Chapter 13

Programming VBA in MS Word

In this chapter, we will learn the following to World Class Standards:

- **Setting the Margins**
- **Setting Font and Font Size**
- **Adding a Header and a Footer**
- **Adding Text to a Document**
- **Adding Numbers to the Beginning of a Sentence**
- **Inserting a Table**
- **Inserting the Control Toolbox Toolbar**
- **Inserting and Modifying the Option Button**
- **Inserting and Modifying Command Buttons**
- **Saving a Document**
- **Programming a Clear Button**
- **Programming the Score Button**
- **Adding Comments in VBA to Relay Programming Information**
- **Declaring Variables in a Program with the Dimension Statement**
- **Assigning Values to Variables**
- **Testing a Case with the If -Then Function**
- **Adding in Visual Basic**
- **Using a Message box in Visual Basic**
- **Using a Label in Visual Basic**

Setting the Margins

To set the margins, go to the Page Setup after selecting File on the Menu Bar. Select the Page Setup (Figure 13-1) and the Page Setup window will appear on the display. The traditional Word margins are 1-inch on the top and bottom and 1.25 inch on the left and right. We want to change the left and right margins to 1-inch (Figure 13-2).

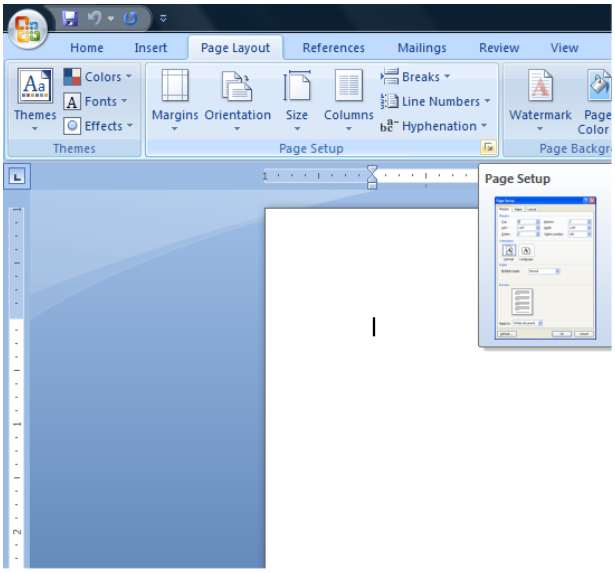

Figure 13-1 – Page Setup Panel

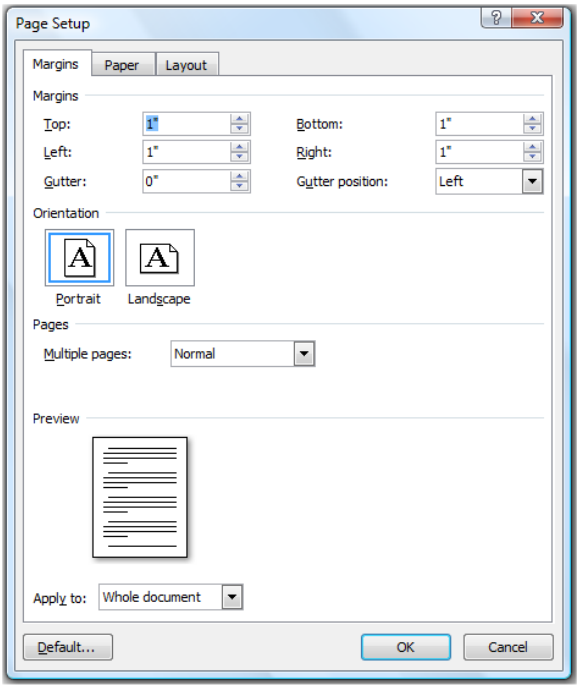

Figure 13-2 – Selecting Margin Setting

Changing the Font and Font Size

On the left side of the **Formatting** toolbar is the **Font** Listbox. The default font and size for Microsoft Word 2003 is **Times New Roman** with a font size of **12**. We are going to change the font to **Georgia** and the size to **10**. Begin by clicking the top left drop-down arrow of the font Listbox to select the font style. The font styles are in alphabetical order so scroll down until we see **Georgia** and select it (Figure 13-3). After picking the style, click the drop-down arrow to the right of the font style box and select a font size of **10** (Figure 13-4).

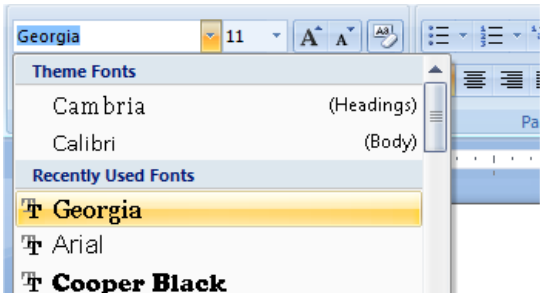

Figure 13-3 – Selecting "Georgia"

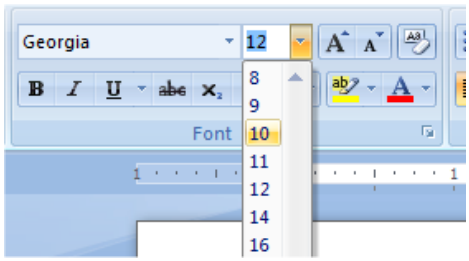

Figure 13-4 – Selecting Size "10" Font

Adding a Header and a Footer

Now, we are going to add a header to our document that reads, "**Algebra Pretest.**" To make the header and footer sections appear, double-click at the top or bottom of the page on the screen. On page one, place the cursor in the header section and choose the **Center** tool on the **Paragraph** panel under the Home tab. With the header centered, we can now type in the text "**Algebra Pretest**" (Figure 13.6).

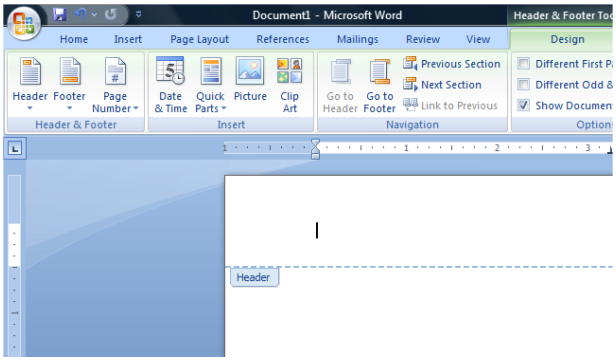

Figure 13.5 – Header Section

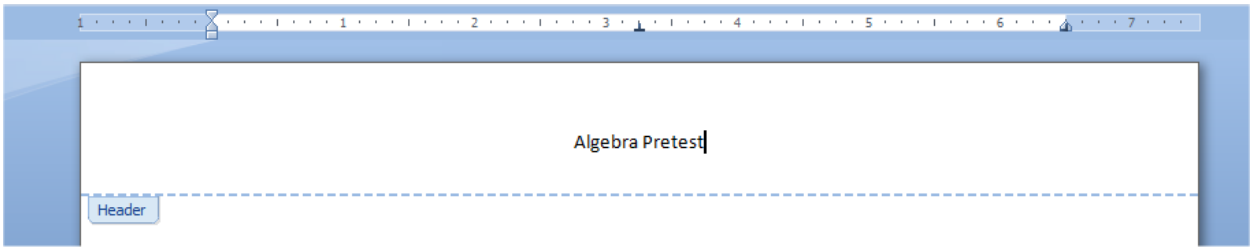

Figure 13.6 – Inserting the Header into the Document

Adding Text to the Document

Now, we will type the instructions after the header as shown in figure 13.7.

