VBA Principles

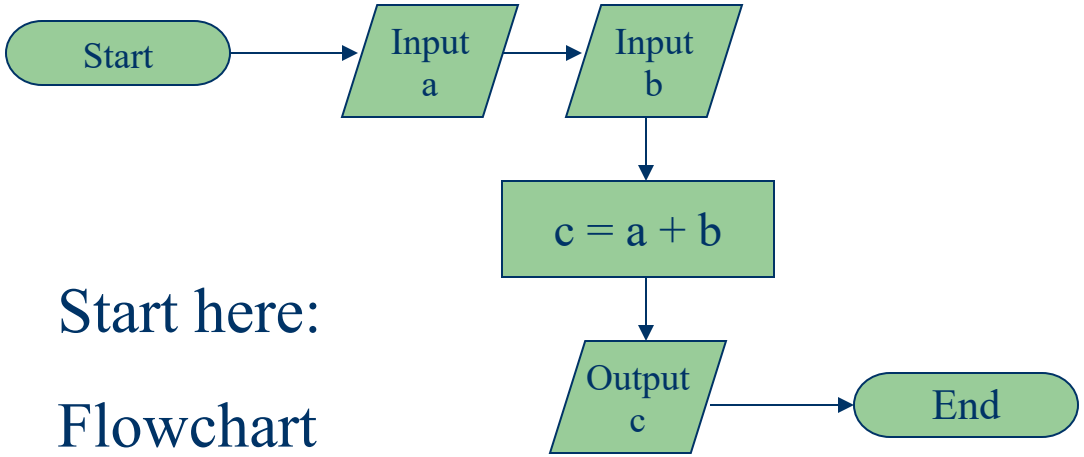
CEE3804: Computer Applications for Civil and Environmental Engineers

Objectives

- Develop a simple set of programs in VBA
- Learn to execute a macro from within a worksheet
- Introduction to loop structures

Before You Program

- Develop a strategy to solve the problem
- Develop pseudo-code
- Develop a flowchart (helps guide your thought process)



Start here:

Flowchart

Then code

- NOTE: most engineers do not to like to develop flowcharts
- This creates problems when someone else looks at your program
- Most programs do not work the first time, so a flowchart of pseudo-code can help you understand bad logic

Flowchart Symbols

- **Review them in Chapra's book (page 115)**

Symbol	Name	Function
Start	Terminal	Represents the beginning of end of a program
→	Flow lines	Represent the flow of logic
c = a + b	Process	Calculation of data manipulation
Input b	Input/Output	Represents input and output of data and information
If (a>b) → True; ↓ False	Decision	Represents a comparison or a condition with more than one path

Flowchart Symbols (cont.)

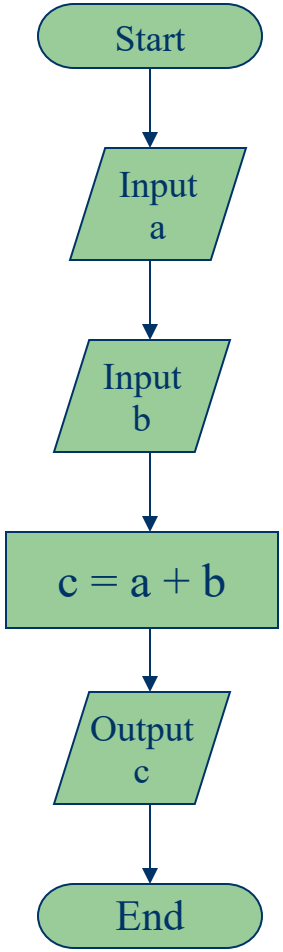
Symbol	Name	Function
	Count-controlled loop	Indicates the number of times a loop is excecuted
	Junction or connector	Represents a confluence of flow lines
	Off-page connector	Break or continuation to another page
	Stored data	Represents data storage in a database

A Simple VBA Program

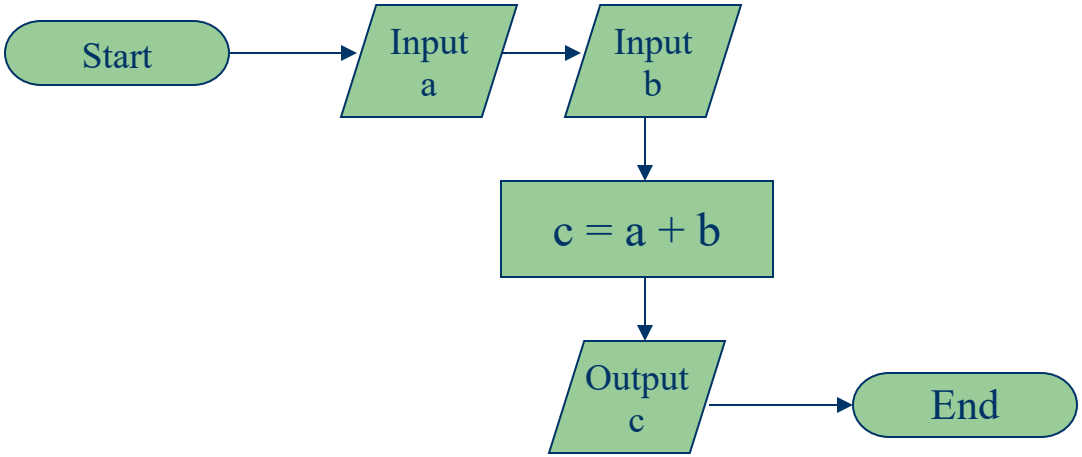
- Lets develop a simple program in VBA to calculate the sum of two numbers of a worksheet
- The program should retrieve two numbers from "sheet1" in a standard Excel worksheet
- The program should add the two numbers and return the result to the worksheet
- Place a "run" button in the worksheet to facilitate future execution of the subroutine

