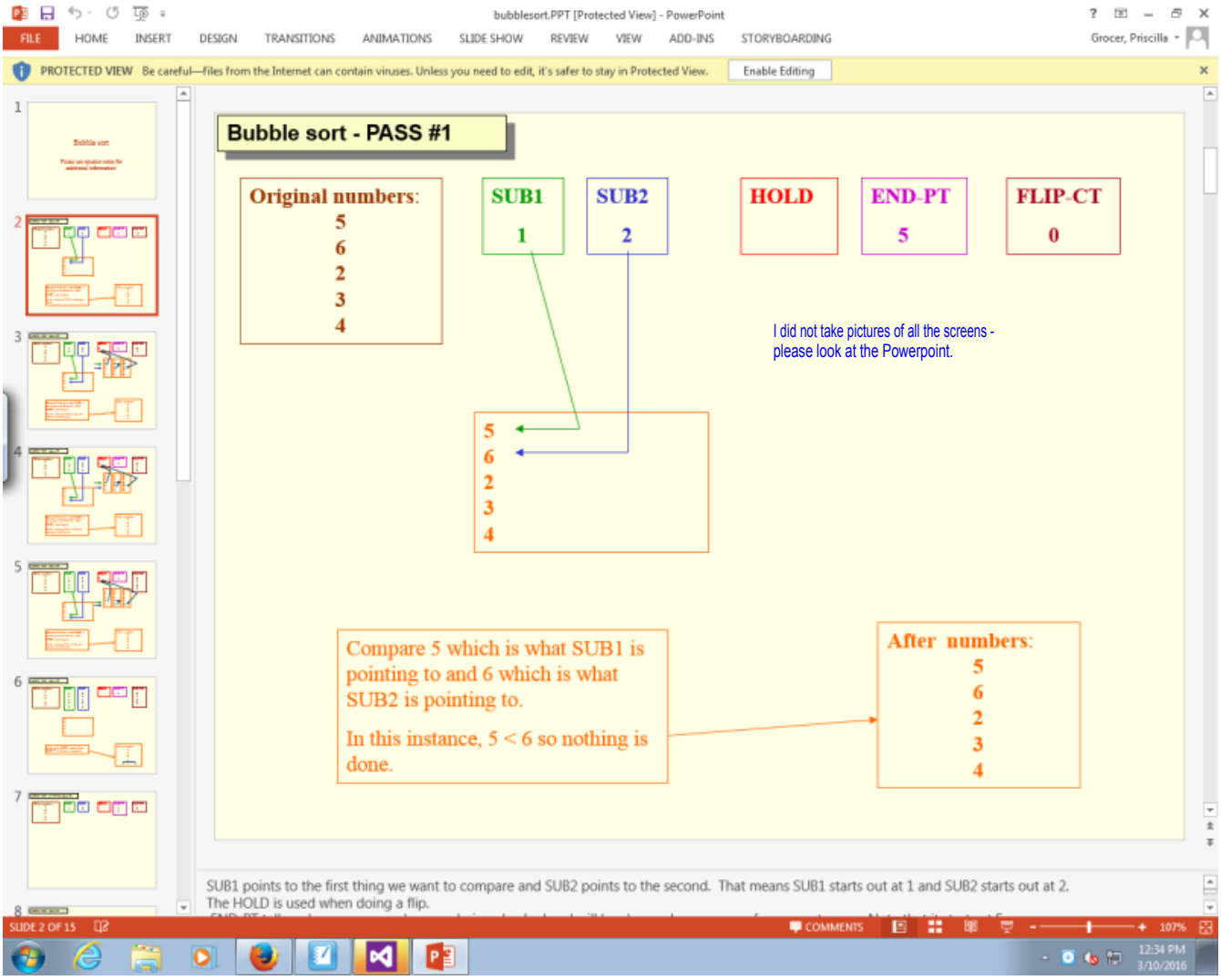
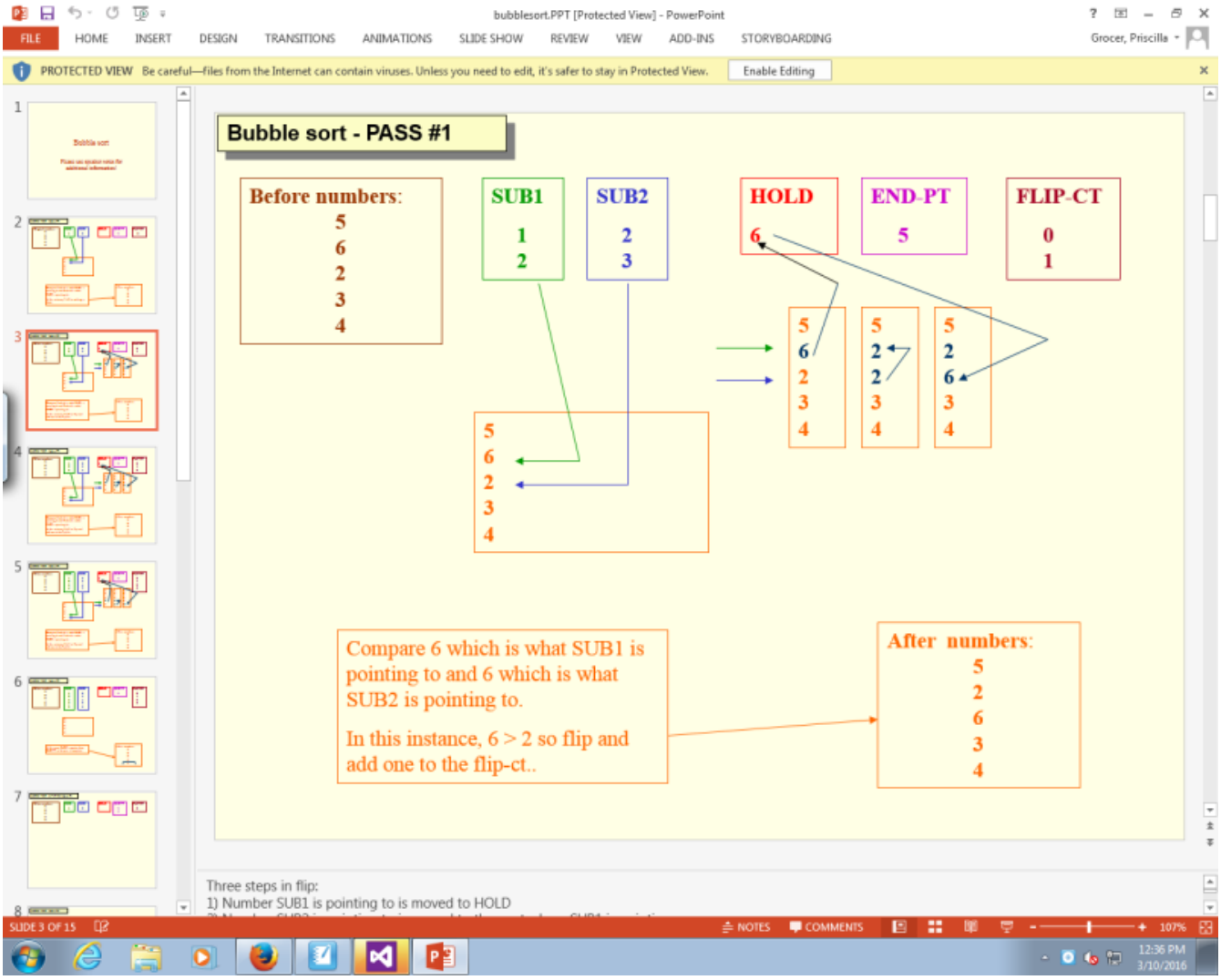