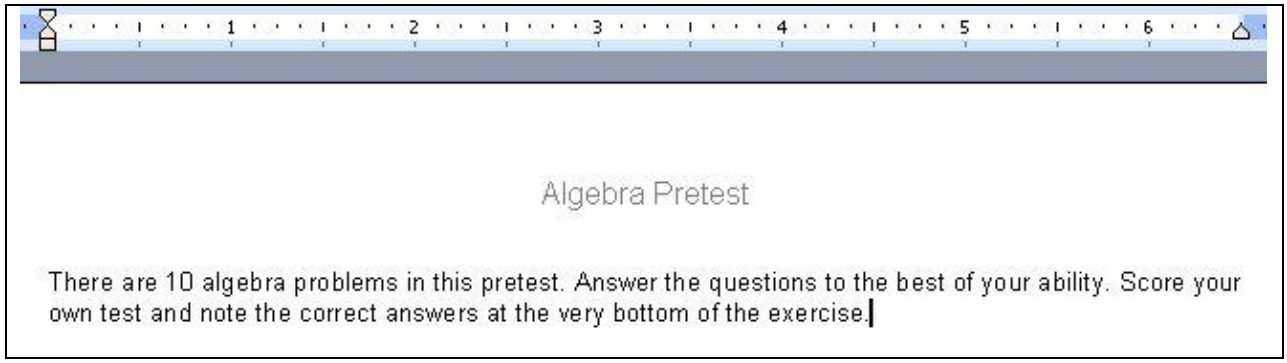

Figure 13.7 – Inserting the Text into the Document

Adding Numbers to the Beginning of a Sentence

Next, we will pick the **Numbering** tool on the **Paragraph** panel on the Home tab to begin the first question in the algebra pretest. After the number one, type the query "The answer to A = 2 + 5 is" as shown in figure 13.8.

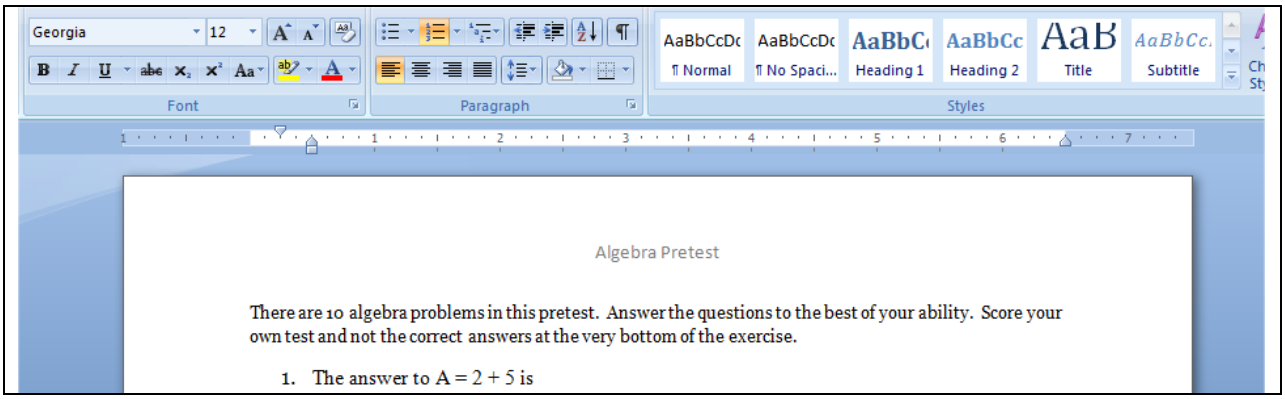

Figure 13.8 – Using the Numbering Tool to Place Numbers in Front of a Sentence

Inserting a Table

After the first question, we will want to have four possible answers, so we will insert a four-column single row table below the question. To insert a table into our document, we go to the Insert table tool on the Standard toolbar. On the drop-down panel, that appears under the tool, select a 1x4 table (Figure 13-9). We can also insert a table by going to Table under the Insert tab. We can pick the number of rows and columns we want in the table in the Insert Table window that appears. In this case, we want 4 columns and 1 row (Figure 13-10). Both methods of inserting a table are equally effective but the first method is generally faster.

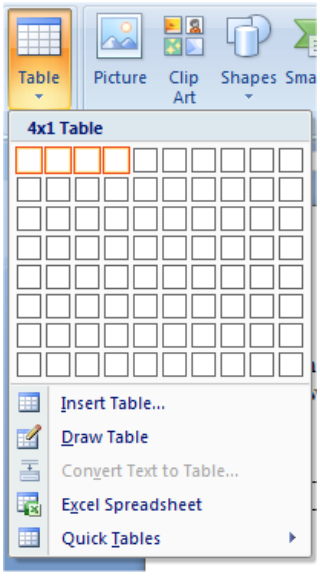

Figure 13-9 – Selecting 1x4 Table

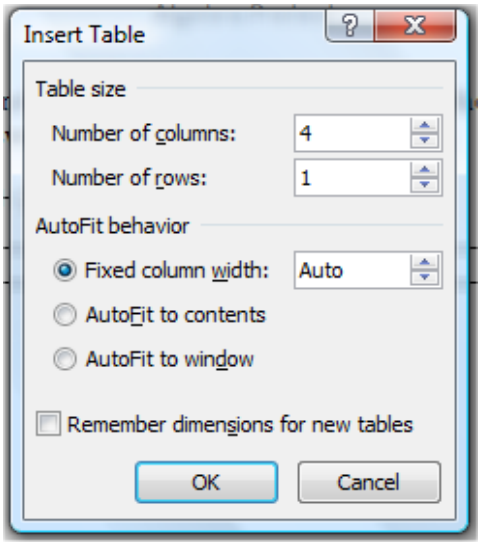

Figure 13-10 – Insert Table Window

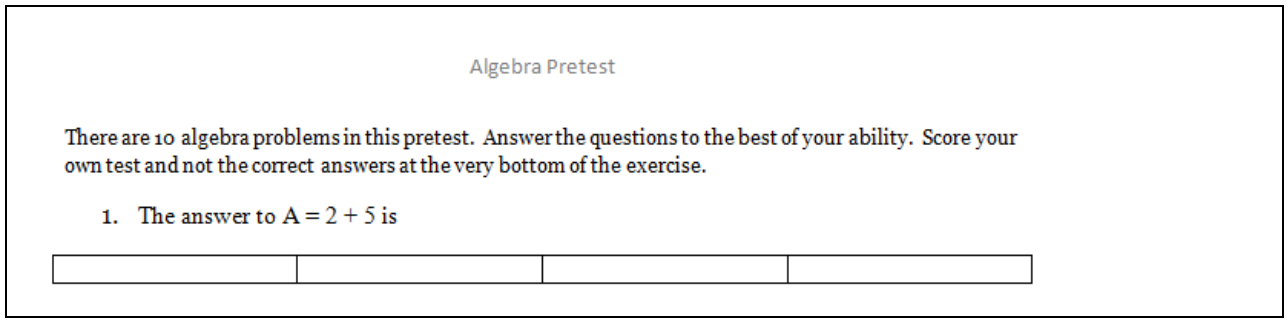

Figure 13.11 – 1x4 Table to Hold the Answers to the Question

After inserting the table, we want to remove the border from around the cells. Clicking on the move button will select the entire table. With the entire table selected, go to the border function on the **Paragraph** panel under the Home and click the drop-down arrow on the bottom right tool (Border Settings). In the list of options that appears, select the **No Border** option (Figure 8.12). The dark lines bordering the table should now turn to light grey lines. Now the table should appear as in Figure 8.13.