Program Flowchart

- **Here is a basic flowchart to build this program**

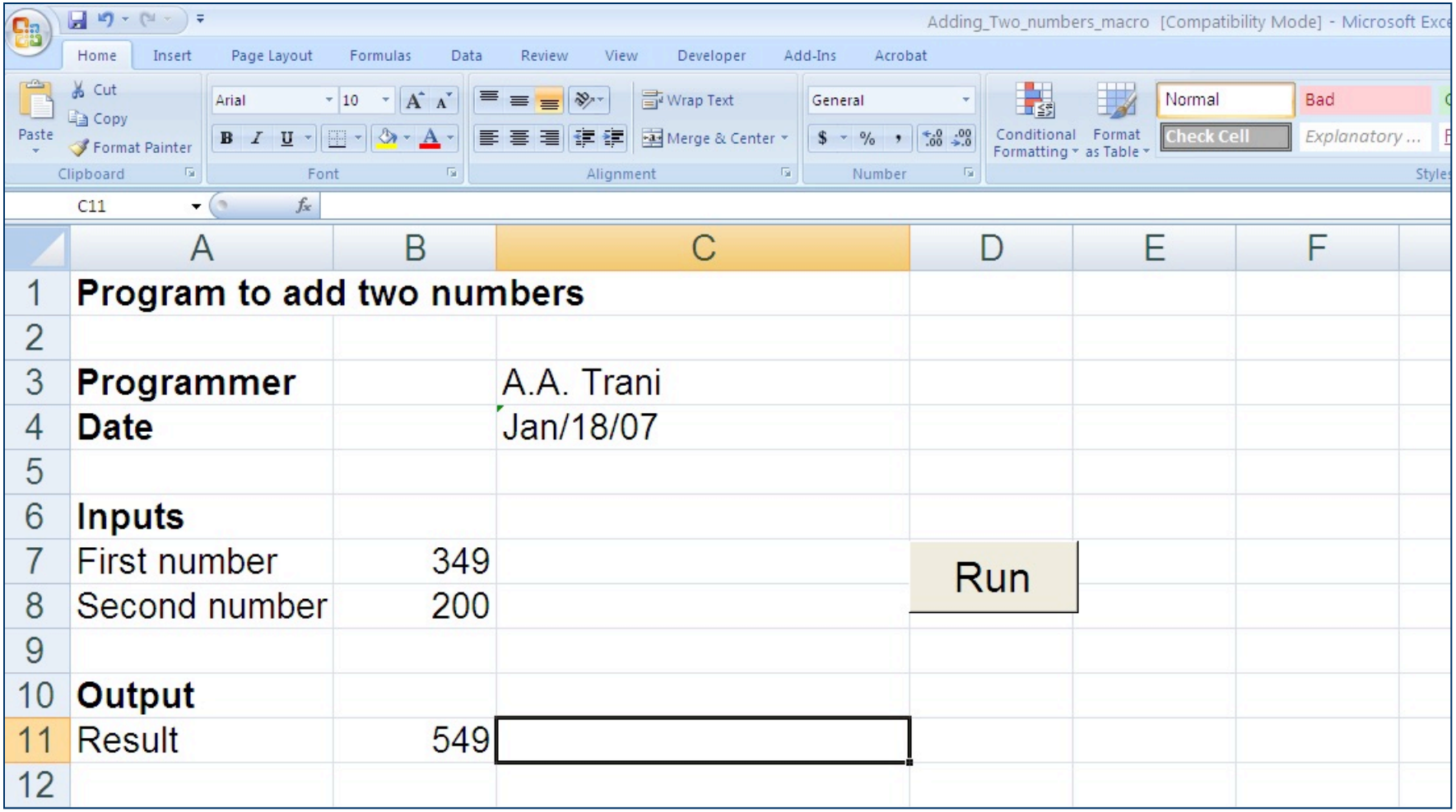


Version 1



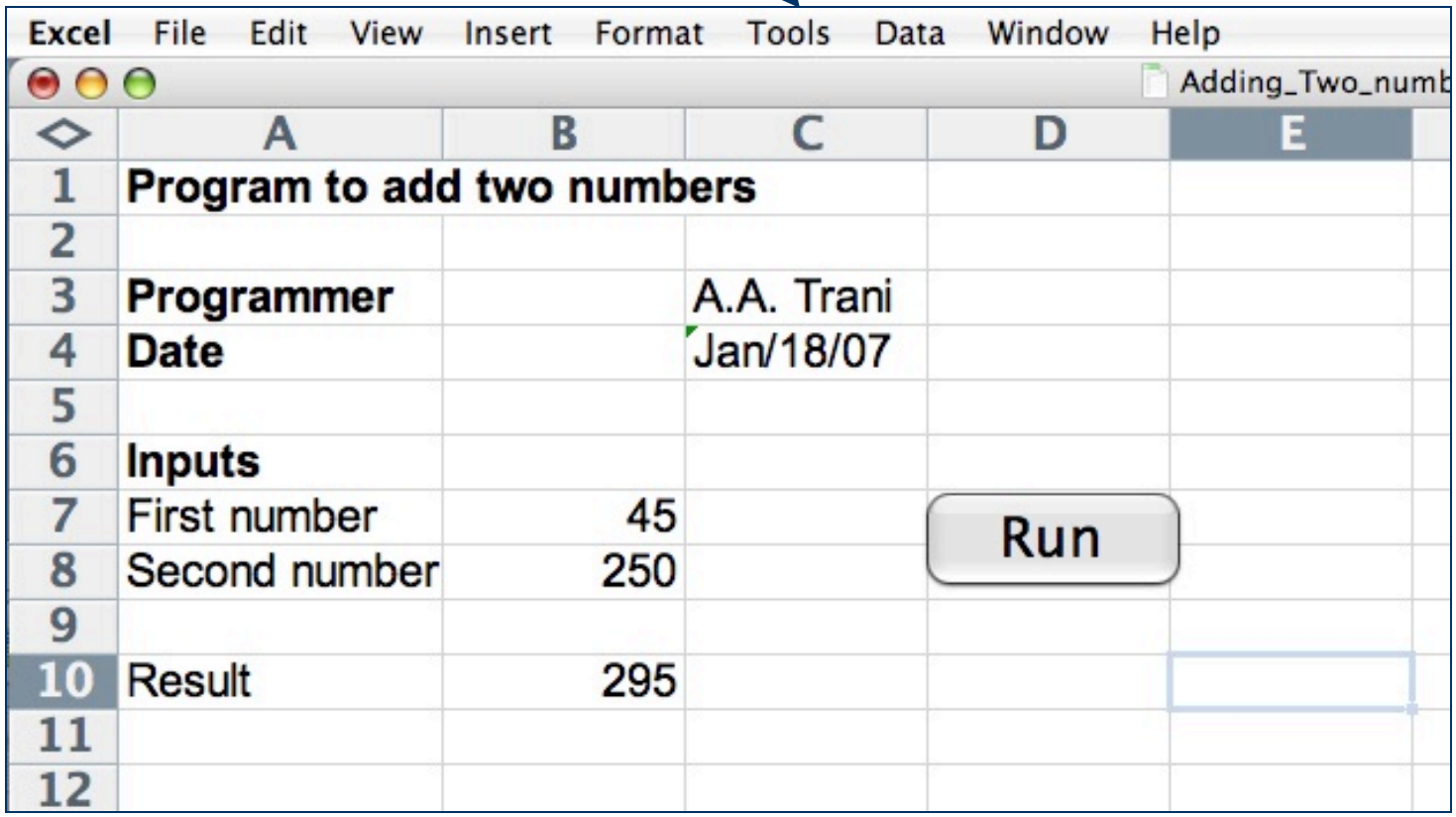
Version 2

Excel Worksheet (final program)