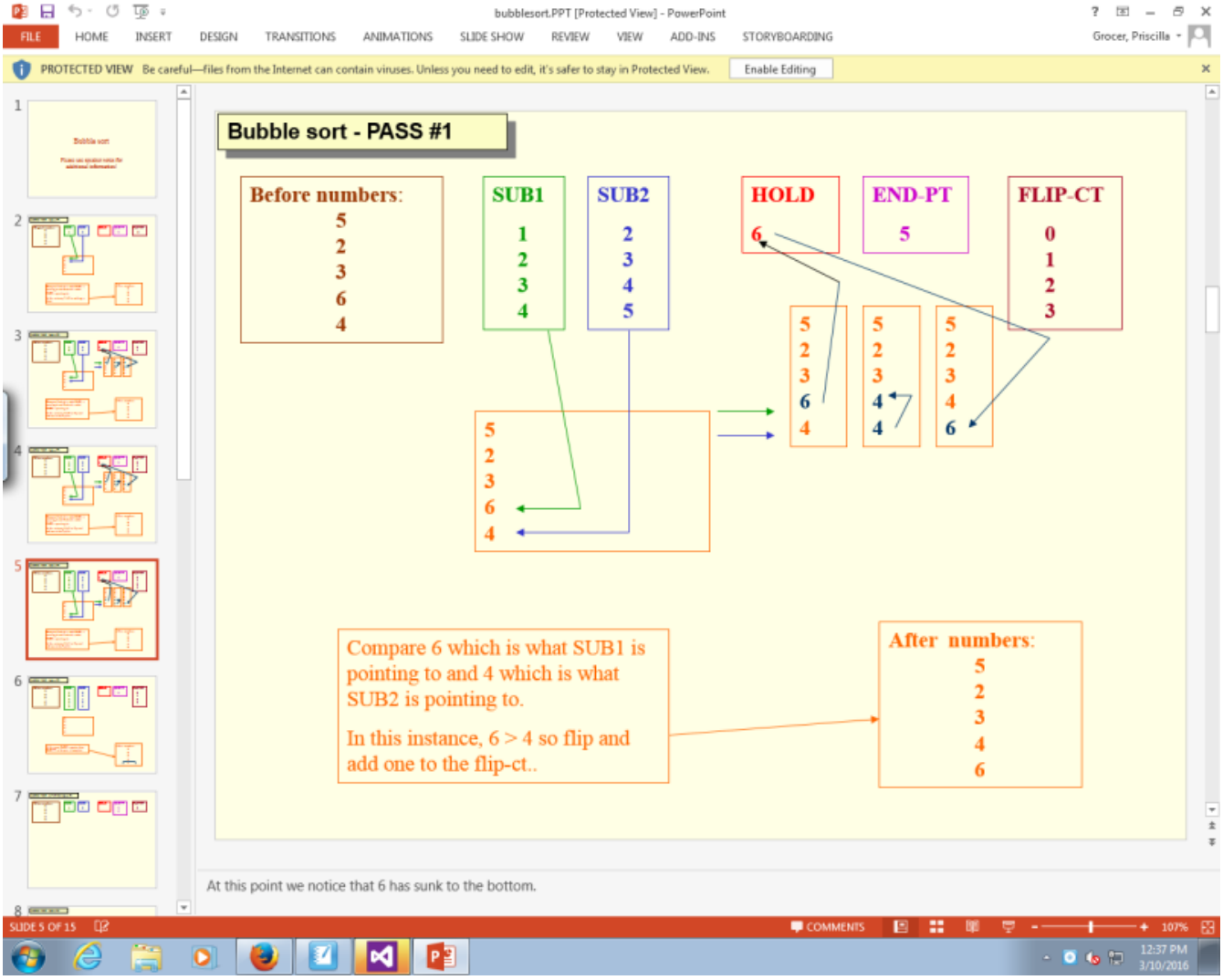