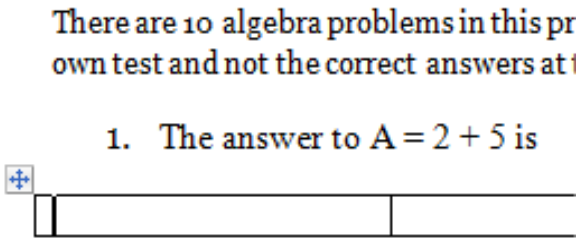

Figure 13.12 – Move Button

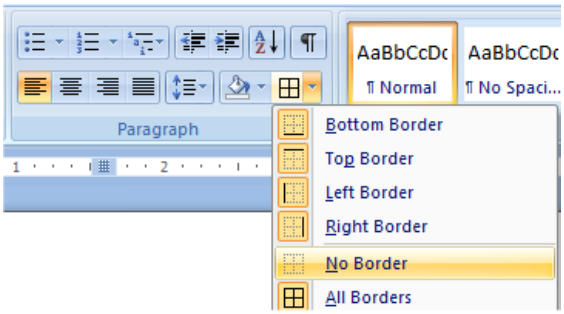

Figure 13.13 – No Border Option

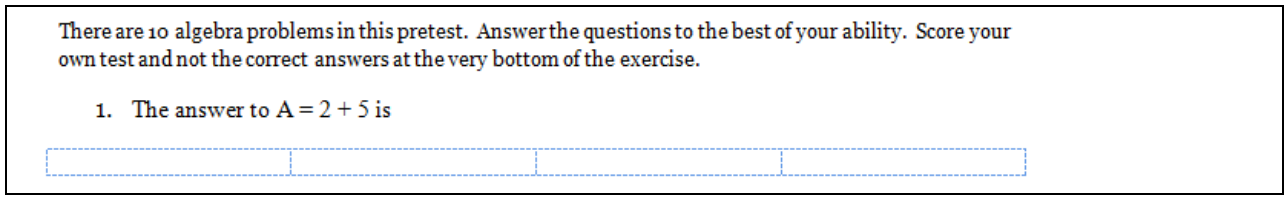

Figure 13.14 – No Border on the Table

Inserting the Control Toolbox Toolbar

We want to add another panel to the ribbon that contains tools we will need to continue. By clicking the Office button at the top left corner of the screen and then selecting Word Options at the bottom right of that menu will bring up the Word Options window as shown in figure 13.15. When the menu appears, check the box that reads, "Show Developer tab in the Ribbon" under the Popular options submenu. Hit OK to save this change.

The Developer tab contains panels that will allow us to add specific elements that are not standard to everyday word processing. The tools we will use are the option button, command button and the label. We will use other tools in future lessons.