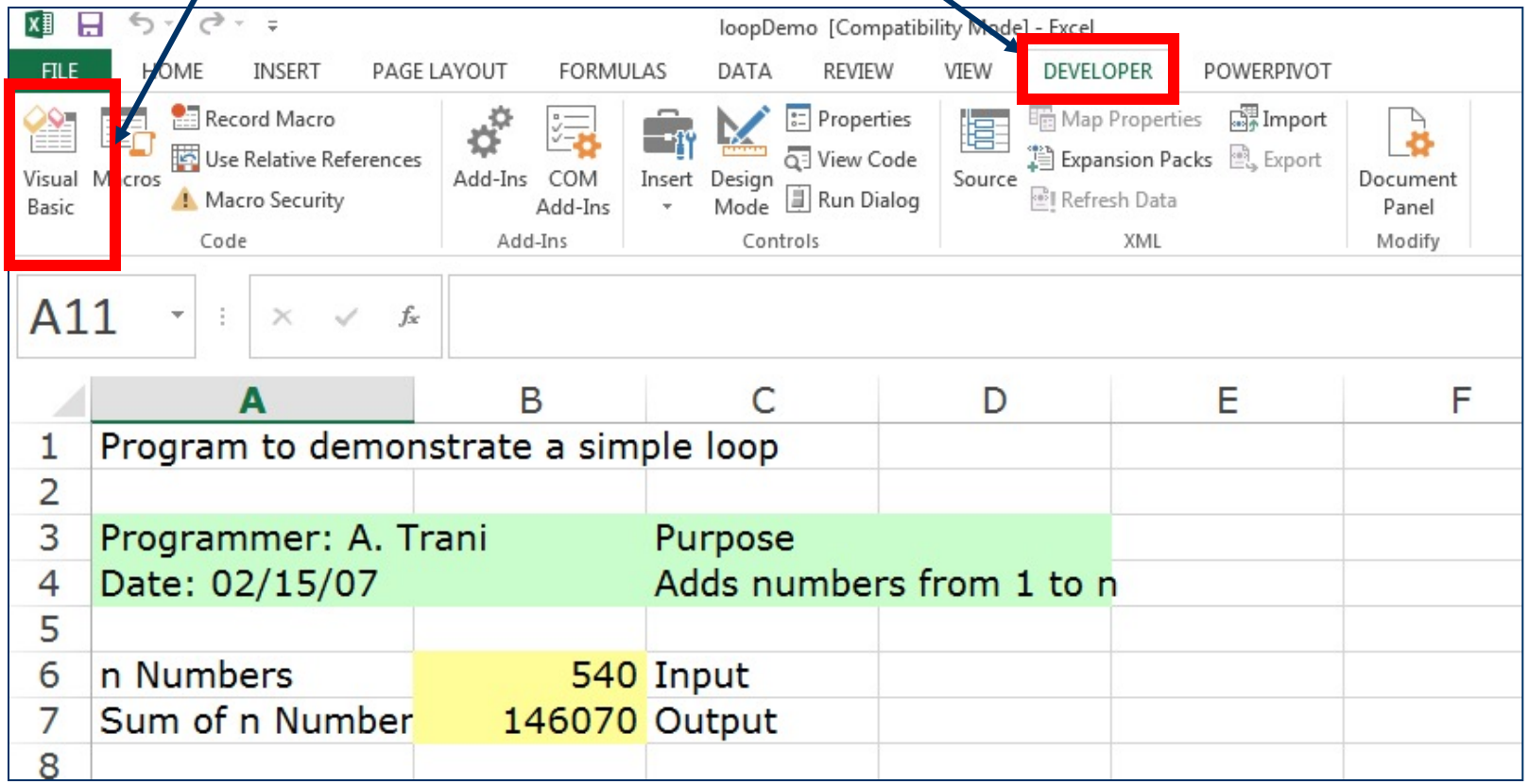
VBA or Macros in Old Excel Versions (including Mac 2011)

- Go to: Tools/Macro
- Then select VBA



VBA or Macros in Excel WIN 2013

- Go to : Developer Tab
- Then Click Visual Basic



VBA Code

Microsoft Visual Basic - Adding_Two_numbers_macro.xls

File Edit View Insert Format Debug Run Tools Add-Ins Window Help

Ln 18, Col 11

Project - VBAProject

VBAProject (Adding_Two
- Microsoft Excel Objects
  - Sheet1 (Sheet1)
  - Sheet2 (Sheet2)
  - Sheet3 (Sheet3)
  - ThisWorkbook
- Modules
  - Module1

Properties - Sheet1

Sheet1 Worksheet

Alphabetic | Categorized

(Name)	Sheet1
DisplayPageBreak	False
DisplayRightToLef	False
EnableAutoFilter	False
EnableCalculation	True
EnableFormatCon	True
EnableOutlining	False
EnablePivotTable	False
EnableSelection	0 - xlNoRestricti
Name	Sheet1
ScrollArea	
StandardWidth	8.14
Visible	-1 - xlSheetVisib

Adding_Two_numbers_macro.xls - Module1 (Code)

(General)

```
Sub Adder()

' Programmer : Toni Trani
' Date: 09/13/07

Sheets("Sheet1").Select


Range("B7").Select
a = ActiveCell.Value


Range("B8").Select
b = ActiveCell.Value

c = a + b

Range("B11").Select
ActiveCell.Value = c

End Sub
```

Name of subroutine

Go to "sheet1"

Select the value of cell B7 and assign it to variable "a"