bubblesort.PPT [Protected View] - PowerPoint
FILE HOME INSERT DESIGN TRANSITIONS ANIMATIONS SLIDE SHOW REVIEW VIEW ADD-INS STORYBOARDING
PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View.
Enable Editing
Bubble sort - PASS #1
Original numbers:
5
6
2
3
4
SUB1
1
SUB2
2
HOLD
END-PT
5
FLIP-CT
0
I did not take pictures of all the screens - please look at the Powerpoint.
5
6
2
3
4
Compare 5 which is what SUB1 is pointing to and 6 which is what SUB2 is pointing to.
In this instance, 5 < 6 so nothing is done.
After numbers:
5
6
2
3
4
SUB1 points to the first thing we want to compare and SUB2 points to the second. That means SUB1 starts out at 1 and SUB2 starts out at 2.
The HOLD is used when doing a flip.
SLIDE 2 OF 15
COMMENTS
107%
12:34 PM
3/10/2016


bubblesort.PPT [Protected View] - PowerPoint
FILE HOME INSERT DESIGN TRANSITIONS ANIMATIONS SLIDE SHOW REVIEW VIEW ADD-INS STORYBOARDING
PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View.
Enable Editing
Bubble sort - PASS #1
Before numbers:
5
6
2
3
4
SUB1
1
2
SUB2
2
3
HOLD
6
END-PT
5
FLIP-CT
0
1
5
6
2
3
4
5
2
2
3
4
5
2
6
3
4
5
6
2
3
4
Compare 6 which is what SUB1 is pointing to and 6 which is what SUB2 is pointing to.
In this instance, 6 > 2 so flip and add one to the flip-ct..
After numbers:
5
2
6
3
4
Three steps in flip:
1) Number SUB1 is pointing to is moved to HOLD
SLIDE 3 OF 15
NOTES
COMMENTS
107%


bubblesort.PPT [Protected View] - PowerPoint
FILE HOME INSERT DESIGN TRANSITIONS ANIMATIONS SLIDE SHOW REVIEW VIEW ADD-INS STORYBOARDING
PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View.
Enable Editing
Bubble sort - PASS #1
Before numbers:
5
2
3
6
4
SUB1
1
2
3
4
SUB2
2
3
4
5
HOLD
6
END-PT
5
FLIP-CT
0
1
2
3
5
2
3
6
4
5
2
3
4
4
5
2
3
4
6
Compare 6 which is what SUB1 is pointing to and 4 which is what SUB2 is pointing to.
In this instance, 6 > 4 so flip and add one to the flip-ct..
After numbers:
5
2
3
4
6
At this point we notice that 6 has sunk to the bottom.
SLIDE 5 OF 15
COMMENTS
107%
12:37 PM
3/10/2016

bubblesort.PPT [Protected View] - PowerPoint

FILE HOME INSERT DESIGN TRANSITIONS ANIMATIONS SLIDE SHOW REVIEW VIEW ADD-INS STORYBOARDING

PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View. Enable Editing

Bubble sort - PASS #3

Before numbers:
2
3
4
5
6

SUB1
1
2
3

SUB2
2
3
4

HOLD

END-PT
3

FLIP-CT
0

END-PT signals the end of the pass. We can test either by using SUB2 > END-PT or SUB1 = END-PT.

At this point SUB2 is greater than END-PT so the pass is complete.

AND since there were no flips the sort is complete.

After numbers:
2
3
4
5
6

The sort is complete because there has been a pass with no flips or because there have been the right number of passes (with 5 elements, the right number of passes would be 4). Note that this can be measured against END-PT.

SLIDE 15 OF 15 COMMENTS 107%

12:41 PM 3/10/2016

bubblesort (Running) - Microsoft Visual Studio

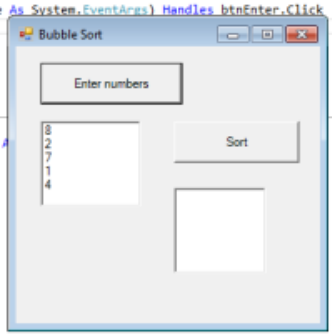
```
Public Class frmBubble
    Inherits System.Windows.Forms.Form
    Dim numArray(4) As Integer

Windows Form Designer generated code

    Private Sub btnEnter_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnEnter.Click
        Dim i As Integer
        For i = 0 To 4
            numArray(i) = InputBox("Enter Number", "Sort")
            lstEnter.Items.Add(numArray(i))
        Next
    End Sub

    Private Sub btnSort_Click(ByVal sender As System.Object, ByVal e A
        Dim i As Integer
        Dim endPt As Integer = 4
        Dim flipCt As Integer = 9
        Dim sub1 As Integer
        Dim sub2 As Integer
        Dim holdSlot As String
        Do Until endPt = 0 Or flipCt = 0
            sub1 = 0
            sub2 = 1
            flipCt = 0
            Do Until sub2 > endPt
                If numArray(sub1) > numArray(sub2) Then
                    holdSlot = numArray(sub1)
                    numArray(sub1) = numArray(sub2)
                    numArray(sub2) = holdSlot
                    flipCt = flipCt + 1
                End If
                sub1 = sub1 + 1
                sub2 = sub2 + 1
            Loop
            endPt = endPt - 1
        Loop

        For i = 0 To 4
            lstSort.Items.Add(numArray(i))
        Next
    End Sub
End Class
```



