School of Applied Sciences And Technology

Department of IST

Program: CNT

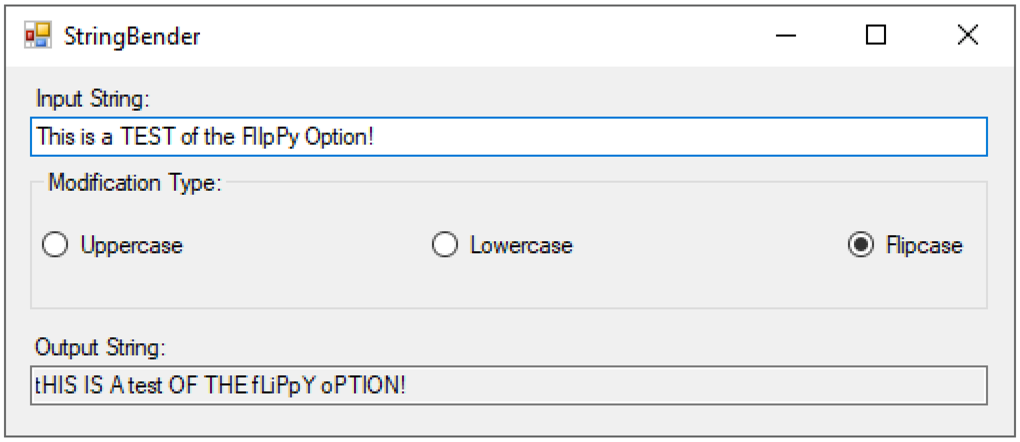
CMPE1666 – ICA 12-StringBender (Early Delegates)

In this ICA, you will modify a string entered by the user in various ways. Each manipulation method will be a separate method in your main form.

The user will be able to alter the modification type with a radio group.

Ensure that the tab order is: input box, followed by the 3 radio buttons in the order they appear on the UI.

The application will have the following appearance:



Changing the input string or the modification type will immediately cause the output string to update.

You will use event consolidation on the radio buttons.

Ultimately, you will use a `delegate` to call the correct modification method, based on the radio button choice. Before you get to that, you should write your code incrementally.