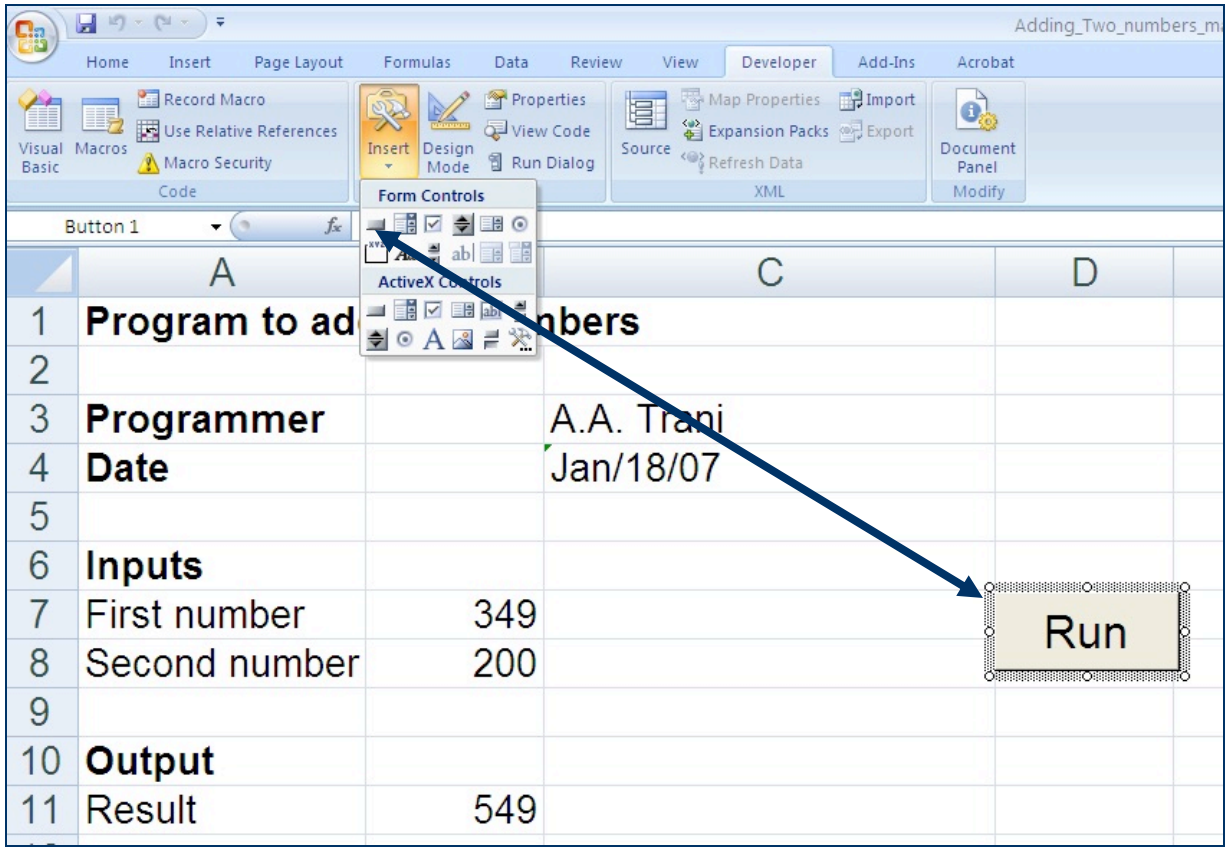
Select the value of cell B8 and assign it to variable "b"

Calculate the value of "c"

Select cell B8 and assign it to variable "c"

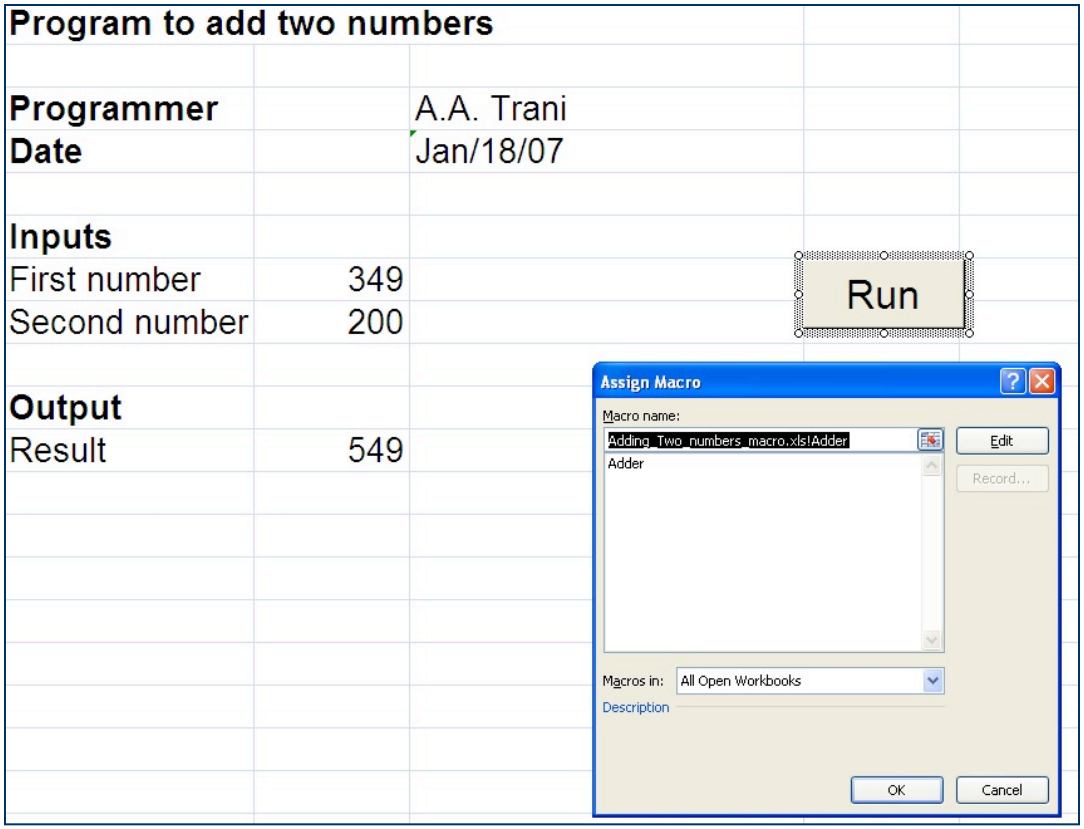
Adding the "Run" Button

- Adding the "run" button requires that you insert a control button and then assign a macro to subroutine "adder"



Assigning the Behavior to the "Run" Button

- Assign the macro "adder" from the list of available macros to button "Run"
- Right-click on the button to assign a behavior



