ARRAYS

Do Now:

Write a program that reads in the grades from your last test. The program should find the average of the grades and display the class average. The program should then print the names of those students above the class average. The program should ONLY read the file ONCE!!

The text file is on my website under Arrays

Example Code

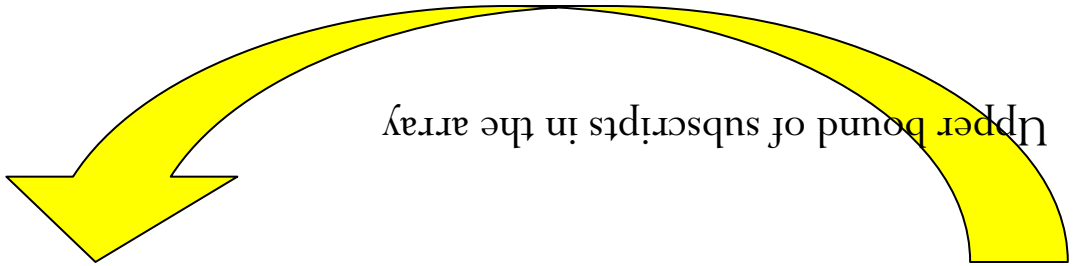
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click

  Dim student1 As String, score1 As Double
  Dim student2 As String, score2 As Double
  Dim student3 As String, score3 As Double
      . . .
  Dim student10 As String, score10 As Double
  Dim sr as IO.Streamreader=IO.File.OpenText("grades.TXT")
      student1 = sr.ReadLine
      score1 = CDbl(sr.ReadLine)
      .     .     .

  'Compute the average grade/display the namesof
  those above the average.
```

7.1 Creating and Accessing Arrays

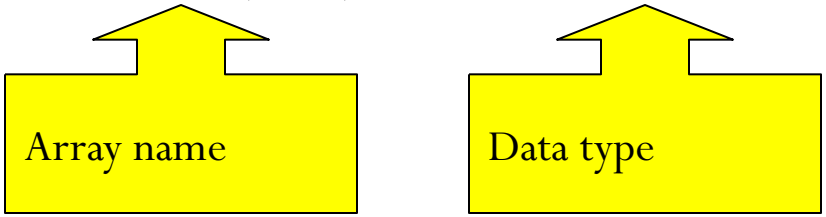
- Declaring an Array Variable
- The Load Event Procedure
- The GetUpperBound Method
- ReDim Statement
- Using an Array as a Frequency Table

- A **variable** (or simple variable) is a name to which Visual Basic can assign a single value.
- An **array variable** is a collection of simple variables of the same type to which Visual Basic can efficiently assign a list of values.

Dim *arrayName*(n) as *varType*

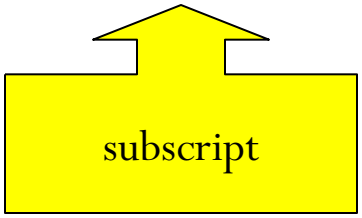



