the Selection project.
5. On the Debug menu, click Start Debugging or press F5.
6. Visual Studio will build and run the application. The UI displays two DatePickercontrols that are named firstDateand secondDate. If you are running the Windows 8.1 version then both controls will display the current date.
7. If you are using Windows 7 or Windows 8, click the calendar icon in the first *DatePicker* con­trol, and then click the current date. Repeat this operation for the second *DatePicker* control.
8. Click Compare.
9. The application will display some pre-canned text in the result list at the bottom of the screen.
10. Close the application or in Visual Studio, on the Debug menu click Stop Debugging.
11. Expand the MainWindows.xaml branch in Solution Explorer and then double-click the MainWindow.xaml.cs file to open the file in the Text Editor window.
12. Locate the private void compareClick(object sender, RoutedEventArgs e) method.
13. You can see where the pre-canned text comes from that is displayed, but also notice that this method runs whenever the user clicks the Compare button on the form. The variables first and second hold DateTime values that get populated with the dates displayed in the two date controls on the form.
14. This method passes the two DateTime values in the call to the dateCompare method.
15. dateCompare compares the two dates and return an integer value of 0 if the two dates are the same, a value of –1 if the first date is less than the second, and +1 if the first date is greater than the second.
16. The show method displays the results of the comparison in the info text box control in the lower half of the form.
17. Scroll to the private int dateCompare(DateTime leftHandSide, DateTime rightHandSide) method.
18. Knowing that we want the method to return 0 if the dates are equal, -1 if first is less and +1 if first is greater, replace the //TO DO comment with an if structure to perform the comparison between leftHandSide and rightHandSide for the three cases mentioned here. You should use an if, else if, and else clause in your code.  
    ***NOTE: you should create an int variable named result or something similar that will be used in the return statement.***
19. Your code should include sufficient if else blocks to check the date parts such as year, month, and day. For a hint, see the complete code solution at the end of this document. Try to figure it out before looking at the solution.
20. Run the application after you enter your code and test the results by changing the date values in the first and last date controls.
21. Fix any errors that may occur.
22. If all goes as planned, close the application.
23. Select the File menu in Visual Studio.
24. Select Close Solution.
25. Open the SwitchStatement project from the Chapter 4 folder
26. Select the Debug menu and choose Start Debugging or press the F5 key
27. If you run the application in its current state, when you type text into the upper text box and click the Copy button, the text will be copied, verbatim, to the lower text box.
28. Stop the application and go back to the Visual Studio IDE
29. In Solution Explorer, expand MainWindow.xaml and then double-click MainWindow.cs to open the C# code file in the text editor window.
30. Scroll to the copyClick() method and note that this method takes the text upper text box, which is named source in the code. The for loop will iterate over all the characters in the text box one at a time. As it does so, it calls copyOne for each character in the string.
31. copyOne() accepts an individual character and contains a single switch statement that has a default: case. You will modify this code but note that the default case simply appends each character to the end of the previous character.
32. In order to work with individual case statements, you will modify this code to perform comparisons on XML tags that may be present in the text from the source text box.
33. Many times, you may need to convert XML or HTML tags to a different value for rendering in different engines. The examples used here are converting the ‘**<**’ symbol to “**&lt;**”.
34. Using the default case code as an example for appending text, create a case statement for each of these characters and their conversions:

|  |  |
| --- | --- |
| < | &lt; |
| > | &gt; |
| & | &amp; |
| \” | &#34; |
| \’ | &#39; |

1. Ensure you surround the converted symbol with double quotes as it is a string value.
2. Run the application again but this time, enter the following text in the upper text box, **inRange = (lo <= number) && (hi >= number);**
3. Click the Copy button. Your output in the lower text box should be, **inRange = (lo &lt;= number) &amp;&amp; (hi &gt;= number);**
4. If not, go back to your code and fix any errors.
5. You can also experiment with adding other case statements for more HTML or XML tags.
6. Stop the application.
7. You can leave Visual Studio open for the next practice session or close it down if you are finished for now.

**Code Solution for if statements**

private int dateCompare(DateTime leftHandSide, DateTime rightHandSide)

{

int result;

if (leftHandSide.Year < rightHandSide.Year)

{

result = -1;

}

else if (leftHandSide.Year > rightHandSide.Year)

{

result = 1;

}

else if (leftHandSide.Month < rightHandSide.Month)

{

result = -1;

}

else if (leftHandSide.Month > rightHandSide.Month)

{

result = 1;

}

else if (leftHandSide.Day < rightHandSide.Day)

{

result = -1;

}

else if (leftHandSide.DayOfWeek > rightHandSide.DayOfWeek)

{

result = 1;

}

else

{

result = 0;

}

return result;

}

**Code Solution for switch Statement**

private void copyOne(char current)

{

switch (current)

{

case '<':

target.Text += "&lt;";

break;

case '>':

target.Text += "&gt;";

break;

case '&':

target.Text += "&amp;";

break;

case '\"':

target.Text += "&#34;";

break;

case '\'':

target.Text += "&#39;";

break;

default:

target.Text += current;

break;

}

}