Test Your Program

- Test the program to make sure it works
- Try variations of the program as well

Subroutine Adder1 with Input Box

- **This variation of the program creates an input box for the first number**

Microsoft Visual Basic - Adding_Two_numbers_inputBox.xls

Adding_Two_numbers_inputBox.xls - Module1 (Code)

```
Sub Adder1()

' Programmer: A. Trani
' Date: 02/13/07
' Purpose: add two numbers with an input box for first value

a = InputBox("Enter the first value to be added")

' Assign the value of a to a number (use the function VAL in VBA)

a = Val(a)

' Write the value of a to cell B7

Range("B7").Select
ActiveCell.Value = a

' retrieve the value of cell B8

Range("B8").Select
b = ActiveCell.Value

' calculate the output

c = a + b

' Assign the value of c to cell B10

Range("B10").Select
ActiveCell.Value = c

End Sub
```

String variable "a" is read using an input box

Returns the number contained in the string "a"

Excel Help for VAL Function

Excel Help

val

Search

Excel Developer Home > Visio > Visio 2007 Automation Reference > Visual Basic for Applications Language Reference > Visual Basic Language Reference > Functions

Val Function

Returns the numbers contained in a string as a numeric value of appropriate type.

Syntax

Val(*string*)

The required *string* argument is any valid string expression.

Remarks

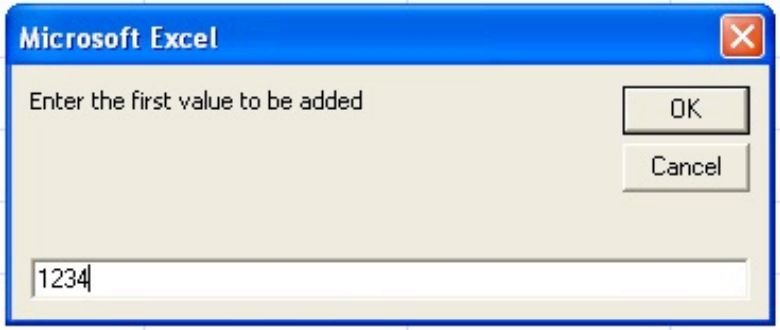
The **Val** function stops reading the string at the first character it can't recognize as part of a number. Symbols and characters that are often considered parts of numeric values, such as dollar signs and commas, are not recognized. However, the function recognizes the radix prefixes

Executing the New Program

Program to add two numbers

Programmer		A.A. Trani
Date		Jan/18/07
Inputs		
First number	67	
Second number	250	
Result	317	

Run

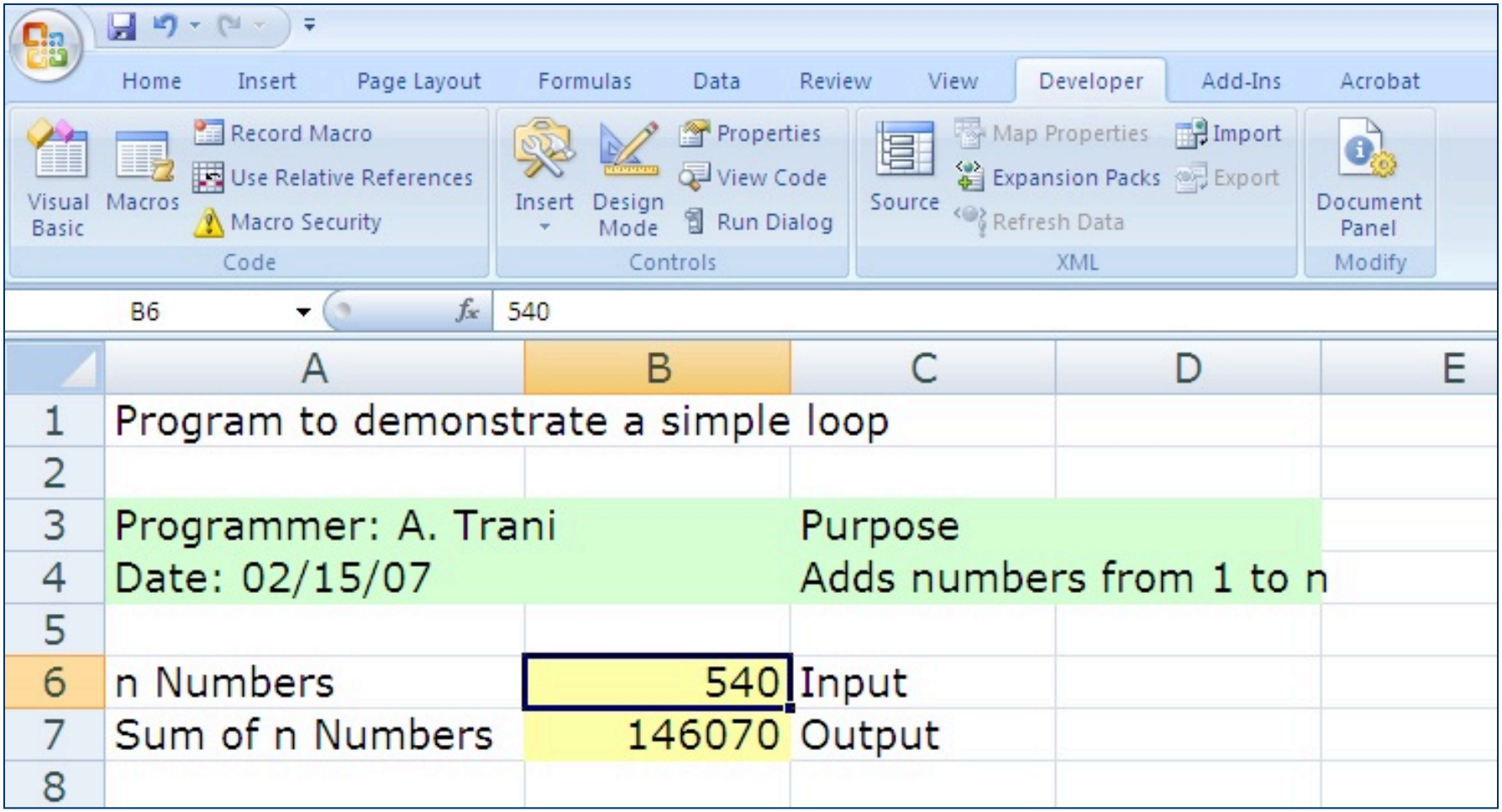


A Simple Program with a Loop

- Loops are natural ways to execute computations that require multiple iterations
- Loops can be conditioned or controlled by a counter

- Conditional loops - when some condition is used to exit the loop
- Counter controlled loops - when the number of passes in the loop is known

