Introduction to Visual C# Programming

Course Description:
Programmers start here. In this course, you'll learn key programming concepts and then apply them using Microsoft® Visual C# ® 2015 Express/Community Edition, the free version of Microsoft's Visual Studio ® developer toolkit. Complete a series of increasingly complex projects while you learn C#. You will learn two different skills: Designing Windows Forms Applications (the "Visual" part) and C# Programming Language and Syntax. At the end of this course you will have a portfolio of completed Visual C# executable programs.

Prerequisite:
No previous course prerequisites; this course is parallel to *Visual Basic*, so previous completion of *Visual Basic* will enrich the knowledge gained in the course. However, this course covers a lot of similar topics to *Visual Basic*, except in a more complex and rich language.

Skills: Moderate logical skills, some concepts from algebra/pre-algebra such as using equations, types of numbers (decimals, whole numbers, integers), and Boolean logic.

Difficulty Level:
Challenging. Mastery of prerequisites, familiarity with computers, or experience with another programming language will reduce difficulty significantly.

Inability to type/spell with accuracy will greatly harm a student's ability to perform well in this course.

Picking the Class That is Right for You:
Introduction to C# Programming focuses on more complex concepts whereas *Visual Basic* focuses more on user interface design. *Introduction to C# Programming* has a few more "free form" projects, where the students will work by themselves than *Visual Basic.* Students will learn basic UI design (the "Visual" part of "Visual C#") in this course, however some advanced concepts such as file input/output are not covered.

This course is a great supplement to a student who is planning on taking AP Computer Science or has already taken AP Computer Science.

Class Room Setup:
1. Install Visual Studio 2015, default settings work well.
2. Update Internet Explorer to Version 11.
3. Log into Account: ABTC.Compucamp2016@outlook.com / Texas2016

Course Overview:

1. Day 1:
    a. Chapter 1
    b. Chapter 2
    c. Chapter 3
        i. Guided Project: Beep
2. Day 2:
    a. Chapter 4
        i. Guided Project: Form Fun
        ii. Extra Project 1: Stop Watch
3. Day 3:
    a. Chapter 5
        i. Guided Project: Savings Account
4. Day 4:
    a. Chapter 6
        i. Guided Project: Guess the Number Game
        ii. Extra Project 2: Times Tables
5. Day 5:
    a. Chapter 7
        i. Guided Project: Sandwich Maker
6. Day 6:
    a. Chapter 8
        i. Guided Project: Blackboard Fun
7. Day 7:
    a. Chapter 9
        i. Guided Project: Card Wars
        ii. Extra Project 3: Dice Rolling
8. Day 8:
    a. Chapter 10
        i. Guided Project: Beach Balls
        ii. Extra Project 5: Memory Game

Comments to the instructor are in [brackets]. This is usually a warning or some tips.

Day 1:

Daily Class Goals:

- Introduce Instructor and Students
- Familiarize Students with Visual Studio & Visual C#
- Open an existing project in VC#
- Successfully Compile a Project in VC#
- Complete a Simple Project, with a Button Control that runs code.

Daily Plan:

- **Lesson 1: Sample**
    - Simple example project to show students how to load existing projects.
    - It will provide about 5 minutes of "exploration."
- **Lesson 2a: FirstTry**
    - [This is a Project where we will primarily just demonstrate the tools and windows in Visual C#.]
    - Explain the structure of a C# program (2-2): Solution, Project, Form, Controls.
    - Explain the concept of an "event." (2-3)
    - Create an example project with the students, and call it "FirstTry" (2-6).
    - Explain Code Window and Designer. (2-9)
        - Drag a button on a Form, resize it, change some of the properties.
    - Explain the Toolbox and Properties Windows
    - Open the "sample" project. (2-13)
    - Open the Code Window and look at the code with the students. (2-13)
        - Focus on trying to demonstrate how the properly named variables and methods can help make the code very straight forward (2-17).
- **Guided Project 3: Beep**
    - Create a new project.
        - We will be bouncing between these steps and the Step-By-Step project below.
    - Explain the process of Dragging and Dropping controls.
    - Review the Naming Conventions for Controls (3-11)
        - btn, lbl, txt, chk, rdo
    - Explain how to change properties with code (3-13)
    - Examine the structure of an Event (3-17): visibility, return types, and parameters. (Use Sample for this).
    - Explain how to Add a new Event (3-21).
    - Explain how Intellisense will help complete code (3-26)

Example Demonstrations and Comments:

- **Guided Project 3 Comments: Beep**
    - Steps are listed below.
    - Watch out for rampant double clicking. Double clicking a control will cause VS# to generate an event for that control. Doing so can make your code very bloated and confusing, and in rare cases break functionality.