The code starts sub1 and sub2 at 0 while the logic code in the powerpoint starts at 1.

bubblesort (Running) - Microsoft Visual Studio

FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP

frmBubble.vb [Design]

btnEnter | Click

```
Public Class frmBubble
    Inherits System.Windows.Forms.Form
    Dim numArray(4) As Integer

    Windows Form Designer generated code

    Private Sub btnEnter_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnEnter.Click
        Dim i As Integer
        For i = 0 To 4
            numArray(i) = InputBox("Enter Number", "Sort")
            lstEnter.Items.Add(numArray(i))
        Next
    End Sub

    Private Sub btnSort_Click(ByVal sender As System.Object, ByVal e A
        Dim i As Integer
        Dim endPt As Integer = 4
        Dim flipCt As Integer = 9
        Dim sub1 As Integer
        Dim sub2 As Integer
        Dim holdSlot As String
        Do Until endPt = 0 Or flipCt = 0
            sub1 = 0
            sub2 = 1
            flipCt = 0
            Do Until sub2 > endPt
                If numArray(sub1) > numArray(sub2) Then
                    holdSlot = numArray(sub1)
                    numArray(sub1) = numArray(sub2)
                    numArray(sub2) = holdSlot
                    flipCt = flipCt + 1
                End If
                sub1 = sub1 + 1
                sub2 = sub2 + 1
            Loop
            endPt = endPt - 1
        Loop

        For i = 0 To 4
            lstSort.Items.Add(numArray(i))
        Next
    End Sub
End Class
```

Bubble Sort

Enter numbers

8
2
7
1
4

Sort

1
2
4
7
8

IntelliTrace

Streaming Video: Collecting and analyzing data in product

To view IntelliTrace data, you must break execution of your application.

Break all

More options:

Open IntelliTrace settings

Learn more about IntelliTrace

IntelliTrace | Solution Explorer

Properties

Ready Ln 91 Col 9 Ch 9 INS

topdownsort.PPT [Protected View] - PowerPoint

FILE HOME INSERT DESIGN TRANSITIONS ANIMATIONS SLIDE SHOW REVIEW VIEW ADD-INS STORYBOARDING

PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View. Enable Editing

Number and name

Original numbers:	Original names:
15	Rake
36	Hoe
24	Shovel
12	Trowel
20	Sprinkler

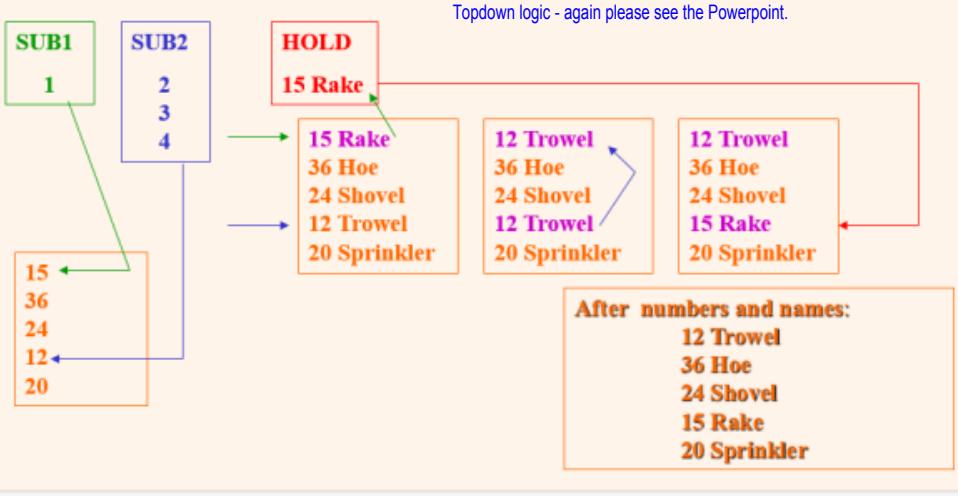
Three steps in flip:

1) Number and name SUB1 is pointing to is moved to hold

2) Number and name SUB2 is pointing to is moved to the spot where SUB1 is pointing

3) Number and from HOLD is moved to spot where SUB2 is pointing

Topdown logic - again please see the Powerpoint.



