

KidCoder™: Windows Programming

Second Edition Errata Sheet

Updated July 11, 2012

This document lists the known typographical or other corrections to the *KidCoder™: Windows Programming* Second Edition course.

- On page 55-56 the “Integer Types” paragraph incorrectly lists the ranges of the **Integer**, **Long**, and **Short** data types. The correct ranges for these three whole number data types (plus a few others) are listed below.

Data Type	Possible Values
Byte	0 through 255
SByte	-128 through 127
Short	-32,768 through 32,767
Integer	-2,147,483,648 through 2,147,483,647
Long	-9,223,372,036,854,775,808 through 9,223,372,036,854,775,807

- On page 96 the example **SubtractButton_Click** method has a typo. The “+” sign should be a “-” sign. The corrected example is shown below. The code included in the “Chapter Sample Programs” has the correct sign.

```

Private Sub SubtractButton_Click(ByVal sender As System.Object, _
                                ByVal e As System.EventArgs) _
    Handles SubtractButton.Click

    ResultTextbox.Text = Str(FirstNumber.Value - SecondNumber.Value)
End Sub

```

- Early printings of the Student Textbook contained a draft section on the bottom of page 126. If you have a section entitled “**Return Keyword**”, then the entire section including the sample code and all text to the bottom of the page should be replaced with this final section instead:

Exit Keyword

There may be times when you want to exit a **For** loop before the loop has been normally completed. To do this, you can use the **Exit For** keywords, like this:

```
For i = 1 To 5
    MsgBox(i)
    Exit For ' break out of the For loop right now
Next
MsgBox("after for")
```

In this example you will only see one **MsgBox** pop-up showing the value 1, then the loop will end and you will see the “after for” message displayed next. You can also use this same “Exit” concept on the **While** and **Do-While** loops we describe in the next lesson! Just use the **Exit While** or **Exit Do** keywords instead.

- In first and second printings on page 147 in the “Zip Zap Latin” project, the two textbox controls are described with incorrect (**Name**) values that do not match the remainder of the activity guide. The following text:

Now you will need to do the following:

- Add a textbox control with the (**Name**) “OriginalText” and the **Text** “Enter a word:”
- Add a textbox control with the (**Name**) “TranslatedText” and the **Text** “Translated word:”

...should be replaced with:

Now you will need to do the following:

- Add a textbox control with the (**Name**) “WordTextBox” and a **Label** “Enter a word:”
- Add a textbox control with the (**Name**) “TranslateTextBox” and a **Label** “Translated word:”

- In first and second printings on page 150 in the “Simple Arrays” lesson of Chapter 12, the description of how to declare and initialize an array at the same time contains a syntax error:

```
Dim studentGrade(10) As Integer = {90, 95, 80, 85, 93, 92, 88, 75, 90, 80}
```

The correct statement should be:

```
Dim studentGrade() As Integer = {90, 95, 80, 85, 93, 92, 88, 75, 90, 80}
```

Visual Basic will automatically figure out how many elements you want in the array based on the listed values. If you attempt to specify the number of elements up front you will get a syntax error.

- The original Chapter Five test contains a question on material that is not covered until Chapter Six. You should move question 9 “**How do you clear all text from a text box on a form?**” to the Chapter Six test, replacing question 7 “**What is input validation?**”

The new question for Chapter Five question 9 is:

“**What operator will change a True value to False and a False value to True?**”

Answer: The “**Not**” operator.

- On page 119 the following code sample does not compile due to an invalid comment line:

```
Dim bottom As Integer = 0
Dim top As Integer = 1

// the next line raises a divide-by-zero exception
Dim result As Integer
result = top / bottom
```

The “//” characters (a C# and Java-style comment) should be replaced with a single quote (VB-style comment):

```
' the next line raises a divide-by-zero exception
```