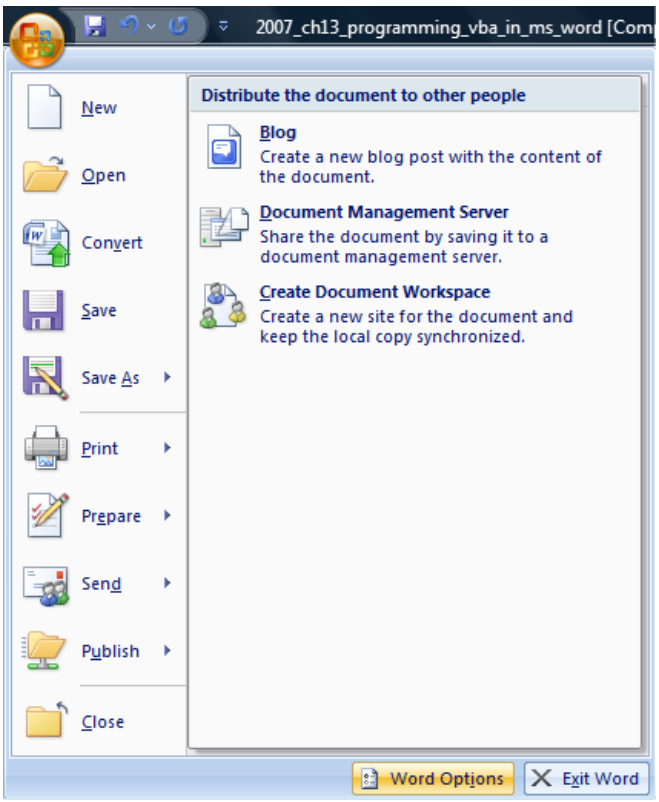

Figure 13.15 –Enabling the Developer tab

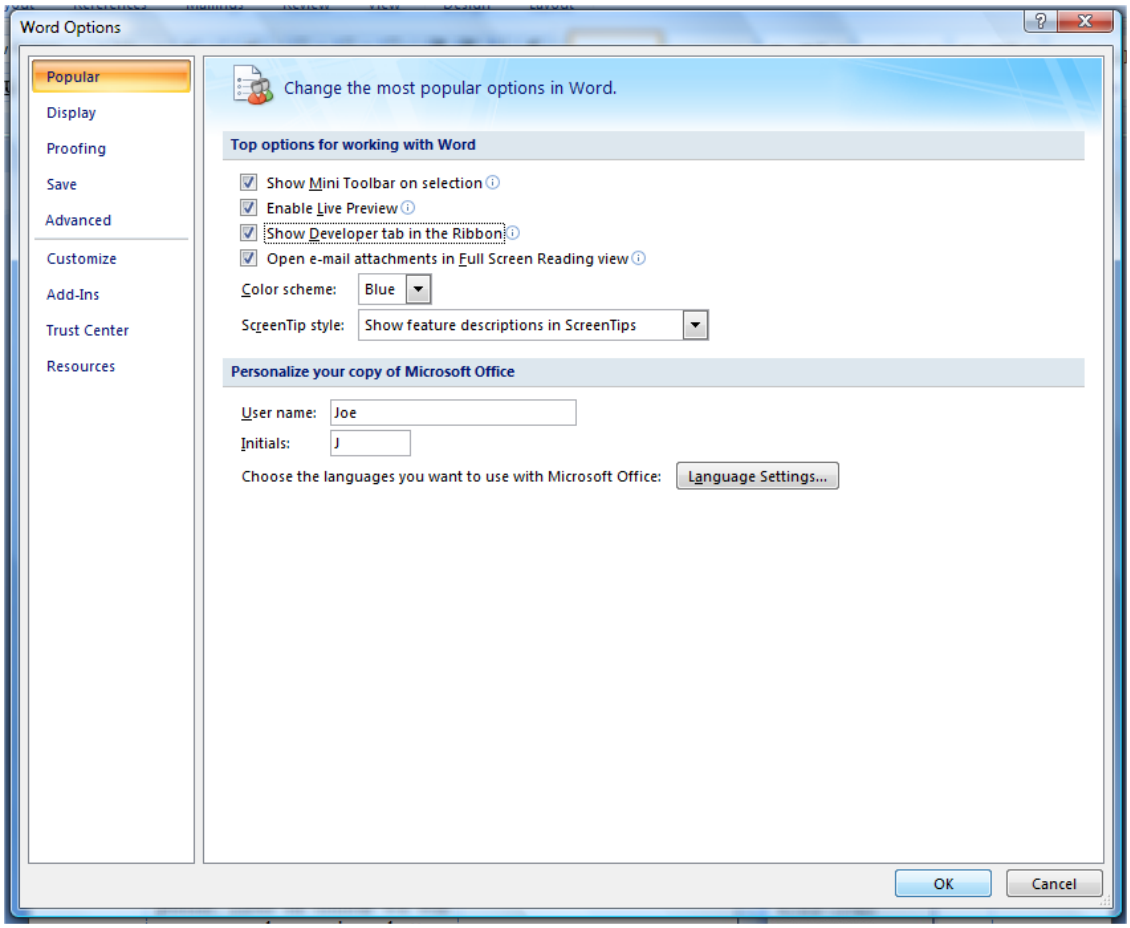

Figure 13.16 – The Word Options menu

Inserting and Modifying the Option Button

To insert an option button into the first cell of the table, we select the Option Button tool on the Controls panel under the Developer tab as shown in figure 13.17. OptionButton1 will appear as shown in figure 13.18. Make sure that Design Mode is enabled on the Controls panel. Right-click on the OptionButton and select Properties to continue.

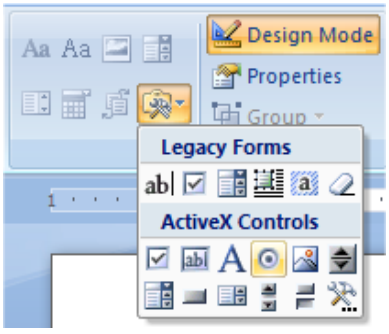

Figure 13.17 – The Option Button Tool

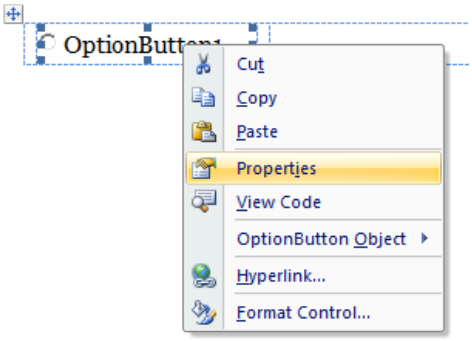

Figure 13.18 – The Properties Tool

We will change the **name** in the Properties panel to **opt1A** since this is answer A for question 1. We make the first option for question 1 as 2 so we type the number 2 in the Caption box. To make the four option buttons for the first question belong to the same group, we call the property, GroupName "1".

Alphabetic	
(Name)	opt1A
Caption	2
GroupName	1

When the four option buttons are in the same group, the black dot will move only to the option button we select.

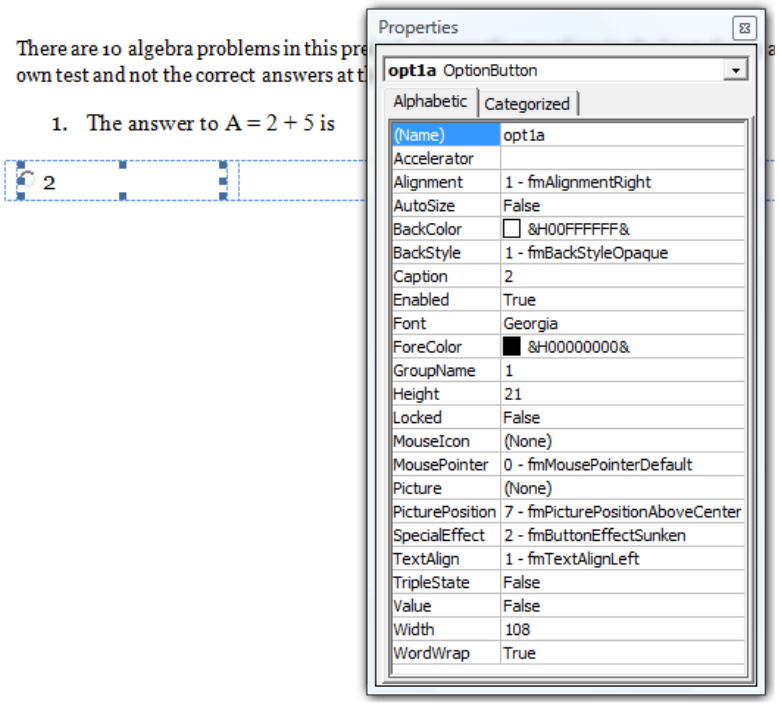

Figure 13.19 – Changing Properties

Insert the second option button into the next cell in the table. We will change the **name** in the Properties pane to **opt1B** since this is answer B for question 1. We will make this option for question one 5 so we type the number 5 in the Caption box. We still call GroupName "1".

(Name)	opt1B
Caption	5
GroupName	1

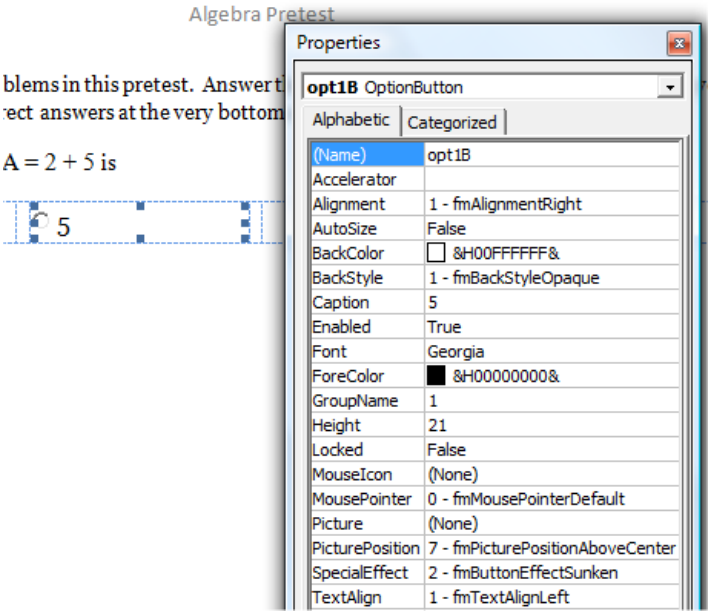

Figure 13.20 – Changing Properties

The third option button we insert into the next cell in the table, we will change the **name** in the Properties pane to **opt1C** since this is answer C for question 1. We will make this option for question 1 as 7 so we type the number 7 in the Caption box. We still call the property, GroupName "1".

(Name)	opt1C
Caption	7
GroupName	1

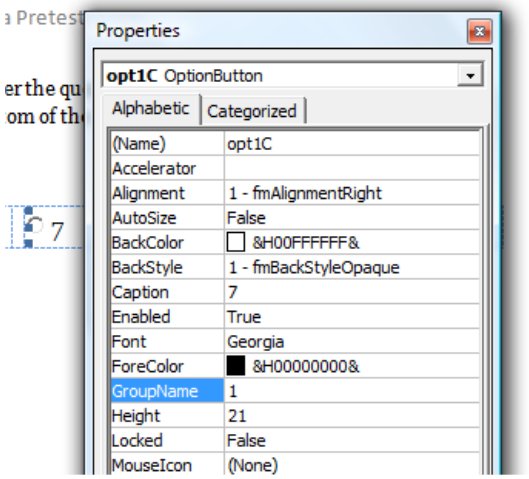

Figure 13.21 – Changing Properties

The second option button we insert in the next cell in the table, we will change the **name** in the Properties pane to **opt1D** since this is answer D for question 1. We will make this option for question 1 as A so we type the letter A in the Caption box. We still call the property, GroupName "1".

