Loops might be the most powerful tool in a programmer's arsenal.  They provide us with a way to consolidate long, drawn out processes into a few lines of code.  There are many different types of loops that are used in different situations.  One of the most popular loops is a do... while loop.  The do... while loop will repeat (or iterate) a line or lines of code while a condition is met that is defined by the programmer.  Sometimes a loop will only go through one iteration, other times it will go through millions.  What are some of the other types of loops?

There is the While Loop and the For Loop which are both popularly used in programming. The While Loop continues until a condition is met such as when a Boolean statement becomes false creating an exit from repetitive process. Boolean statements allow for the ability to test conditions on a true or false premise. The For Loop for instance can be used to increment or decrement a number every time that a loop is passed through. The For Loop would be considered a pretest loop because the condition is taken into account before any code is processed within the loop. The While Loop is considered a post-test loop as it executes lines of code before checking the condition and the same can be said for the Do While loop.

Declare Number As Integer int Number = 2  
  
Set Number = 2 Loop   
   
Repeat   
  
    Write Number  
  
    Set Number = Number - 1  
  
Until Number = 0  
  
1. Is this loop a pre-test loop or a post-test loop?

This kind of loop would have to be considered a post-test loop as the condition for whether the number is equal to 0 is not checked until after the number in the loop is set.   
  
2. List the statements in the body of the loop

The statements in the body of the loop would read:

2

1

0  
  
3. What is the test condition for this loop?

The test condition is to have the loop end if the integer variable of 2 reaches 0.

* + Class,

Part of writing programs is to define the variable types used to save the information in a program. There are a standard number of variable types available to a developer with moth program languages. See the C# types below: One decision of the programmer is to select the correct variable type for the data used in the program. If you had the following data, what would be your best choice for a variable type?  
  
   
1. Age of a client  
  
2. Name of the client   
  
3. Phone number of the client  
  
4. Social Security Number  
  
5. Credit limit of the client

The variable type I would use for age would most likely be a double as it can hold large numbers and does not need to hold any floating point value information. I would choose a string variable type to hold the name of a client, because we would be dealing with string literals. It’s also possible to use the char datatype to hold social security numbers. For a phone number the string variable data type would work as there wouldn’t be any issues with formatting. The integer data type can be used for storing someone’s credit limit as it can store up to 4 bytes of data with great range.

This chapter brought up an interesting question.  What is the difference between a correct programming solution and a good programming solution?  There can be quite a few different correct programming solutions for a given problem, but not all of them are good solutions.  A correct programming solution is a solution that solves the problem, but a good programming solution solves the problem coherently.  It allows for the next programmer to come along and read the code and know exactly what the first programmer was writing.  It is always best to not only think about solving the problem at hand, but also to do it in a way that others can understand.  Using good programming solutions as a professional programmer will help tremendously during collaborative efforts, where many programmers are working together achieve a common goal.

I think it’s very important to make a program work accurately, but also important to structure code so that it simplifies the editing process for future programmers who need to make alterations. Being able to solve a problem with less lines of code or with blocks of code or methods that perform independent operation in a program keeps things clean and easy to read for everyone involved with a project. It takes a great collaborative effort and although a solution to a problem may work it is also possible that it can take up much more space in a program than needed to get the job done.

While loops are often used for input validation, as we had learned in week 2. It is easy to use a while loop to keep garbage input  from entering your program. A simple example of a while loop used for input validation is shown below.  
  
Display "Please enter a number between 1 and 100"  
  
Get number  
  
While (number < 1) AND (number > 100)  
  
Display "Please choose a number between 1 and 100"  
  
Get number  
  
End While  
  
This loop makes sure that the user doesn't input a number lower than 1 or higher than 100.

I can see from the Boolean statement of if a number is greater than 1 and the number is less than 100 to continuously loop. I also see that if this condition becomes false or a number is entered that is less than 1 or over 100 to terminate the whole loop. I know this concept will come in handy for many future programing assignments, especially when working with Visual Logic in this particular class.

What is an iteration structure? What problem does a repetition structure solve? Provide an example of a process in which a repetition structure is appropriately used

An iteration structure would be one that specifies a certain block of code to run until a condition is met whether true or false. Sometimes an iteration structure can be setup to run infinitely without stopping. A repetition or an iteration structure can solve the task of store specific values into an array and retrieving those values at a later time or perform a number of tasks multiple times rather than having redundant code to complete the job. A good example of a repetition structure would be one where the user is asked to enter their first name and if it did not match the name stored in a string variable the loop would continue, else if the name did match than the loop would end.

Each task must include the following:

* A conditional step
* Some form of iteration

Example tasks include the following:

* Entering a number of items and calculating sales tax on a sale; include a step offering a warranty for each item
* Converting from Fahrenheit to Celsius or the reverse over temperatures for several days
* Figuring out a total bulk sale price based on price per unit and number of units

These items may build on the tasks used in your Week Two assignment or may be unique.

**Obtain** approval from your instructor for your items by sending a Private Message.

**Create** a 1/2-page Word document for each of the tasks. Each document should include:

* A brief description of the task
* The pseudocode associated with the task. Base the pseudocode on the examples provided in Chs. 4, 5, and 6 of Prelude to Programming.

**Create** Visual Logic® files to execute each of the tasks.