First Program with a Loop



The VBA Code Behind

Microsoft Visual Basic - loopDemo.xls - [Module1 (Code)]

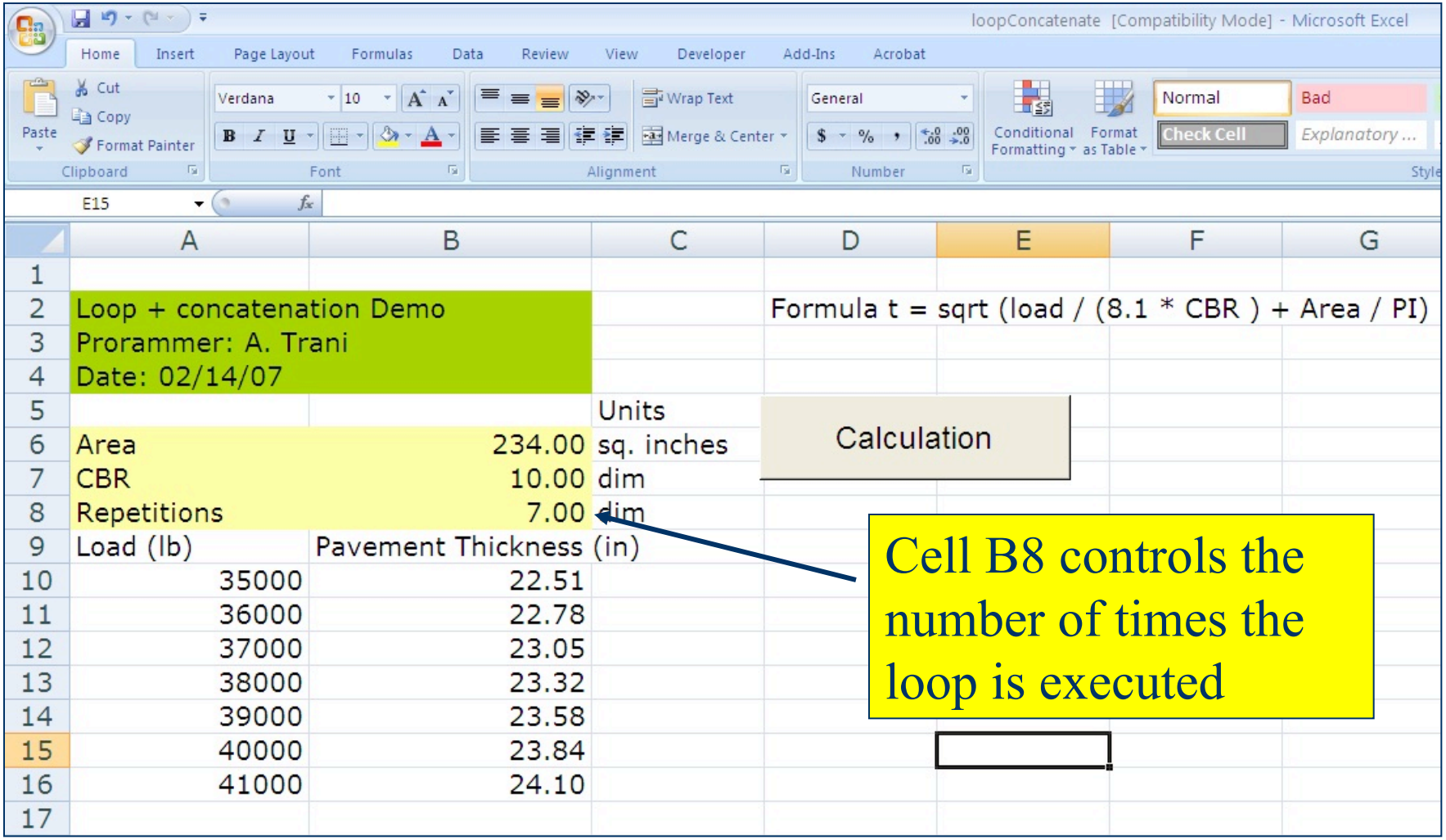
```
Sub Looptest()

' Programmer: T. Trani
' Date: 09/13/07
' Testing a loop with counter control
' Adds numbers from 1 to n
' Assigns the value of the calculation to variable x
' Writes the value of x in cell B7

' Extract the value of n

Sheets("Sheet1").Select
Range("B6").Select
n = ActiveCell.Value

' initialize the variable x to keep track of our calculation

x = 0

' start the calculation loop

For i = 1 To n
    x = x + i
Next i

' select cell B7 and write down the calculation of the loop

Range("B7").Select
ActiveCell.Value = x

End Sub
```

Loop code

A Loop with Concatenation Control

- The program in worksheet: loopConcatenate.xls offers a sample of a loop computation and the use of concatenation control to estimate pavement thicknesses
- The pavement thickness function created in previous classes in “called” by the VBA code

Worksheet Interface



The Code Behind the Worksheet

```
Sub LoopConcatenate()

' testing a loop with concatenation to control where do we write calculations
' in a workheet

' Programmer : A. Trani
' Date: 02/17/07

Pi = 3.1415

' retrieve values of constant parameters from cells b6 and b7

Sheets("Sheet1").Select

Range("b6").Select
area = ActiveCell.Value

Range("b7").Select
CBR = ActiveCell.Value

' retrieve the value of n from cell B8

Range("B8").Select
n = ActiveCell.Value

' start the loop to compute pavement thicknesses for n repetitions
```

Code (cont.)

```
' retrieve the value of n from cell B8

Range("B8").Select
n = ActiveCell.Value

' start the loop to compute pavement thicknesses for n repetitions

For i = 1 To n

    cellNumber = "A" & (i + 9)                  ' assign the cell to write load values
    Range(cellNumber).Select                    ' select cell assigned in previous step
    appliedLoad = 35000 + 1000 * (i - 1)        ' compute load (lb) at 1000 lb increments
    ActiveCell.Value = appliedLoad              ' assign computed load to cells A+ (n+9)

    ' calculate the pavement thickness

    thickness = Sqr(appliedLoad / (8.1 * CBR) + area / Pi)

    cellNumber = "B" & (i + 9)                   ' assign the cell to write pavement thickness values
    Range(cellNumber).Select                     ' select cell
    ActiveCell.Value = thickness                 ' write value of pavement thickness

Next i                                          ' next value of i

End Sub
```