(Name)	opt1D
Caption	A
GroupName	1

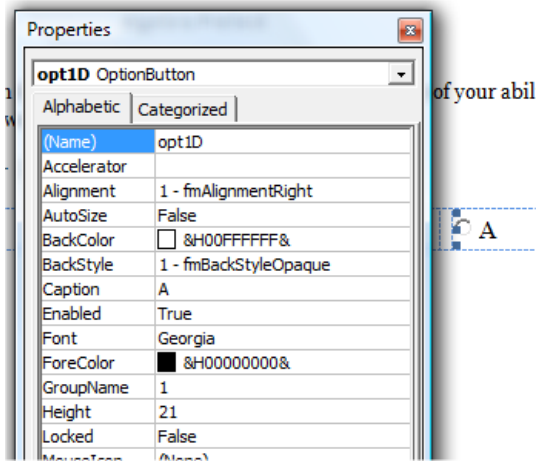

Figure 13.22 – Changing Properties

We continue to add ten questions, so we type "**2. The answer to B = -4 -5 is**" and then we add another 4 x 1 table and place another four option buttons in the cells. Here is the list of questions and possible answers, so we can finish the document.

2. The answer to B = -4 – 5 is			
-5	4	9	-9
3. The answer to y = 2x + 3 when x = -2 is			
-1	-2	-7	1
4. The answer to 2y = 5x – 6 when x = 1 is			
0.5	-0.5	1	-2
5. Simplify the expression 2A + 3B + 4A + B			
2A + 4B	4A + 4B	6A + 4B	7A + 3B
6. Simplify the expression -3M + M – 7 + 5M			
9M - 7	-3M + 7	3M + 7	3M - 7
7. Simplify the expression 2 (x + 2y)			
2x + 4y	x + 8y	4x + 4y	-2x - 4y
8. Simplify the expression -3 (x - 4y)			
3x - 12y	12x - 3y	3x - 3y	-3x + 12y
9. What is the answer to (x + y) (x – y) if x = 2 and y = 3			
-10	5	-5	-3
10. What is the answer to (x + y) (x – y) if x = -1 and y = 4			
9	-9	-15	15

After the last question, we will want to have two command buttons and a label to show correct answers, so we will insert a three-column single row table below the last question. To insert another table into our document, we go to the Table tool on the Insert tab. On the drop-down

panel, that appears under the tool, select a 1x3 table (Figure 13.23).

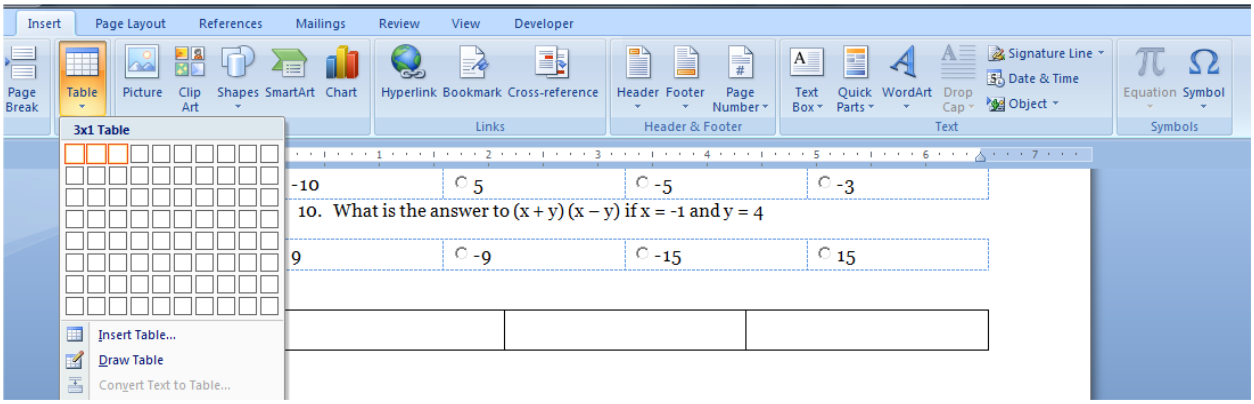

Figure 13.23 – Inserting another Table

We place our mouse on the bottom horizontal table lines and we hold down the left button and we make the row higher. We need to resize the cells approximately as shown in figure 13.24. Now, we are ready to insert the command buttons.

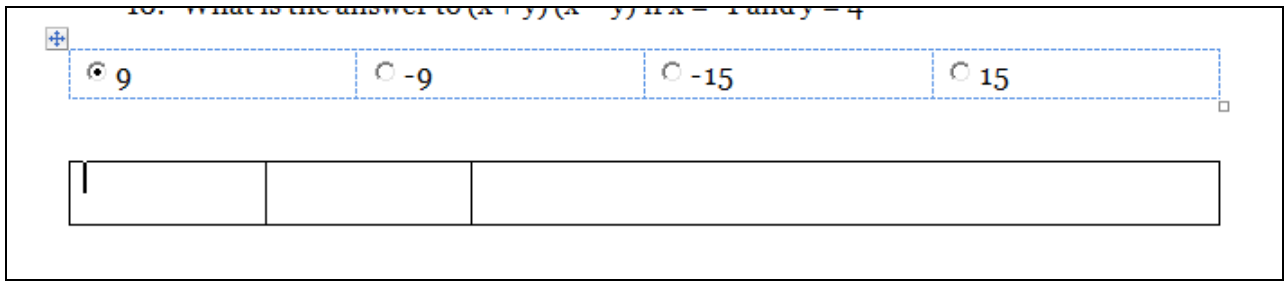

Figure 13.24 – Changing the Table Size and Turning Off the Border

Inserting and Modifying Command Buttons

We want to add a **Command Button** off the **Control panel** that will score the pretest, so with the insertion point in the first cell of the new table, choose the tool as shown in figure 13.25. After the command button appears in the cell, right click on the button and select Properties.

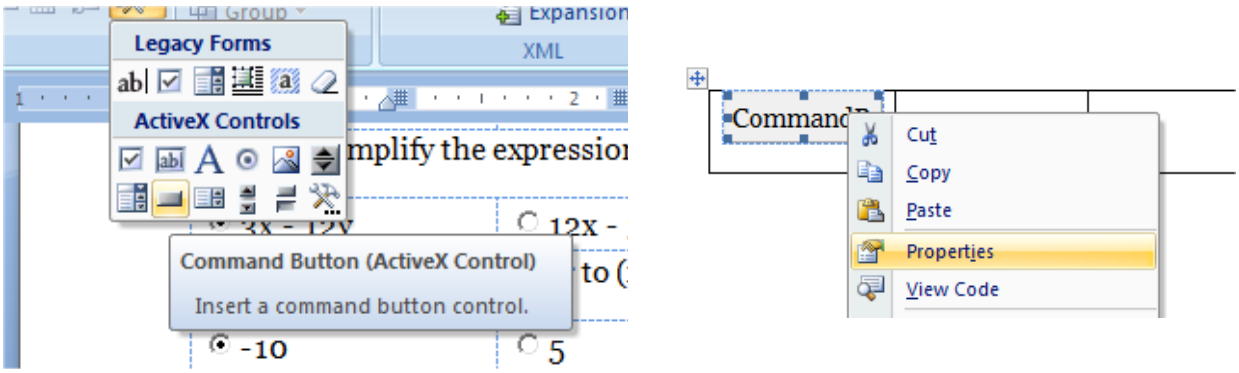

Figure 13.25 – The Command Button Tool **Figure 13.26 – The Properties Tool**

Next, we will change the **name** in the Properties pane to **cmdScore_quiz**. We use the cmd prefix for all command buttons. We type "Score" for the Caption.