After numbers and names:
12 Trowel
36 Hoe
24 Shovel
15 Rake
20 Sprinkler

Note that when another element is involved such as name, it must be flipped as well. If number and name appear in the same array they can be held in the same hold area. If two arrays are involved (one for number and one for name), then you will need to have a hold area for the number and a hold area for the

SLIDE 16 OF 16 COMMENTS 107%

bubblesort - Microsoft Visual Studio

```
Public Class frmBubble
    Inherits System.Windows.Forms.Form
    Dim numArray(4) As Integer

    Windows Form Designer generated code

    Private Sub btnEnter_Click(ByVal sender As System.Object, ByVal e As System.Even
        Dim i As Integer
        For i = 0 To 4
            numArray(i) = InputBox("Enter Number", "Sort")
            lstEnter.Items.Add(numArray(i))
        Next
    End Sub

    Private Sub btnSort_Click(ByVal sender As System.Object, ByVal e As System.Event
        Dim i As Integer
        Dim endPt As Integer = 4
        Dim flipCt As Integer = 9
        Dim sub1 As Integer
        Dim sub2 As Integer
        Dim holdSlot As String
        Do Until endPt = 0 Or flipCt = 0
            sub1 = 0
            sub2 = 1
            flipCt = 0
            Do Until sub2 > endPt
                If numArray(sub1) > numArray(sub2) Then
                    holdSlot = numArray(sub1)
                    numArray(sub1) = numArray(sub2)
                    numArray(sub2) = holdSlot
                    flipCt = flipCt + 1
                End If
                sub1 = sub1 + 1
                sub2 = sub2 + 1
            Loop
            endPt = endPt - 1
        Loop

        For i = 0 To 4
            lstSort.Items.Add(numArray(i))
        Next
    End Sub
End Class
```

Back to the bubble. The topdown is the one you need to write for homework.

File Edit View History Bookmarks Tools Help

MOD 11 Check Digit | Convert Upper Case to Lo... | http://www.pgrocer.net/Cis17/... | New Tab

www.pgrocer.net/Cis156/mod11.html | html uppercase command

Most Visited

MOD 11 Check Digit

A check digit is a number that is used to validate a series of numbers whose accuracy you want to insure. Frequently the last digit of a number string such as identification number is a check digit. Lets say the identification number starts out at 6 digits. A calculation is done using the six digits and a seventh digit is produced as a result of the calculation. This number is the check digit. There are many calculations that can be used - this example illustrates the logic of the MOD11 check digit.

Steps to calculate the MOD11 check digit for a number such as an id #:

- Assign weights to each digit of the id #. The weights in MOD11 are from 2 through a maximum of 10 beginning with the low order position in the field.
- Each digit in the id # is multiplied by its weight
- The results of the multiplication are added together
- This product is divided by the modulus number 11
- The remainder is subtracted from the modulus number 11 giving the check digit

Example: find the check digit for the number **036532**

0	3	6	5	3	2
x7	x6	x5	x4	x3	x2
0	18	30	20	9	4

7

+ 1

```
0 + 18 + 30 + 20 + 9 + 4 = 81

81/11 = 7 remainder 4

11 - 4 = 7
```

7 is therefore the check digit.

PROBLEMS: If the remainder from the division is 0 or 1, then the subtraction will yield a two digit number of either 10 or 11. This won't work, so if the check digit is 10, then X is frequently used as the check digit and if the check digit is 11 then 0 is used as the check digit. If X is used, then the field for the check digit has to be defined as character text/string or there will be a numeric problem.

Steps to verify if the check digit is included as part of the number:

- The entire number is multiplied by the same weights that were used to calculate and the check digit itself is multiplied by 1.
- The results of the multiplication are added together.
- The sum is divided by 11 and if the remainder is 0, the number is correct.

PROBLEM: Note that if the check digit is X then 10 is used in the multiplication. Code for this occurrence must be included.

Example of verifying the number **0365327** where 7 is the calculated MOD11 check digit:

0	3	6	5	3	2	7

File Edit View History Bookmarks Tools Help
MOD 11 Check Digit | Convert Upper Case to Lo... | http://www.pgrocer.net/Cis17/... | New Tab
www.pgrocer.net/Cis156/mod11.html
html uppercase command
Most Visited

- This product is divided by the modulus number 11
- The remainder is subtracted from the modulus number 11 giving the check digit

Example: find the check digit for the number **036532**

0	3	6	5	3	2
x7	x6	x5	x4	x3	x2
0	18	30	20	9	4

```
0 + 18 + 30 + 20 + 9 + 4 = 81

81/11 = 7 remainder 4

11 - 4 = 7
```

7 is therefore the check digit.

PROBLEMS: If the remainder from the division is 0 or 1, then the subtraction will yield a two digit number of either 10 or 11. This won't work, so if the check digit is 10, then X is frequently used as the check digit and if the check digit is 11 then 0 is used as the check digit. If X is used, then the field for the check digit has to be defined as character text/string or there will be a numeric problem.

Steps to verify if the check digit is included as part of the number:

- The entire number is multiplied by the same weights that were used to calculate and the check digit itself is multiplied by 1.
- The results of the multiplication are added together.
- The sum is divided by 11 and if the remainder is 0, the number is correct.

PROBLEM: Note that if the check digit is X then 10 is used in the multiplication. Code for this occurrence must be included.

Example of verifying the number **0365327** where 7 is the calculated MOD11 check digit:

0	3	6	5	3	2	7
x7	x6	x5	x4	x3	x2	x1
0	18	30	20	9	4	7

```
0 + 18 + 30 + 20 + 9 + 4 + 7 = 88

88/11 is 8 remainder 0
```

Since the remainder from this calculation is 0, the check digit 7 is valid.

1:24 PM
3/10/2016

File Edit View History Bookmarks Tools Help

MOD 11 Check Digit | Convert Upper Case to Lo... | http://www.pgrocer.net/Cis17/... | New Tab

www.pgrocer.net/Cis156/mod11.html

Most Visited

- This product is divided by the modulus number 11
- The remainder is subtracted from the modulus number 11 giving the check digit

Example: find the check digit for the number **036532**

0	3	6	5	3	2
x7	x6	x5	x4	x3	x2
0	18	30	20	9	4

```
0 + 18 + 30 + 20 + 9 + 4 = 81

81/11 = 7 remainder 4

11 - 4 = 7
```

array(0)
array(1)

7 is therefore the check digit.