Concatenation

Calculates pavement thickness

Try Other Refinements

- Currently the loop counter just overwrites the values of pavement thickness without erasing previous computation
- Try adding a line or two of code to erase the previous table of computations while executing the code
  - Use a statement such as :
  - Range(“A1:C5”).clear
  - This clears the range between cells A1 and C5
  - You can always clear a large range of cells so that your program will always work correctly

Simple Control Structure

- If-Then-Else is a powerful, yet simple control structure that could be used to make "branching" decision in a program
  - If (condition) then
    - Code 1 is executed
  - Elseif (condition 2) then
    - Code 2 is executed
  - Elseif (condition 3) then
    - Code 3 is executed
  - End if
- You can add as many else-if-then statements as needed in your program logic

Modification of Pavement Analysis Program (Loop with Concatenation)

- Suppose we would like to modify the previously created program to replace the value of California Bearing Ratio (CBR) for the type of soil used in construction
- Suppose we have a simple table with typical values of CBR as follows:

Material	Value of CBR (dim)
Crush stone	100
Sandy soil	25
Silty soil	13
Clay soil	7
Organic soil	4

Modification of Pavement Analysis Program (Loop with Concatenation)

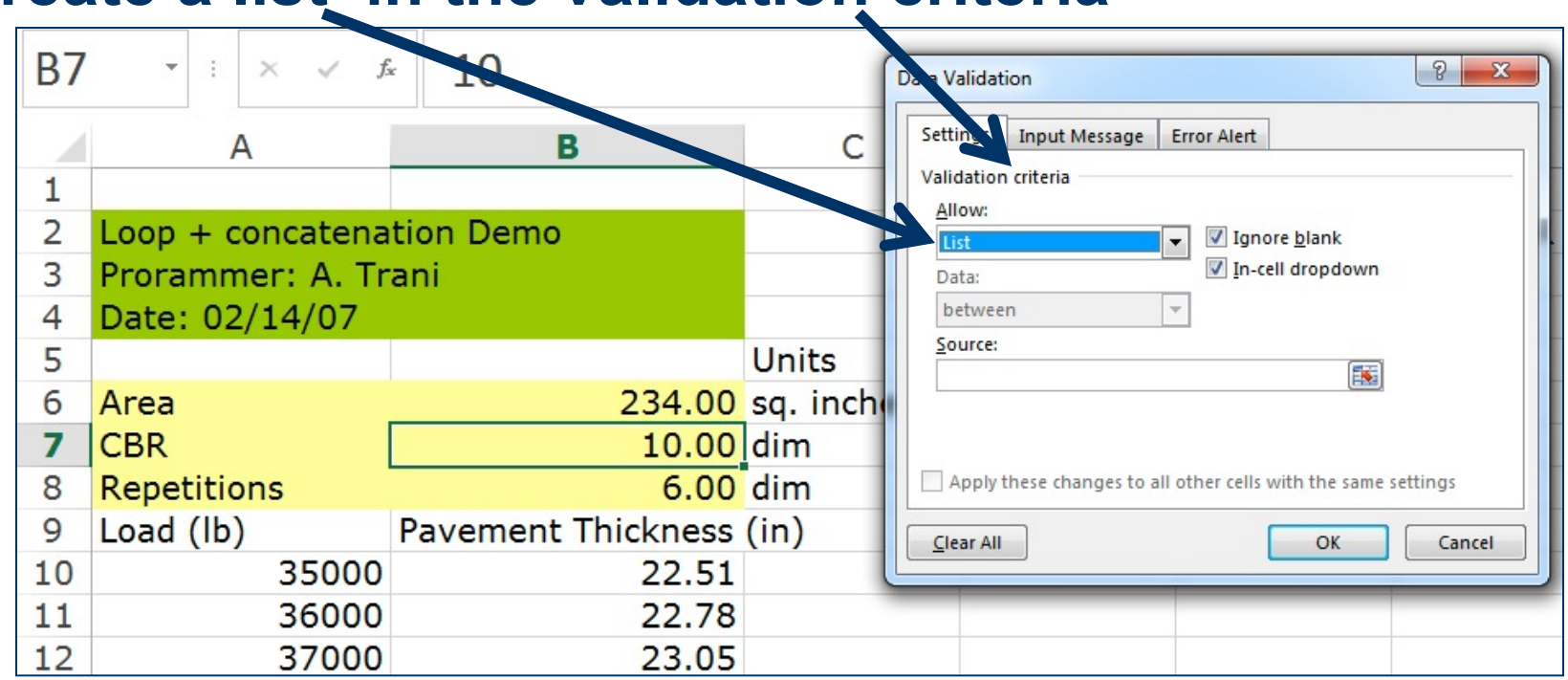
- **Recall the original spreadsheet**

	A	B	C	D	E	F	G
1							
2	Loop + concatenation Demo			Formula t = sqrt (load / (8.1 * CBR) + Area / PI)			
3	Prorammer: A. Trani						
4	Date: 02/14/07						
5			Units	Calculation			
6	Area	234.00	sq. inches				
7	CBR	10.00	dim				
8	Repetitions	6.00	dim				
9	Load (lb)	Pavement Thickness (in)					
10	35000	22.51					
11	36000	22.78					

Material	Value of CBR (dim)
Crush stone	100
Sandy soil	25
Silty soil	13
Clay soil	7
Organic soil	4