Alphabetic	
(Name)	cmdScore_quiz
Caption	Score

After we finish the Score button, we will insert another Command Button in the next cell so we can clear our entries of we desire to do so. Highlight the second button, right click and select Properties.

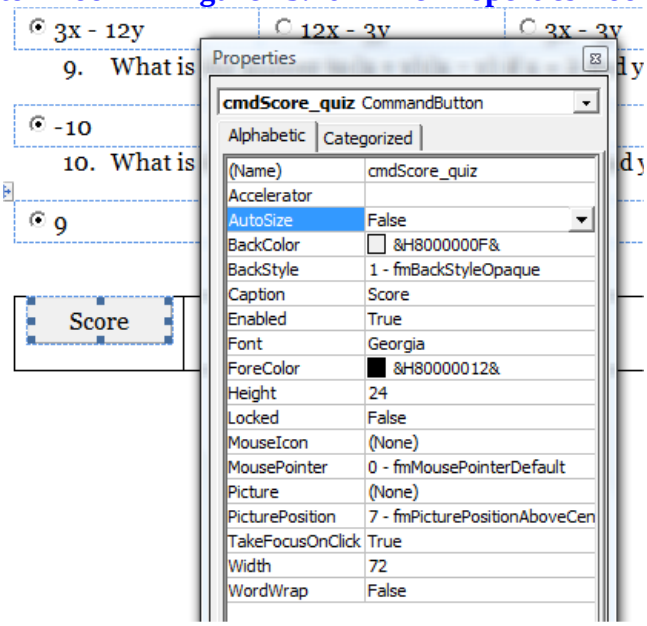

Figure 13.27 – The Properties Tool

For the Clear button, we will change the **name** in the Properties pane to **cmdClear**. We use the cmd prefix for all command buttons. We type "Clear" for the Caption.

Alphabetic	
(Name)	cmdClear
Caption	Clear

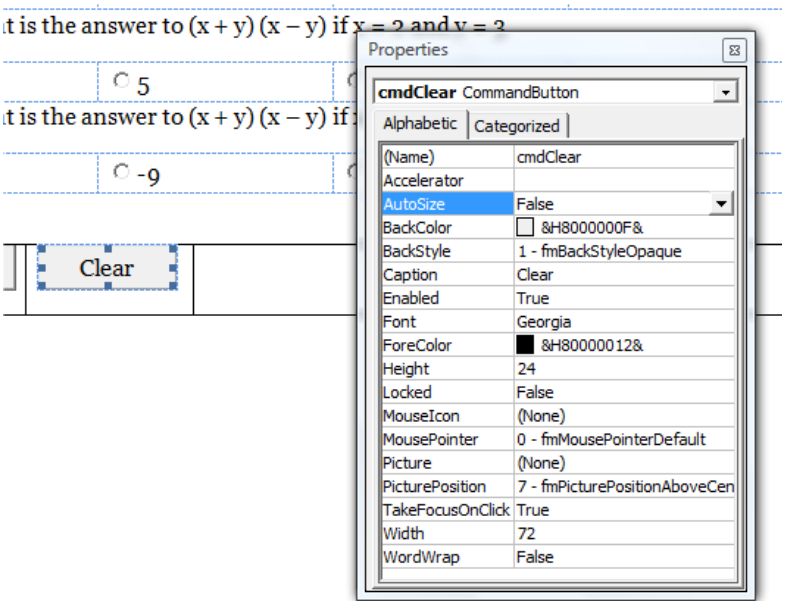

Figure 13.28 – The Properties Tool

Finally, we add a label in the last cell of the last table by picking the Legacy Tools tool on the Control panel under the Insert tab as shown in figure 13.29. Then highlight the label in the cell and choose the Properties tool on the Control Toolbox.

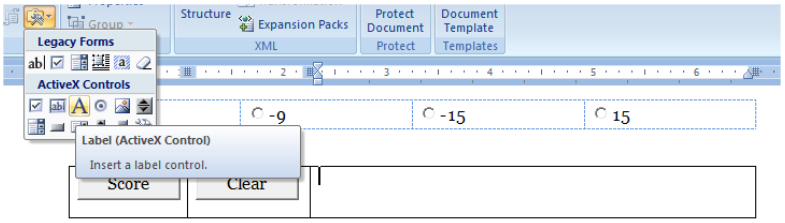

Figure 13.29 - The Label Tool

Name the label, lblErrors and make the height 18 and the width 230. The BorderStyle should be set to 0.

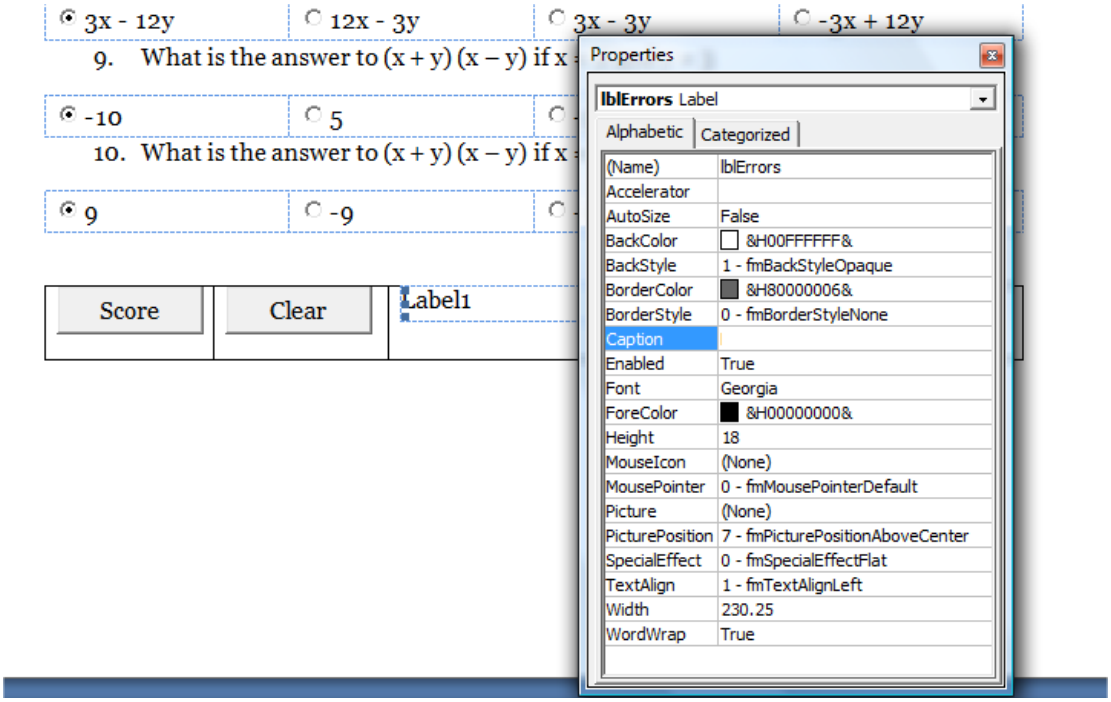

Figure 13.30 – The Properties Tool for the Label

Saving the Document

To save the document, click on the **Save** tool on the **Quick Access** toolbar which is located at the top left of the program. The Save As window will appear in the display. We can save the file into a folder we have already created.