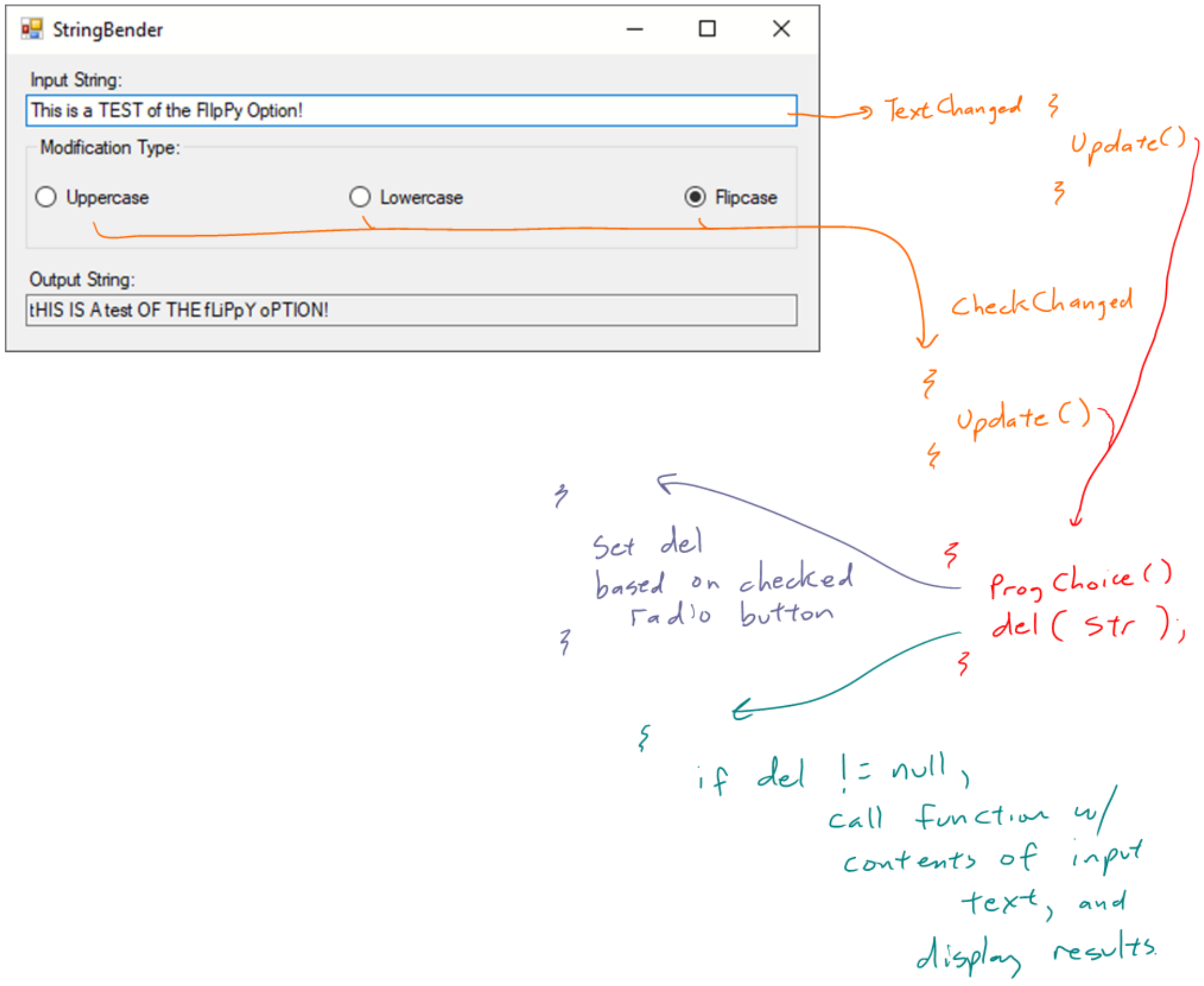
- Create and test methods that take a `string` and return a modified `string`, for each of the 'Uppercase', 'Lowercase', and 'Flipcase' methods. You can use the output window to validate that these functions are working before you write any other code.
- Consolidate the radio button `CheckChanged` events into a single handler. Alter the code so that the input text is manipulated by the selected choice and shown in the output.
- Use method consolidation so that a change in input text or a change in radio button call the same function. This function will take the input text and call the appropriate manipulation

function based on the currently selected radio button. Put the result in the output textbox. At this point, your program should completely function.

Now for the highly contrived part. As you are aware, delegates are like variables that hold method entry points. Since we have three different, but functionally similar methods, we may use a delegate as a variable to hold the manipulation method entry point. When a radio button changes, the delegate may be set to hold the selected modification method.

- Create a `delegate` type that matches the signature for the modification methods. Create an instance of this delegate but mark it `null` initially. Do these definitions at class level.
- Modify your update function so that it calls a method to program the delegate instance, based on the current radio selection. Then call the delegate, passing in the input string. The return value should be put into the output field.

The following diagram shows the flow of your code:



Notes:

- The `string` type contains functions to perform some useful case manipulations.
- The `char` type contains static methods to perform some useful character classifications. Flipping case is not a single method call. You will need to iterate over the source string and build the output string.
- You will be writing a bunch of methods to ensure that there is not duplicate code (except function calls, maybe).
- Your delegate will be started as `null`, you will therefore be required to perform a `null` check before you attempt to invoke a method through that delegate.

Rubric- Max Marks: 30

This application will require visual inspection of functionality and code.

Mark loss is at your instructor's discretion but will be applied consistently across all students.

Item	Marks	Penalties
UI Design (30%)		
• UI is as directed.	3	
• Tab Order is Correct.	2	
• Effective use of space	2	
• Radio buttons have default selection	2	
Code Design and Implementation (50%)		
• Appropriate consolidation for events and support methods.	2	
• All button functions are as described.	2	
• Appropriate delegate declaration and usage.	5	
• All operations on all events work as expected.	3	
• Use separate methods for the string manipulations (not placed code directly in event listener)	3	
Documentation (20%): • Programmer Block • Well commented code • Appropriate Variable Names • Proper spacing between blocks of code • Control names are consistent and appropriate.	6	