- The concept of the "active" control is very important, as it will be the control that the properties are linked to, and also the control that will be the source when using "align" and other similar tools.
- Demonstrate how it is possible to select a control by clicking on it, or selecting it from the drop down menu in the Properties Panel. (3-10)
- For the AP Comp Sci people out there, since all (well, the default place) of the code is on the Form, the keyword "this" will refer to the current object, which is the Form. The Form actually has instance variables of the controls (which are also Objects, and their instantiation is hidden from us). It might seem contrary to what is taught in AP Comp Sci, because we are accessing public variables of Objects!
- Also for the AP Comp Sci people, the events typically are private methods on the Form (because why should an external object be calling an event?). This is further heresy when contrasted to the standard AP lesson of "always public methods, always private variables." Just a heads up for any students who are took/are taking AP Comp Sci.

<u>Guided Project 3 Steps:</u>

1. **<u>Create a new Project, call it Beep. (3-2)</u>**
2. Resize the Window. (3-3)
3. Add a Button. (3-5)
4. Move and Resize the Button. (3-6)
5. Add another Button, and other types of Contols. (3-6)
6. Delete all the Controls except one Button on the Form (3-6)
7. Using the Properties, set the Form's Width to 400 and Height to 250. (3-9)
8. Using the Properties, set the Button's Location->Top to 110 and the Location->Left to 170 (3-9)
9. Change the Button's Text to "Click Me". (3-10)
10. Change the Form's Text to "My First Code. (3-19)
11. Change the Button's Name to "btnBeep." (3-19)
12. Change *btnBeep*'s Text to "Beep!!" (3-19)
13. Add a new Click event to *btnBeep* (3-22)
14. Code the *btnBeep_Click* method by adding the code on page 3-23.
15. Test your Project
16. Add more code to the *btnBeep_Click* method by adding the code on page 3-25.
17. Test your Project
18. Complete!

Day 2:

Daily Class Goals:

- Discuss the idea of "Project Design"
- Learn how to create Events for Controls
- Learn the controls: Forms, Buttons
- Learn the basic datatypes of C#
- Guided Project: Form Fun
- Complete Extra Project 1: StopWatch

Daily Plan:

- **Lesson 4: Experimenting the Properties**
    - Show students how to access Help with F1 (4-5).
    - Explain the important properties of the Form (4-8).
        - Name, Text, BackColor, Icon, Width, Height, FormBorderStyle, StartPosition
    - Explain the important events of a Form (4-10)
        - Click, Load, FormClosing
    - Explain the important properties of a Button (4-12).
        - Name, Text, TextAlign, Font, BackColor, ForeColor, Left, Top, Width, Height, Enable, Visible
    - Explain the important events of a Button (4-17)
        - Click
- **Discuss the strictness of C# programming syntax (4-19)**
    - Case Sensitive, Spelling, Curly Braces, Parentheses for Function Calls, Indenting, and Semicolons.
- **C# Language:**
    - Assignment Operator, = (4-21)
    - Integers, int (4-22)
    - Colors, Color (4-23)
        - Not a Primitive datatype.
    - Booleans, bool (4-23)
        - Minor annoying point, in code, use all lowercase, despite what the Property Viewer might show.
    - Strings, String (4-24)
    - Comments, // (4-25)
- **Guided Project 4: Form Fun**
    - There is a Completed version of this project in the BVCS Projects folder.
    - See the steps below.
- **Extra Project 1: StopWatch**
    - Have the students work on StopWatch after this completing Project 4.

Suggestion Demonstration and Comments:

- **Lesson 4: Experimenting the Properties**
    - This project doesn't have a full set of steps.
    - Add two buttons as a "Start" and "Stop" button. Create code and set

initial properties so that Stop is disabled by default, and when you press Start, it enables Stop, but disables Start (4-15).