Save the document with a description, that tells us what the document is about without having to open the file. Typing our name followed by the words "vba quiz" is a good choice. Make sure to save in a file type that enables macros in the "Save as Type" drop-menu. Once we have typed in the file name, click on the Save button and our document will be saved (Figure 13.31).

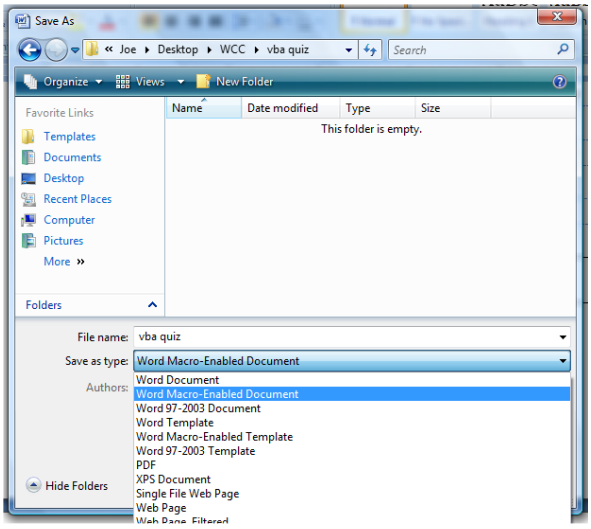

Figure 13.31 – Save the Document

Our document will be one page long as shown in figure 13.32.

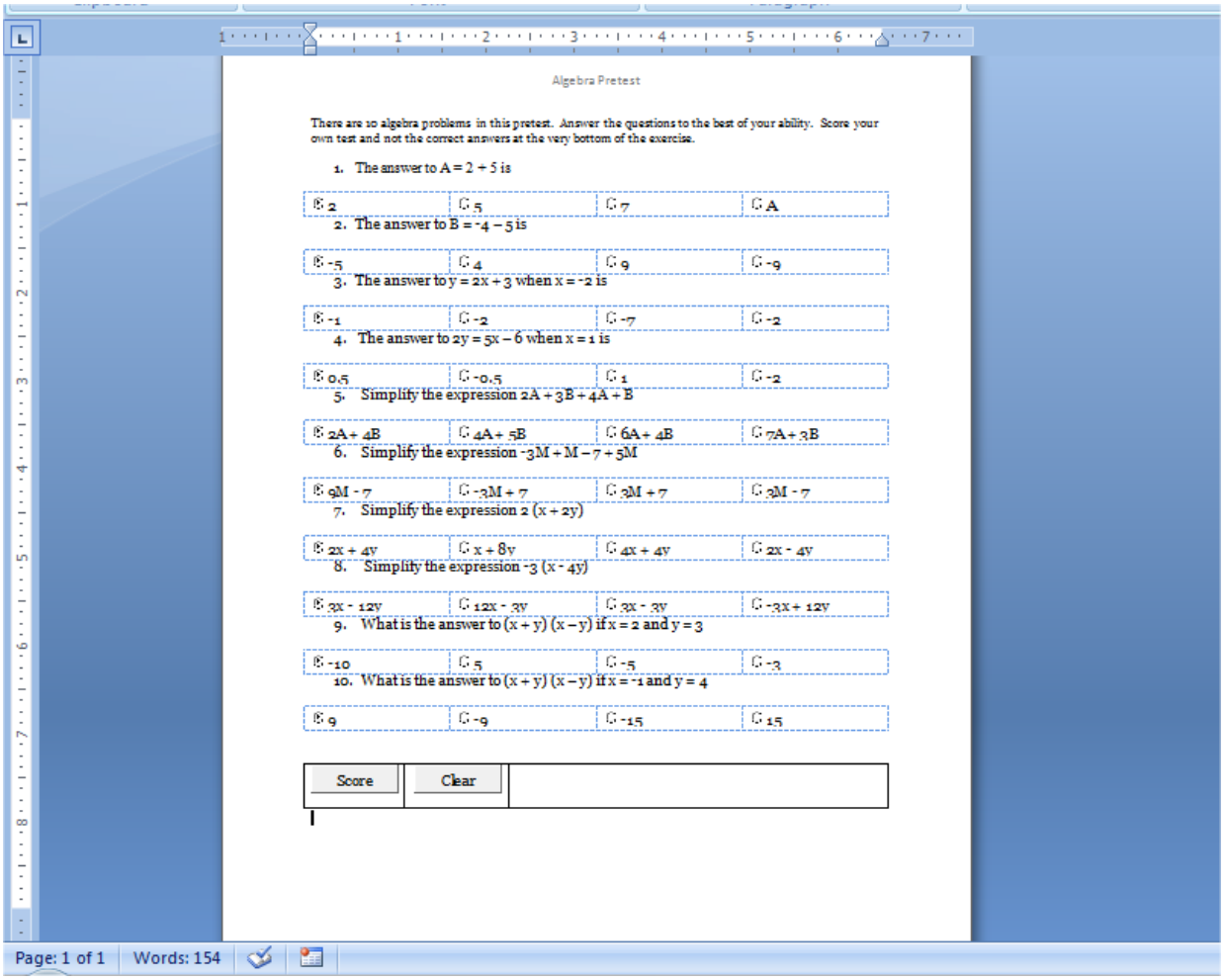

Figure 13.32 – The Finished VBA Document

Programming a Clear Button

When we double click on the Clear Command Button, we enter the Visual Basic Application program and we see:

Private Sub cmdClear_Click()

End Sub

After the first line, **Private Sub cmdClear_Click(),** we will type the code to clear all of the option buttons and the label.

To clear the option buttons containing the user input, we will set the option buttons value

property to false by using the equal sign "=".This makes the property equal to false as a default. We do this for 40 option buttons. For the first option button, opt1A this would be:

```
opt1A.Value = False
```

To clear the single label containing the list of errors, we will set the label caption value property to null ("") by using the equal sign "=".This makes the property equal to nothing as a default. For the label this would be:

```
lblErrors.Caption = ""
```

Programming the Score Button

When we double click on the Score Command Button, we enter the Visual Basic Application program and we see:

```
Private Sub cmdScore_quiz_Click()

End Sub
```

Adding Comments in VBA to Relay Programming Information

The comments we place in the program will inform the individual opening and reading the code. We place a single quote character (') in front of a comment and when the code is compiled, the comments are ignored. An example is:

```
' Declare variables
```

Declaring Variables in a Program with the Dimension Statement

When we are going to use a number, text string or object that may change throughout the life of the code, we create a variable to hold the value of that changing entity. In Visual Basic, the dimension or dim statement is one of the ways to declare a variable at the script of procedure level. The other two ways are the Private and Public statements, which we will use in later chapters.

In our program, we will declare a variable to enable us to capture the score for each correct answer, to total the score, to capture a statement and to report errors in the user choices. Type the following lines of code after the comment.

```
' Declare variables

Dim A1 As Double
```

```
Dim A2 As Double
Dim A3 As Double
Dim A4 As Double
Dim A5 As Double
Dim A6 As Double
Dim A7 As Double
Dim A8 As Double
Dim A9 As Double
Dim A10 As Double
Dim Score As Double
Dim Com As String
Dim Err As String
```

When selecting variable names, they should be a word or a phrase without spaces that represents the value that the variable contains. If we want to hold a value of one's date of birth, we can call the variable, DateofBirth. The keywords Date and Birth are in sentence case with the first letter capitalized. There are no spaces in the name. Some programmers use the underscore character (_) to separate words in phrases. This is acceptable, but a double underscore (__) can cause errors if we do not detect the repeated character.