PROBLEMS: If the remainder from the division is 0 or 1, then the subtraction will yield a two digit number of either 10 or 11. This won't work, so if the check digit is 10, then X is frequently used as the check digit and if the check digit is 11 then 0 is used as the check digit. If X is used, then the field for the check digit has to be defined as character text/string or there will be a numeric problem.

Steps to verify if the check digit is included as part of the number:

- The entire number is multiplied by the same weights that were used to calculate and the check digit itself is multiplied by 1.
- The results of the multiplication are added together.
- The sum is divided by 11 and if the remainder is 0, the number is correct.

PROBLEM: Note that if the check digit is X then 10 is used in the multiplication. Code for this occurrence must be included.

Example of verifying the number **0365327** where 7 is the calculated MOD11 check digit:

0	3	6	5	3	2	7
x7	x6	x5	x4	x3	x2	x1
0	18	30	20	9	4	7

```
0 + 18 + 30 + 20 + 9 + 4 + 7 = 88

88/11 is 8 remainder 0
```

Since the remainder from this calculation is 0, the check digit 7 is valid.

File Edit View History Bookmarks Tools Help

MOD 11 Check Digit | Convert Upper Case to Lo... | http://www.pgrocer.net/Cis17/... | New Tab

www.pgrocer.net/Cis156/mod11.html

Most Visited

- This product is divided by the modulus number 11
- The remainder is subtracted from the modulus number 11 giving the check digit

Example: find the check digit for the number **036532**

0	3	6	5	3	2
x7	x6	x5	x4	x3	x2
0	18	30	20	9	4

```
0 + 18 + 30 + 20 + 9 + 4 = 81

81/11 = 7 remainder 4

11 - 4 = 7
```

11 - 1 = 10. use X

11 - 0 = 11 - use 0

10 - 0 = 10 - use X

7 is therefore the check digit.

PROBLEMS: If the remainder from the division is 0 or 1, then the subtraction will yield a two digit number of either 10 or 11. This won't work, so if the check digit is 10, then X is frequently used as the check digit and if the check digit is 11 then 0 is used as the check digit. If X is used, then the field for the check digit has to be defined as character text/string or there will be a numeric problem.

Steps to verify if the check digit is included as part of the number:

- The entire number is multiplied by the same weights that were used to calculate and the check digit itself is multiplied by 1.
- The results of the multiplication are added together.
- The sum is divided by 11 and if the remainder is 0, the number is correct.

PROBLEM: Note that if the check digit is X then 10 is used in the multiplication. Code for this occurrence must be included.

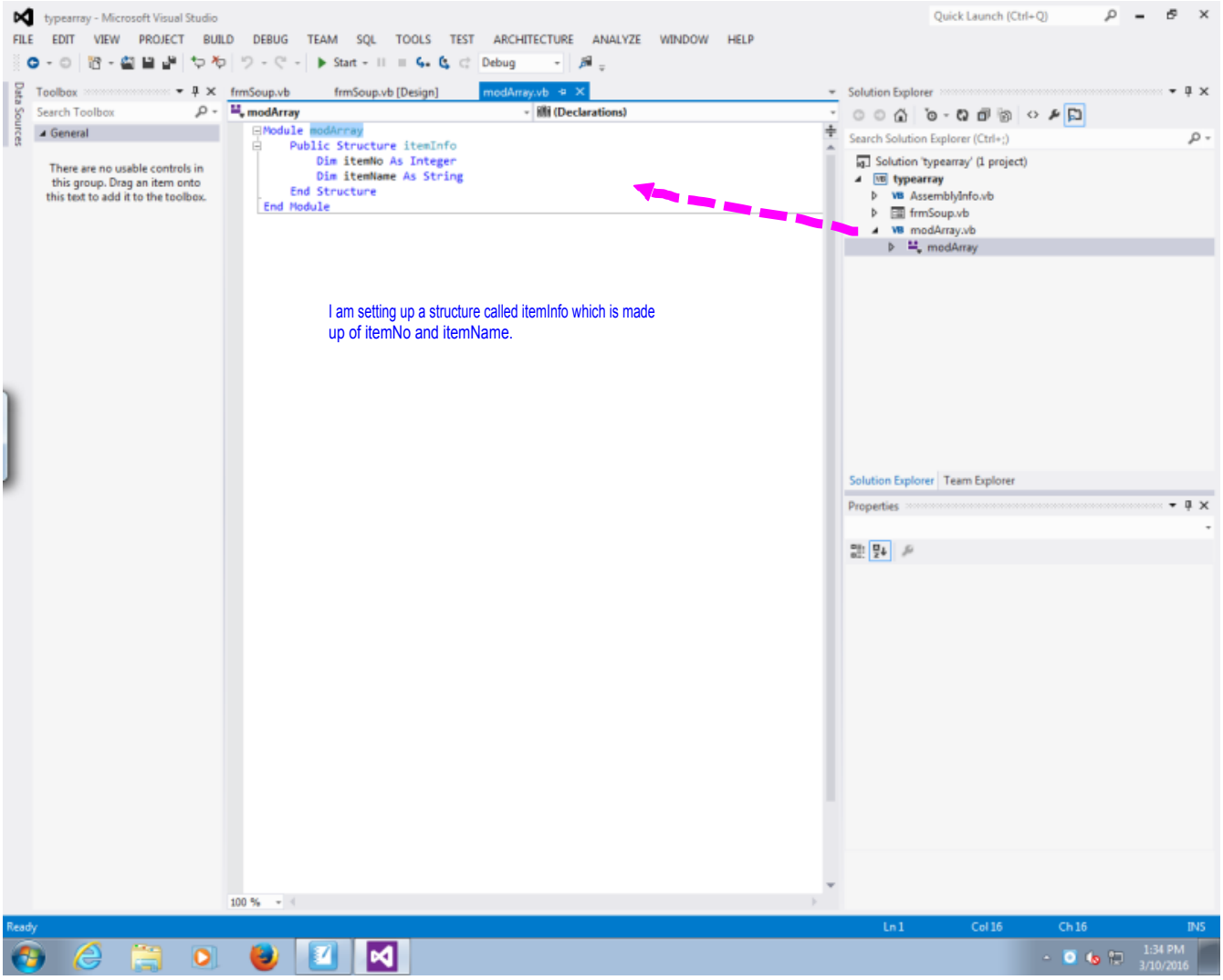
Example of verifying the number **0365327** where 7 is the calculated MOD11 check digit:

0	3	6	5	3	2	7
x7	x6	x5	x4	x3	x2	x1
0	18	30	20	9	4	7

```
0 + 18 + 30 + 20 + 9 + 4 + 7 = 88

88/11 is 8 remainder 0
```

Since the remainder from this calculation is 0, the check digit 7 is valid.


typearray - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Toolbox
Search Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
frmSoup.vb
frmSoup.vb [Design]
modArray.vb
modArray
(Declarations)
Module modArray
Public Structure itemInfo
Dim itemNo As Integer
Dim itemName As String
End Structure
End Module
I am setting up a structure called itemInfo which is made up of itemNo and itemName.
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'typearray' (1 project)
typearray
AssemblyInfo.vb
frmSoup.vb
modArray.vb
modArray
Solution Explorer Team Explorer
Properties
Ready
Ln 1 Col 16 Ch 16 INS

typearray - Microsoft Visual Studio

frmSoup.vb

```
Public Class frmSoup
    Inherits System.Windows.Forms.Form
    Dim itemArray(8) As itemInfo

Windows Form Designer generated code

    Private Sub frmSoupCreate_Load(ByVal sender As System.Object, ByVal e As System.
        itemArray(0).itemNo = 3
        itemArray(1).itemNo = 12
        itemArray(2).itemNo = 15
        itemArray(3).itemNo = 17
        itemArray(4).itemNo = 24
        itemArray(5).itemNo = 25
        itemArray(6).itemNo = 27
        itemArray(7).itemNo = 28
        itemArray(8).itemNo = 45
        itemArray(0).itemName = "SEAFOOD CHOWDER"
        itemArray(1).itemName = "CORN CHOWDER"
        itemArray(2).itemName = "CLAM CHOWDER"
        itemArray(3).itemName = "TOMATO SOUP"
        itemArray(4).itemName = "CHICKEN SOUP"
        itemArray(5).itemName = "VEGETABLE SOUP"
        itemArray(6).itemName = "ONION SOUP"
        itemArray(7).itemName = "GREEN PEA SOUP"
        itemArray(8).itemName = "WONTON SOUP"
        FileOpen(1, "soup", OpenMode.Input)
    End Sub

    Private Sub btnClear_Click(ByVal sender As System.Object, ByVal
        txtOrderNo.Clear()
        txtItemNo.Clear()
        txtNumOrd.Clear()
    End Sub

    Private Sub btnExit_Click(ByVal sender As System.Object, ByVal
        FileClose(1)
        Me.Close()
    End Sub

    Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.Event
        Dim wkOrderNo As Integer, wkItemNo As Integer
        Dim wkNumOrd As Integer
        If Not EOF(1) Then
            Input(1, wkOrderNo)
            Input(1, wkItemNo)
            Input(1, wkNumOrd)
            txtOrderNo.Text = wkOrderNo
            txtItemNo.Text = wkItemNo
            txtNumOrd.Text = wkNumOrd
            txtItemName.Text = SearchArray(wkItemNo)
```

Untitled - Notepad

```
Module modArray
    Public Structure itemInfo
        Dim itemNo As Integer
        Dim itemName As String
    End Structure
End Module
```