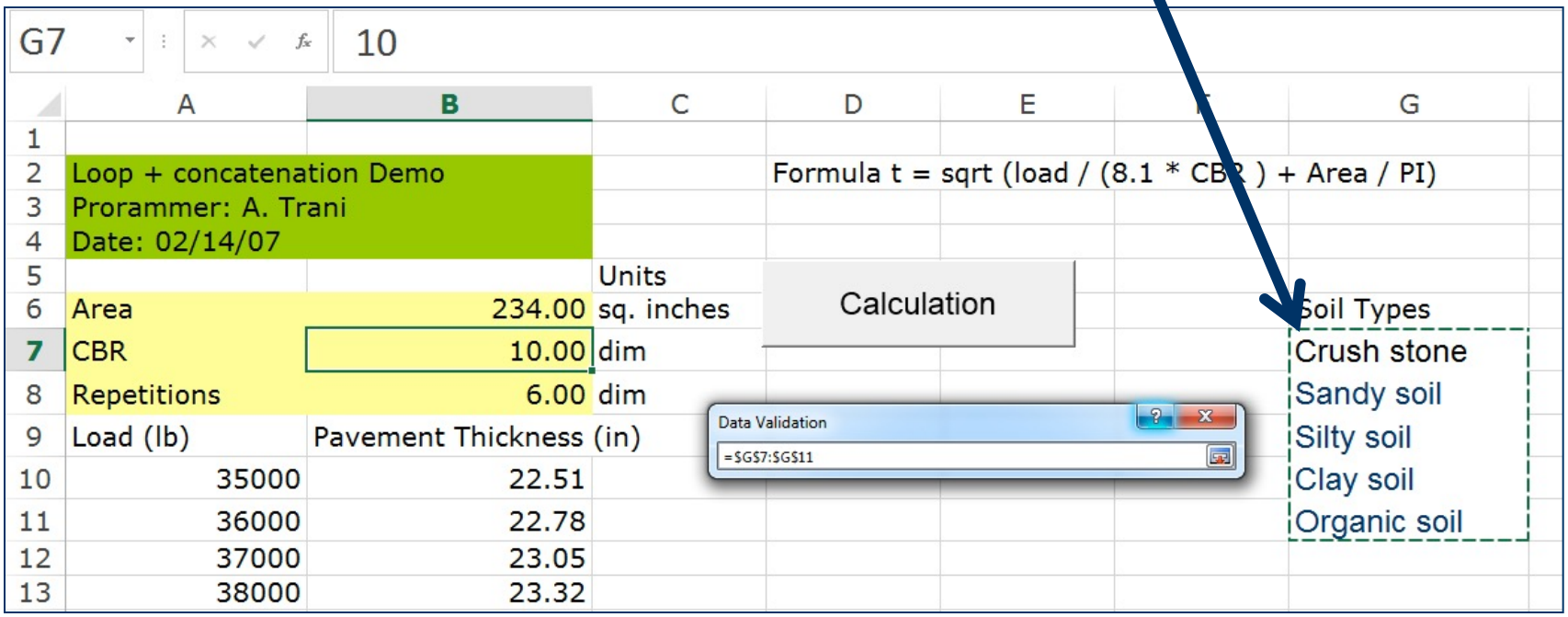
Modification of Pavement Analysis Program (Loop with Concatenation)

- Create a list of items to be selected by the user in cell B7
- Use the Data Validation in the Data Tab
- Create a list in the validation criteria



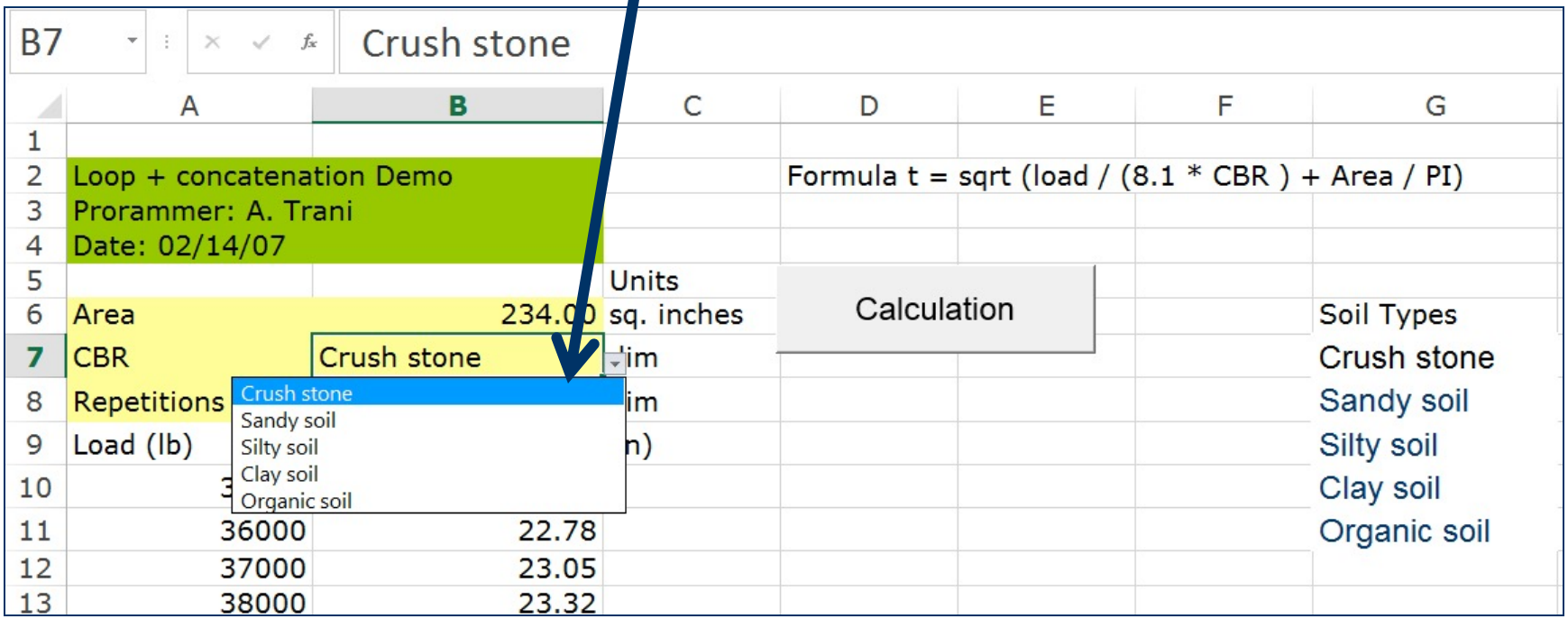
Modification of Pavement Analysis Program (Loop with Concatenation)

- The list of items can be enumerated in another section of the spreadsheet
- In this example I used cells G7:G11 to write the names of the types of soils to be selected



Modification of Pavement Analysis Program (Loop with Concatenation)

- **Once the cell is connected with data validation only the names of the soils types can be selected in that cell**



Modification of Pavement Analysis Program (Loop with Concatenation)

- **Complete the VBA code to assign the value of CBR for a selected soil type in cell B7**
- **Use the if-then-else statement construct**

```
Range("b6").Select
area = ActiveCell.Value

' In this section we assign the value of CBR for a type of soil selected

Range("b7").Select
soilType = ActiveCell.Value    ' Reads the type of soil selected

If soilType = "Crushed Stone" Then
    cbr = 100
ElseIf soilType = "Sandy Soil" Then
    cbr = 25
ElseIf soilType = "Silty Soil" Then
    cbr = 12
ElseIf soilType = "Clay Soil" Then
    cbr = 7
ElseIf soilType = "Organic Soil" Then
    cbr = 4
End If
```

Retrieves the content of cell B7

The If-Then-Else block assigns a value of cbr to the selected soil type