- Do the Same with Visible. Add a Label with the text “Running” on this form, which visibility set to false. When you click start, Visible will be True, when you click Stop, visible will be false.
- When going over the syntax of C#, it might be good to show examples in this project. Could even declare a few variables so students can see how it works.

<u>Guided Project 4: Form Fun Steps:</u>

1. Create a new Project. (4-26)
2. Add Six Buttons and arrange them in a 3x2 grid (4-26)
3. Change the Form.Text to "Form Fun" and Form.StartPosition to "CenterScreen" (4-27)
4. Set the Properties on the Buttons (4-27):
5. Create a Click event for *btnShrink*. (4-30)
6. Code *btnShrink_Click()*: (4-30)
7. Create a Click event for *btnGrow* (4-31)
8. Code *btnGrow_Click()*: (4-32)
9. Create a Click event for *btnRed* (4-32)
10. Code *btnRed_Click()* (4-32)
11. Create a Click event for *btnBlue* (4-32)
12. Code *btnBlue_Click()* (4-32)
13. Create a Click event for *btnHide* (4-32)
14. Code *btnHide_Click()* (4-32)
15. Create a Click event for *btnShow* (4-33)
16. Code *btnShow_Click()* (4-33)
17. Test your Project!
18. Project Complete!

Day 3:

Daily Class Goals:

- Discuss the Three Types of Errors
- Learn the controls: Labels and TextBox
- Learn about Working with Variables and Converting Types
- Guided Project: Saving Account

Daily Plan:

- **Lesson 5: Labels and TextBoxes (5-3)**
    - Explain the three types of errors (5-4)
        - Syntax, Run Time, and Logic
    - Explain the important properties of a Label (5-12).
        - Name, Text, TextAlign, Font, BackColor, ForeColor, Left, Top, Width, Height, BorderStyle, Visible, AutoSize
    - Explain the important events of a Label (5-15)
        - Click
    - Explain the important properties of a Text Box (5-17)
        - Name, Text, TextAlign, Font, MultiLine, ScrollBars, MaxLength, BackColor, ForeColor, Left, Top, Width, Height, ReadOnly, TabStop, BorderStyle, Visible
    - Explain the important events of a Text Box (5-20)
        - TextChanged, Leave
    - Discuss what "focus" means in terms of Controls (5-20)
- **C# Language:**
    - Variables (5-22)
        - Naming Rules (5-23)
    - Decimal/Floating Point Numbers, double (5-24)
    - Variable Declaration (5-25)
        - Scope: General Declaration and Local Declaration (5-26)
    - Type Casting (5-28)
        - Explicit Casting to convert type
    - Arithmetic Operators (5-29)
        - Addition, Multiplication, Subtraction, Division, Modulus
    - Operator Precedence (Like PEMDAS), same tier = same time (5-30)
        - 1st: * and /
        - 2nd: %
        - 3rd + and –
    - Method Calls (5-34)
    - Converting Datatypes (5-33)
        - Convert.ToInt32(str)
        - Convert.ToDouble(str)
        - Convert.ToString(intOrDouble)
    - String Concatenation (5-37):
        - String newStr = str1 + " and " + str2

Example Demonstrations and Comments:

- **Lesson 5: Labels and TextBoxes**
  - This project has no individual steps, aside from showing examples of the three kinds of errors.
    - Syntax Error (5-4)
    - Run-Time Error (5-6)
    - Logic Error (5-11)
  - This project has no steps, as it is another "test these things" type of project.
  - Add a label, turn off AutoSize and resize it and move it around. (5-13)
    - Labels typically serve as a means of data output and Form styling.
  - COMMON ERROR: A lot of students will try to use code to change the text of a label without accessing the .Text field. For example this line: lblExample.Text = "My Label Box"; will get incorrectly turned into: lblExample = "My Label Box";
    - Students think the label is being turned into that text, not the Text property.
    - Also, the Quotes on the String value are only needed when using Code to set the .Text field, not when using the Properties Window.
  - Clicking a Label is an odd event (mostly because the user doesn't have a clear cue like they do for a button), but it can be useful.
  - Try adding a Text Box to the project. Try having a button that when you click it, it sets the text in the text box to something, or sets a label equal to the stuff in the Text Box. (5-19)
    - Text Boxes typically serve two purposes: input control or a display. (5-19)
    - When using a TextBox as a display, remember to set ReadOnly to True.
  - The TextChanged event happens every keypress, so don't go crazy with it. (5-20)
  - Once the discussion turns to C# coding, it might not hurt to set up a button that updates a label with some math operations to demonstrate the different arithmetic operators. You might need to use String conversions without explaining them at first. After operators, THEN explain the String conversions.
- **Guided Project 5: SavingsAccount**
  - Steps are below.
  - For an extension, include the ability to use an interest rate and another TextBox and Label to to an input for the interest rate (5-46). See this page in the book to complete these extensions.