typearray - Microsoft Visual Studio

FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP

frmSoup.vb | frmSoup.vb [Design] | modArray.vb

btnRead | Click

```
Public Class frmSoup
    Inherits System.Windows.Forms.Form
    Dim itemArray(8) As itemInfo

Windows Form Designer generated code

    Private Sub frmSoupCreate_Load(ByVal sender As System.Object, ByVal e As System.
        itemArray(0).itemNo = 3
        itemArray(1).itemNo = 12
        itemArray(2).itemNo = 15
        itemArray(3).itemNo = 17
        itemArray(4).itemNo = 24
        itemArray(5).itemNo = 25
        itemArray(6).itemNo = 27
        itemArray(7).itemNo = 28
        itemArray(8).itemNo = 45
        itemArray(0).itemName = "SEAFOOD CHOWDER"
        itemArray(1).itemName = "CORN CHOWDER"
        itemArray(2).itemName = "CLAM CHOWDER"
        itemArray(3).itemName = "TOMATO SOUP"
        itemArray(4).itemName = "CHICKEN SOUP"
        itemArray(5).itemName = "VEGETABLE SOUP"
        itemArray(6).itemName = "ONION SOUP"
        itemArray(7).itemName = "GREEN PEA SOUP"
        itemArray(8).itemName = "WONTON SOUP"
        FileOpen(1, "soup", OpenMode.Input)
    End Sub

    Private Sub btnClear_Click(ByVal sender As System.Object, ByVal
        txtOrderNo.Clear()
        txtItemNo.Clear()
        txtNumOrd.Clear()
    End Sub

    Private Sub btnExit_Click(ByVal sender As System.Object, ByVal
        FileClose(1)
        Me.Close()
    End Sub

    Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.Event
        Dim wkOrderNo As Integer, wkItemNo As Integer
        Dim wkNumOrd As Integer
        If Not EOF(1) Then
            Input(1, wkOrderNo)
            Input(1, wkItemNo)
            Input(1, wkNumOrd)
            txtOrderNo.Text = wkOrderNo
            txtItemNo.Text = wkItemNo
            txtNumOrd.Text = wkNumOrd
            txtItemName.Text = SearchArray(wkItemNo)
```

Solution Explorer

Solution 'typearray' (1 project)
- typearray
  - AssemblyInfo.vb
  - frmSoup.vb
  - modArray.vb
    - modArray

Untitled - Notepad

```
Module modArray
    Public Structure itemInfo
        Dim itemNo As Integer
        Dim itemName As String
    End Structure
End Module
```

Ready Ln 204 Col 9 Ch 9 INS

typearray - Microsoft Visual Studio

FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP

Add Windows Form...
Add User Control...
Add Component...
Add Module...
Add Class...
Add New Data Source...
Add New Item... Ctrl+Shift+A
Add Existing Item... Shift+Alt+A
Exclude From Project
Show All Files
Add Reference...
Add Service Reference...
Set as StartUp Project
Manage NuGet Packages...
Enable NuGet Package Restore
Refresh Project Toolbox Items
typearray Properties...

frmSoup.vb [Design] modArray.vb

```
Class frmSoup
erits System.Windows.Forms.Form
 itemArray(8) As itemInfo

s Form Designer generated code

vate Sub frmSoupCreate_Load(ByVal sender As System.Object, ByVal e As System.
 itemArray(0).itemNo = 3
 itemArray(1).itemNo = 12
 itemArray(2).itemNo = 15
 itemArray(3).itemNo = 17
 itemArray(4).itemNo = 24
 itemArray(5).itemNo = 25
 itemArray(6).itemNo = 27
 itemArray(7).itemNo = 28
 itemArray(8).itemNo = 45
 itemArray(0).itemName = "SEAFOOD CHOWDER"
 itemArray(1).itemName = "CORN CHOWDER"
 itemArray(2).itemName = "CLAM CHOWDER"
 itemArray(3).itemName = "TOMATO SOUP"
 itemArray(4).itemName = "CHICKEN SOUP"
 itemArray(5).itemName = "VEGETABLE SOUP"
 itemArray(6).itemName = "ONION SOUP"
 itemArray(7).itemName = "GREEN PEA SOUP"
 itemArray(8).itemName = "WONTON SOUP"
 FileOpen(1, "soup", OpenMode.Input)
End Sub

Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e As System.Even
    txtOrderNo.Clear()
    txtItemNo.Clear()
    txtNumOrd.Clear()
End Sub

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As System.Event
    FileClose(1)
    Me.Close()
End Sub

Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.Event
    Dim wkOrderNo As Integer, wkItemNo As Integer
    Dim wkNumOrd As Integer
    If Not EOF(1) Then
        Input(1, wkOrderNo)
        Input(1, wkItemNo)
        Input(1, wkNumOrd)
        txtOrderNo.Text = wkOrderNo
        txtItemNo.Text = wkItemNo
        txtNumOrd.Text = wkNumOrd
        txtItemName.Text = SearchArray(wkItemNo)
```

Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'typearray' (1 project)
typearray
AssemblyInfo.vb
frmSoup.vb
modArray.vb
modArray

Solution Explorer | Team Explorer

Properties

Ready Ln 204 Col 9 Ch 9 INS

typearray - Microsoft Visual Studio

frmSoup.vb [Design] | modArray.vb

```
        FileClose(1)
        Me.Close()
    End Sub

    Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.Event
        Dim wkOrderNo As Integer, wkItemNo As Integer
        Dim wkNumOrd As Integer
        If Not EOF(1) Then
            Input(1, wkOrderNo)
            Input(1, wkItemNo)
            Input(1, wkNumOrd)
            txtOrderNo.Text = wkOrderNo
            txtItemNo.Text = wkItemNo
            txtNumOrd.Text = wkNumOrd
            txtItemName.Text = SearchArray(wkItemNo)
        Else
            MessageBox.Show("EOF reached")
            btnRead.Visible = False
        End If
    End Sub
    Function SearchArray(ByVal wkItemNo)
        Dim itemSub As Integer = 0
        Dim matchInd As String = "NO"
        Do Until itemSub > 8 Or matchInd = "YES"
            If wkItemNo = itemArray(itemSub).itemNo Then
                matchInd = "YES"
            Else
                itemSub = itemSub + 1
            End If
        Loop
        If matchInd = "YES" Then
            Return itemArray(itemSub).itemName
        Else
            Return "Match Not Found"
        End If
    End Function
End Class
```

Untitled - Notepad

```
Module modArray
    Public Structure itemInfo
        Dim itemNo As Integer
        Dim itemName As String
    End Structure
End Module
```