Assigning Values to Variables

After we declare the variables and before we start to score the answers, we will assign the variables the value of zero or nil. Type the following code right below the declared variables

```
' set variables

A1 = 0
A2 = 0
A3 = 0
A4 = 0
A5 = 0
A6 = 0
A7 = 0
A8 = 0
A9 = 0
A10 = 0
Score = 0
Com = nil
Err = nil
```

Testing a Case with the If -Then Function

Whenever we are confronted with making a choice between two or more options in a computer program, then the **if-then** function becomes a very popular solution to this challenge. The **if-then**

function will execute the statements within the then section of the **if-then** expression when the logical test is true. The **if-then** function also will execute the else section of the **if-then** expression when the logical test is false.

The **if-then** function is arranged to work in a more complex fashion than other Visual Basic tools. If is initially, and after that an expression containing the logical test is written right after the **if**. The logical expression tests for a true or false response. In this program, the test is whether the logical value is true. If the answer is true, then the variable **A1** is assigned the value of 10. If the answer is false, then the variable **A1** remains a zero as assigned earlier in the program.

So type the following expression in the routine:

```
' check test

If opt1C.Value = True Then A1 = 10 Else Err = "1C"
```

In our first **if-then** statement, we are going to use the **else** section of the function. If the person taking the pretest gets the answer wrong, the correct answer will be saved to the variable **Err** and reported in the label at the when the user presses the Score command button. In this program, we get to practice our first **if-then** statements another nine times. Type the following lines of code to test the answers for the other nine questions.

```
If opt2D.Value = True Then A2 = 10 Else Err = Err & " 2D"
If opt3A.Value = True Then A3 = 10 Else Err = Err & " 3A"
If opt4B.Value = True Then A4 = 10 Else Err = Err & " 4B"
If opt5C.Value = True Then A5 = 10 Else Err = Err & " 5C"
If opt6D.Value = True Then A6 = 10 Else Err = Err & " 6D"
If opt7A.Value = True Then A7 = 10 Else Err = Err & " 7A"
If opt8D.Value = True Then A8 = 10 Else Err = Err & " 8D"
If opt9C.Value = True Then A9 = 10 Else Err = Err & " 9C"
If opt10C.Value = True Then A10 = 10 Else Err = Err & " 10C"
```

Adding in Visual Basic

The first arithmetic function we will address is one of the most common, the adding function which is displayed by the icon +. The addition function allows us to add two or more numbers. The values of the numbers can be whole like 1,2,3… or decimals, positive or negative. Remember we can have more than two numbers like (2 + 3 + 7 + 4).

In this program, we will add the ten variables that are holding the numbers. Type the following code:

```
Score = A1 + A2 + A3 + A4 + A5 + A6 + A7 + A8 + A9 + A10
```

The variable **score** will equal the sum of the value in variables **A1** through **A10**.

Using a Message box in Visual Basic

The function MsgBox will launch a message box in Windows. The text or as programmers state "string" will be enclosed in quotes. Before we launch the message box, we use an if-then-else statement to assign the phrase "excellent work" to the message or assign the correct answers to the error statement. Type the following code:

```
If Score = 100 Then Com = "Excellent work." Else Err = "Answers: " & Err
```

When typing the message box, we have the string "Your score is " & Score & "." & com concatenated with the & sign. We can use multiple & signs to add text strings and variables containing text together. Type the following code.

```
MsgBox "Your score is " & Score & ". " & Com
```

Using a Label in Visual Basic

Remember when we made the label earlier we left the caption blank. Therefore, the last Visual Basic code will set the caption for lblErrors to the string in variable Err. Type the following code.

```
lblErrors.Caption = Err
```

Written below is the entire program for the Algebra pretest.

```
Private Sub cmdClear_Click()

opt1A.Value = False
opt1B.Value = False
opt1C.Value = False
opt1D.Value = False
opt2A.Value = False
opt2B.Value = False
opt2C.Value = False
opt2D.Value = False
opt3A.Value = False
opt3B.Value = False
opt3C.Value = False
opt3D.Value = False
opt4A.Value = False
opt4B.Value = False
opt4C.Value = False
opt4D.Value = False
opt5A.Value = False
opt5B.Value = False
```

```
opt5C.Value = False
opt5D.Value = False
opt6A.Value = False
opt6B.Value = False
opt6C.Value = False
opt6D.Value = False
opt7A.Value = False
opt7B.Value = False
opt7C.Value = False
opt7D.Value = False
opt8A.Value = False
opt8B.Value = False
opt8C.Value = False
opt8D.Value = False
opt9A.Value = False
opt9B.Value = False
opt9C.Value = False
opt9D.Value = False
opt10A.Value = False
opt10B.Value = False
opt10C.Value = False
opt10D.Value = False
lblErrors.Caption = ""

End Sub
Private Sub cmdScore_quiz_Click()

' Declare variables

Dim A1 As Double
Dim A2 As Double
Dim A3 As Double
Dim A4 As Double
Dim A5 As Double
Dim A6 As Double
Dim A7 As Double
Dim A8 As Double
Dim A9 As Double
Dim A10 As Double
Dim Score As Double
Dim Com As String
Dim Err As String

' set variables

A1 = 0
A2 = 0
```

```
A3 = 0
A4 = 0
A5 = 0
A6 = 0
A7 = 0
A8 = 0
A9 = 0
A10 = 0
Score = 0
Com = nil
Err = nil

' check test

If opt1C.Value = True Then A1 = 10 Else Err = "1C"
If opt2D.Value = True Then A2 = 10 Else Err = Err & " 2D"
If opt3A.Value = True Then A3 = 10 Else Err = Err & " 3A"
If opt4B.Value = True Then A4 = 10 Else Err = Err & " 4B"
If opt5C.Value = True Then A5 = 10 Else Err = Err & " 5C"
If opt6D.Value = True Then A6 = 10 Else Err = Err & " 6D"
If opt7A.Value = True Then A7 = 10 Else Err = Err & " 7A"
If opt8D.Value = True Then A8 = 10 Else Err = Err & " 8D"
If opt9C.Value = True Then A9 = 10 Else Err = Err & " 9C"
If opt10C.Value = True Then A10 = 10 Else Err = Err & " 10C"

Score = A1 + A2 + A3 + A4 + A5 + A6 + A7 + A8 + A9 + A10

If Score = 100 Then Com = "Excellent work." Else Err = "Answers: " & Err

MsgBox "Your score is " & Score & ". " & Com

lblErrors.Caption = Err

End Sub
```

After we save the file, we can release the pretest onto the school server and students can open the file on their personal computer across the network, take the test, and view their score and see the correct answers after they press the score command button. We can practice the skills learned in this lesson by making tens to hundreds of training documents that are challenging and where it is easy for individuals to learn.

*** World Class CAD Challenge 30-12 * - Create a simple test using VBA that has ten questions. Use options buttons or check boxes to record the user's responses. Have two command buttons, one to score the pretest and the other to clear the responses. Have a label to show incorrect responses. Repeat this exercise at least three more times to maintain your World Class ranking.**