<u>Guided Project 5: SavingsAccount Steps:</u>

1. Create a new Project called "Savings". (5-38)
2. Add three Labels and across from each, a TextBox (total of 3) (5-38)
3. Add two Buttons at the bottom. (5-38)
4. Set the Properties on the Form (5-39):
5. Set the Properties on the Labels (5-39):
6. Set the Properties on the TextBoxes (5-40):
7. Set the Properties on the Buttons (5-40):
8. Code a general declaration:
9. Create a Click event for *btnCompute* (5-43)
10. Code *btnCompute_Click()*: (5-44):
11. Create a Click event for *btnExit* (5-44)
12. Code *btnExit_Click()*: (5-44):
13. Test your Project!
14. Project Complete!
15. Try Extension on 5-46 for other Ideas.

Day 4: Chapter 6: UpDown Control, Decisions, Random Numbers

Daily Class Goals:

- Discuss the Three Types of Errors
- Learn the controls: UpDown Control
- Learn about If Statements and Random Numbers
- Guided Project: Guess the Number Game
- Extra Project 2: Times Tables

Daily Plan:

- **Lesson 6: Numeric Controls (6-2)**
    - Explain the important properties of a UpDown Controls (6-2).
        - Name, Value, Increment, Maximum, Minimum, TextAlign, Font, BackColor, ForeColor, Left, Top, Width, Height, ReadOnly, Enable, Visible
    - Explain the important events of an UpDown Control (6-5)
        - ValueChanged
- **C# Language:**
    - Logic Expressions (6-6)
        - Comparison Operators (6-8)
            - Equal to: ==
            - Not Equal to: != (6-9)
            - Greater Than: >
            - Less Than: <
            - Greater Than or Equal to: >= (6-10)
            - Less Than or Equal to: <=
        - Logical Operators (6-11)
            - And: &&
            - Or: || (6-12)
            - Not: !
            - Xor: ^
            - Operator Precedence: Not, And, Or, Xor (6-14)
    - If-Statements and If-Else Statements (6-16)
    - If-Else-If Statements (6-19)
    - Random Numbers (6-22)
        - Random Number Object Instantiation (6-22)
        - .Next(limit) command (6-23)
        - Example of a die roll (6-24)

- **Guided Project 6: Guess the Number Game (6-25)**
    - The Steps are included below.
    - The extensions include adding a “cold / warmer / hot” system that tells you how close you are to the number, in addition to the higher or lower system. (6-37)
        - This uses Math.Abs (num).

Project Notes and Comments:

- **Lesson 6: Numeric Controls**
    - This project has no individual steps, aside from demonstrating the UpDown control and Random Numbers.
    - Insert an UpDown control and change the settings (6-4)
        - Keep in mind that the UpDown's value is a Double, so if you plan to use it as an Int, you would need to Cast it as an Int.
    - During the C# Language discussion, use the project to demo the different logical expressions.
    - For if-statements, the Lemonade Stand example could be added to this project (6-17)
        - Watch out for the typo with the capital Else. Else should be lowercase in the code.
        - The logic error on (6-21) is a great example to demonstrate to the students of else-if chaining.
    - Add a button and a label and test changing the label to a random number by pressing the button (6-23)

- **Guided Project 6: Guess the Number Game (6-25)**
    - Steps are below.
    - Extension: Textbox to set the max value (6-37)
    - Extension: Absolute value Cold / Warm / Warmer hints

- **Extra Project 2 – Times Tables**
    - This a good project to reinforce random numbers.
    - The only odd parts of this project come from the *txtAnswer_KeyPress()* method.
        - The first part (the e.KeyChar part) is checking for only numeric key presses and if the enter key is pressed.
        - The other confusing part is the String.Format command on the percentage.

<u>Guided Project 6: Guess the Number Game:</u>

1. Create a new Project called "GuessNumber". (6-26)
2. Add a TextBox, a Numeric UpDown, and Three Buttons (6-26).
    a. The TextBox will display the result, the UpDown is the guess, and the buttons will be a "Guess" button, a "New Number" button, and an "Exit"
3. Set the Properties on the Form: (6-27):
4. Set the Properties on the TextBox and UpDown (6-27):
5. Set the Properties on the Buttons (6-28):
6. Create a Click event for all three Buttons (6-30)
7. Code a general declaration: (add these after the Form1 constructor):
8. Code *btnPick_Click()*: (6-33):
9. Code *btnCheck_Click()*: (6-34):
10. Code *btnExit_Click()*: (5-44)
11. Test your Project!
12. Project Complete!

Try Extension on 6-37 for other Ideas.
-Use a textbox to set the max value of the game
-Add a freezing / cold / warm / hot system using Absolute value command Math.Abs(int)

<u>**Day 5: Chapter 7: Icons, Group Boxes, Check Boxes, Radio Buttons**</u>

<u>Daily Class Goals:</u>

- Discuss the Importance of grouping controls and how that can help create more complex interactivity between input and output (as opposed to just simple buttons).
- Learn the controls: Icons, Group Boxes, Check Boxes, Radio Button
- Learn about Switch Statements
- Guided Project: Sandwich Maker

<u>Daily Plan:</u>

- **Lesson 7: Grouping Controls (7-2)**
    - Explain how .ico (<u>**Icon**</u>) files work (7-2).
        - Assigning Icon Files (7-7).
    - Explain the important properties of a <u>**Group Box Control**</u> (7-8).
        - Standard Properties such as Name
        - Putting other controls into a Group Box and Moving them (7-10).
    - Explain the important properties of a <u>**Check Box Control**</u> (7-13).
        - ***Checked***: True or False if this box is checked
        - When using check boxes, it is recommended to always use a Group Box.
        - Events: ***CheckedChanged***: When the checked property changes.
    - Explain the important properties of a <u>**Radio Button Control**</u> (7-17).
        - ***Checked:*** True or False if this is selected—only one Radio Button per group can be Checked.
        - Radio Buttons are best used with Group Boxes because they will exhibit mutually exclusive behavior automatically if in a group.
        - Should always default one Radio button to True.
        - Events: ***CheckedChanged***: When the checked property changes (from either setting to the other)
- **C# Language:**
    - Logic Expressions (7-22)
        - Switch (7-22)
            - Case
            - Break
            - Default

- **Guided Project 7: Sandwich Maker (7-25)**
    - The goal of this project is to create a computerized sandwich ordering menu. The design will allow for the user to pick one bread (white, wheat, rye), any number of meats (roast beef, ham, turkey, pastrami, salami), one cheese (none, American, Swiss), and any of six condiments (mustard, mayonnaise, lettuce, tomato, onion, pickles).
    - The extension adds the ability to calculate price and change.
    - The Steps are included below.

Project Notes and Comments:

- **Lesson 7: Grouping Controls (7-2)**
  - This project has no individual steps, aside from demonstrating the Group Boxes, Check Boxes, and Radio Buttons.
  - Insert Group Control and other controls (7-11).
    - Keep things to demonstrate:
      - Deleting the group, deletes all the controls
      - Setting Enable or Visible for the group will change all of the controls inside.
  - When using switch statements, break; is required, otherwise it will cascade into the next case.

- **Guided Project 7: Sandwich Maker (7-25)**
  - They use the CheckChanged feature of Radio Buttons to determine which type of bread or cheese is selected (by changing the variable). Technically, you can also do this just with a bunch of if-statements when they click the "order" button (checking each radio button for Checked). Using CheckedChanged event is more efficient.
  - The logic explanations on 7-36, 7-39, and 7-42 are great to include in your guided discussion.
  - Steps are below.

<u>Guided Project 7: Sandwich:</u>

1. Create a new Project called “Sandwich”. (7-26)
2. Add three Radio Buttons and group them together (Group 1) in the top left of the UI. (7-26)
3. Add five Check Boxes and group them together (Group 2) in the bottom left of the UI (7-26)
4. Add three Radio Buttons and group them together (Group 3) in the top middle of the UI (7-26)
5. Add six Check Boxes and group them together (Group 4) in bottom middle of the UI (7-26)
6. Add a Label and Textbox in the top right (7-26).
7. Add two Buttons to the bottom right (7-26)
8. Set the Properties on the Form: (7-27):
9. Defaults for Groups:
10. Defaults for Controls:
11. Names for Group 1: (7-27):
12. Names for Group 2: (7-28):
13. Names for Group 3: (7-29):
14. Names for Group 4: (7-30):
15. Set the Properties on Label and TextBox (7-31)
16. Set the Properties on the Buttons (7-32)
17. Create a Click event for both Buttons (7-34)
18. Code a general declaration: (add these after the Form1 constructor)
19. Code *Form1_Load(object sender, EventArgs e)*: (7-36)

Set up the bread Radio Buttons:

20. Code *rdoWhite_CheckedChanged*: (7-37)
21. Code *rdoWheat_CheckedChanged*: (7-37)
22. Code *rdoRye_CheckedChanged*: (7-37)

Set up the Cheese Radio Buttons:

23. Code *rdoNone_CheckedChanged*: (7-38)
24. Code *rdoAmerican_CheckedChanged*: (7-38)
25. Code *rdoSwiss_CheckedChanged*: (7-38)

Set up the main order button, read the explanation on 7-41:

26. Code *btnOrder_Click*: (7-39)
27. Code *btnExit_Click()*: (7-43)
28. Test your Project!
29. Project Complete!

Try Extension on 7-45 for other Ideas.

-Create an Exit Button
-Calculate Price
-Enter a payment amount and calculate change
-Super challenge: calculate the bills and coins required

Day 6: Chapter 8: Panels, Mouse Events, Colors

Daily Class Goals:

- Learn how to draw shapes in Visual Basic
- Learn how to use panels to add custom graphics to your designs
- Learn how to use the mouse as a source of input
- Guided Project: Blackboard Fun

Daily Plan:

- **Lesson 8: Panel Control (8-2) [10 mins]**
    - Explain the important properties of a Panel(8-2)
        - Similarity to Group Box(8-2)
        - Name, BackColor, Left, Top, Width, Height, Enabled, Visible
- **C# Language [90 mins]**
    - Creating a Graphics object
        - Declaration
        - Initialization (CreateGraphics()) (8-5)
    - Creating a Color Object (8-6)
        - Declaration
        - Initialization
        - Dot Operator (.) to find a list of Colors (8-8)
        - FromARGB (8-9)
    - Creating a Pen Object (8-12)
        - Using a Pen object to draw on panels
        - Instantiation
        - Dispose() (8-12)
    - Coordinate Systems (8-14)
        - (0, 0) is the top left
    - Graphics Methods
        - DrawLine() (8-15)
        - Clear() (8-7)
        - Dispose()
    - Mouse Events (8-23)
        - Properties - Button, X, and Y
        - Events: MouseDown (8-24), MouseUp (8-29), MouseMove (8-30)
- **Guided Project 8: Blackboard Fun(8-31) [75 mins]**
    - The goal of this project is to understand graphics in Visual Basic by creating a small paint application. Students can use two buttons to select between colors and an eraser in order to draw on a panel.

- Steps are included below

<u>Example Demonstrations and Comments:</u>

- **Project 8: DrawLine Example(8-15)**
    - Note the use of a variable MyPen vs the use of a static reference Pens.Black
    - There is a graphics review to help remember the steps to creating graphics(8-17)

<u>Guided Project 8: Blackboard Fun:</u>

1. Create a project called "Blackboard" (8-33)
2. Add a Panel on the left side of the UI (8-33)
3. Add 9 Radio Buttons and group them together (Group 1) (8-34)
4. Add two Buttons to the bottom right of the UI (8-36)
5. Set the properties on the form: (8-33)
6. Set the properties for the Panel
7. Name for Group1:
    a. groupBox1
        i. Name: grpColor
        ii. Text: "Color"
        iii. BackColor: "Black"
        iv. ForeColor: "White"
        v. Font Size: 10
        vi. Font Style: Bold
    b. radioButton1: rdoGray, Text: 10-15 blank spaces in order to display the colors, this will be the same for all Radio Buttons in this group
    c. radioButton2: rdoBlue,
    d. radioButton3: rdoGreen
    e. radioButton4: rdoCyan
    f. radioButton5: rdoRed
    g. radioButton6: rdoMagenta
    h. radioButton7: rdoYellow
    i. radioButton8: rdoWhite
8. Set these properties for Button 1:
    a. Name: rdoWhite
    b. Text: Erase
9. Set these properties for Button 2:
    a. Name: btnExit
    b. Text: Exit

10. Create a Load event for the Form (8-39)
11. Create a CheckedChanged event for each Radio Button (8-39)
12. Create a Click event for both Buttons(8-41)
13. Code global declarations (8-38)
14. Code the Load event for the Form(8-39)
15. Code the CheckedChange Event for each RadioButton(8-40)
16. Code the btnErase_Click event(8-41)
17. Code the btnExit_Click event(8-41)
18. Create MouseDown, MouseMove, and MouseUp Event for the Panel(8-42)
19. Create a Closing event for the Form(8-44)
20. Code the MouseDown event(8-42)
21. Code the MouseMove event(8-43)
    a. Explore what happens if you don't include the XLast and YLast lines
22. Code the MouseUp event(8-43)
23. Code the FormClosing event(8-44)
24. Test your project

Day 7: Chapter 9: Picture Boxes, Arrays

Daily Class Goals:

- Learn how to display images in order to create more sophisticated user interfaces
- Explore the importance of using data structures such as arrays to hold data
- Learn properties of different image types
- Learn how to loop through data with For loops
- Guided Project: Card Wars
- Complete Extra Project 3: Dice Rolling

Daily Plan:

- **Lesson 9: PictureBox Control (9-2) [25 mins]**
    - Explain the important properties of PictureBoxes (9-2)
        - Name, Image, SizeMode, BorderStyle, Left, Top, Width, Height, Enabled, Visible
    - Explain the difference in image formats (9-4)
        - Bitmap, GIF, JPEG
    - Demonstrate how to use a PictureBox (9-6)
    - Demonstrate and explain the SizeMode property (9-11)
        - Normal, CenterImage, StretchImage, AutoSize, Zoom
    - Explain the important events associated with PictureBoxes (9-16)
        - Click, MouseDown, MouseMove, MouseUp
- **C# Language [45 mins]**
    - Arrays (9-18)
    - For Loops (9-21)
    - Block/Method Level Variables(9-26)
    - Shuffle Routine (9-28)
        - Shuffle items randomly in a list
- **Guided Project: Card Wars [90 mins]**
    - This project allows one to create a copy of the card game War. Using the shuffle routine and a basic computer AI, players can create their own version of the game.

Project Notes and Comments :

- **Lesson 9: PictureBox Example (9-6)**
    - There are many more image formats than the ones found on previous pages. If someone wants to use an image from an outside source, it may not show up in the file chooser by default. If this happens, make sure All

  - Files is selected as the file type in the file chooser.
  - There is a quickstart guide to using PictureBoxes at (9-17) if there are too many steps for the kids to remember.
- **C#: Loops**
  - There may be errors at first due to the syntax of initialization, expression, and increment.
- **Shuffle Routine**
  - If the concept proves to be too difficult to explain, (9-28) has a visual diagram that may help.
- **Guided Project: Card Wars**
  - This is a large project tying together many concepts from previous chapters. More time and explanations may need to be allotted for this one.
  - The card comparisons can be fairly confusing at first. Go over the logic from (9-41) to (9-42) for a breakdown of steps.
- **Extra Project: Dice Rolling**
  - This is a good project to further demonstrate randomness. It's fairly straight-forward, but it may be confusing seeing MyRandom.Next(6) + 1 in order to get a random number from 1 to 6. Originally, a random number is chosen from 0 to 5, and adding one shifts it up.

<u>Guided Project: Card Wars</u>

1. Create a project called CardWars (9-34)
2. Place two Panels, one on top of the other, in the middle of the screen (9-35)
3. Place a Label and a PictureBox in each Panel
4. Place a Label and TextBox to the left of each Panel
5. Place a TextBox, two Buttons, and Four PictureBoxes on the right side of the form.
6. Set the properties for each of these elements (9-36)
7. Create a Click event for btnNew and btnExit (9-41)
8. Code the btnNew_Click event (9-45)
9. Code the btnExit_Click event (9-49)
10. Project Complete!
11. To add to the project, try adding more players or having the deck reshuffle and continue playing when cards run out.

<u>Day 8: Chapter 10: Timers, Animation, and Keyboard Events</u>

<u>Daily Class Goals:</u>

- Learn how to use Timers in a program in order to execute code at specific intervals
- Learn new ways to use Graphics objects to draw on panels
- Learn the basics of programming driven animation
- Learn about basic collision handling between animated elements
- Explore optimization techniques for animating images
- Learn how to utilize keyboard inputs
- Learn about ASCII values
- Guided Project: Beach Balls
- Extra Project: Memory Game

<u>Daily Plan:</u>

- **Lesson 10: Timer Control [30 mins]**
    - Describe the important properties of the Timer (10-2)
        - Name, Interval, Enabled
    - Explain the important event of the Timer (10-3)
        - Tick
    - Explain the Not operator (10-7)
    - Learn the DrawEllipse method to draw ellipses (10-9)
- **C# Language [75 mins]:**
    - Cover new methods of Graphics
        - DrawImage (10-13), FillRectangle(10-25)
    - Examine different types of border collision detection
        - Image Disappearance (10-17), Border Crossing (10-20)
    - Cover Brushes (10-25)
    - Cover Collision Detection (10-28)
    - Cover the important events for keyboard events and explain their differences (10-33)
        - KeyDown (Gets any key, but doesn't differentiate between capitals and lowercase), KeyPress(Gets any key with an ASCII value; can differentiate upper and lower case)
    - Discuss giving focus to elements (10-33)
    - Discuss a way to get the ASCII value of a character (10-38)
    - Discuss key trapping (10-39)
- **Guided Project: Beach Balls [75 mins]**
    - This project utilizes topics from previous chapters along with much new

  - information. It has lots of typing so slow typers may need extra time to complete this one.
  - This project is fairly straightforward but has a lot of logic dealing with collisions that can become tricky
- **Extra Project: Memory Game [60 mins]:**
  - This is a fairly simple project with a lot of typing. You might not have time after the last project.

Project Notes and Comments:

- **Lesson 10: Timers (10-2)**
  - Timer intervals are in milliseconds, not seconds. Students may think put in values that are too small at first
- **Lesson 10: Animations**
  - Makes sure students are drawing the image after the erasing occurs, otherwise they will not see an image. (10-19)
- **Lesson 10: Image Disappearance and Border Crossing**
  - Make sure the students get all of the comparisons and signs right in their logic (10-23)
- **Lesson 10: Keyboard Events**
  - Makes sure the students use the Focus method to focus on a particular element.
  - If keys aren't working, make sure the student is using the correct event or the correct ASCII code.

Guided Project: Beach Balls

1. Place a Panel, Label, two TextBoxes, two Buttons, two Timers, and two PictureBoxes on the form. (10-43)
2. Set the properties for each of these elements (10-44)
3. Create a Click event for btnStart and btnExit. (10-48)
4. Create a KeyDown event for Form1
5. Create a tick event for both timers
6. Create global variable declarations (10-49)
7. Code the Form1_Load and Form1_FormClosing Events (10-50)
8. Code the Form1_KeyDown event (10-51)
9. Code the btnExit_Click event (10-52)
10. Code the btnStart_Click event(10-53)
11. Code the timGame_Tick event (10-54)
12. Code the timBalls_Tick event(10-56)
13. Project Complete!
14. If you still have time, you can move on to the extra project, or you can try to improve this project by adding an animation when the ball pops, adding difficulty levels to the game, make the game harder as time goes on, or display the number

of seconds remaining to the player.