```
Dim student(29) As String
Dim score(29) As Double
```



Putting Values into an Array

student(0) = “Chris"



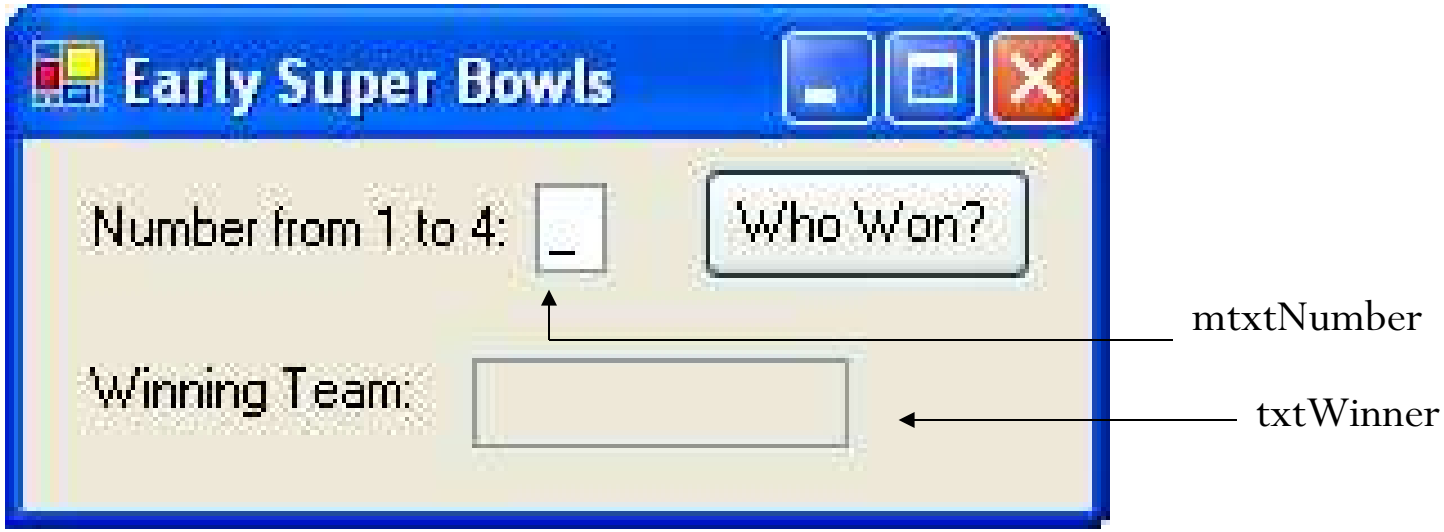
Read: "student sub zero equals Chris"

Which means that the string “Chris" is being stored at the first location in the array called student… **because all arrays begin counting at 0.**

Array Terminology

- Dim arrayName(n) As DataType
- 0 is the "lower bound" of the array
- n is the "upper bound" of the array – the last available subscript in this array
- The number of elements, n + 1, is the *size* of the array

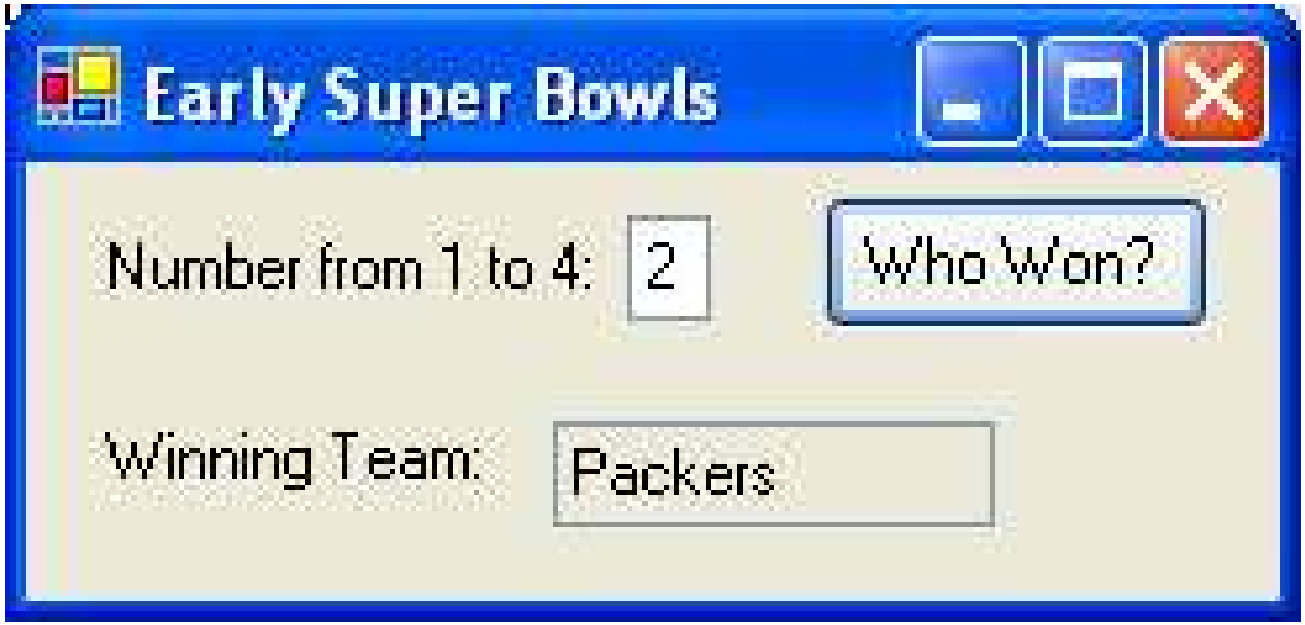
Example 1: Form



Example 1

```
Private Sub btnWhoWon_Click(...) _
                                Handles btnWhoWon.Click
  Dim teamName(3) As String
  Dim n As Integer
  'Place Super Bowl Winners into the array
  teamName(0) = "Packers"
  teamName(1) = "Packers"
  teamName(2) = "Jets"
  teamName(3) = "Chiefs"
  'Access array
  n = CInt(txtNumber.Text)
  txtWinner.Text = teamName(n - 1)
End Sub
```

Example 1: Output



Initializing Arrays

- Arrays may be initialized when they are created:

```
Dim arrayName() As varType = {value0,_
  value1, value2, ..., valueN}
```

- declares an array having upper bound *N* and assigns *value0* to *arrayName*(0), *value1* to *arrayName*(1), …, and *valueN* to *arrayName*(*N*).

GetUpperBound Method

The value of ***arrayName*.GetUpperBound(0)**

is the upper bound of *arrayName*().

```
Dim teamName() As String = {"Packers", "Packers",_
                            "Jets", "Chiefs"}
txtBox.Text = CStr(teamName.GetUpperBound(0))
```

Output: 3

ReDim Statement

The size of an array may be changed after it has been created.

```
ReDim arrayName(m)
```

where *arrayName* is the name of the already declared array and *m* is an Integer literal, variable, or expression, changes the upper bound of the array to *m*.

Preserve Keyword

`ReDim arrayName(m)`

resets all values to their default. This can be prevented with the keyword **Preserve**.

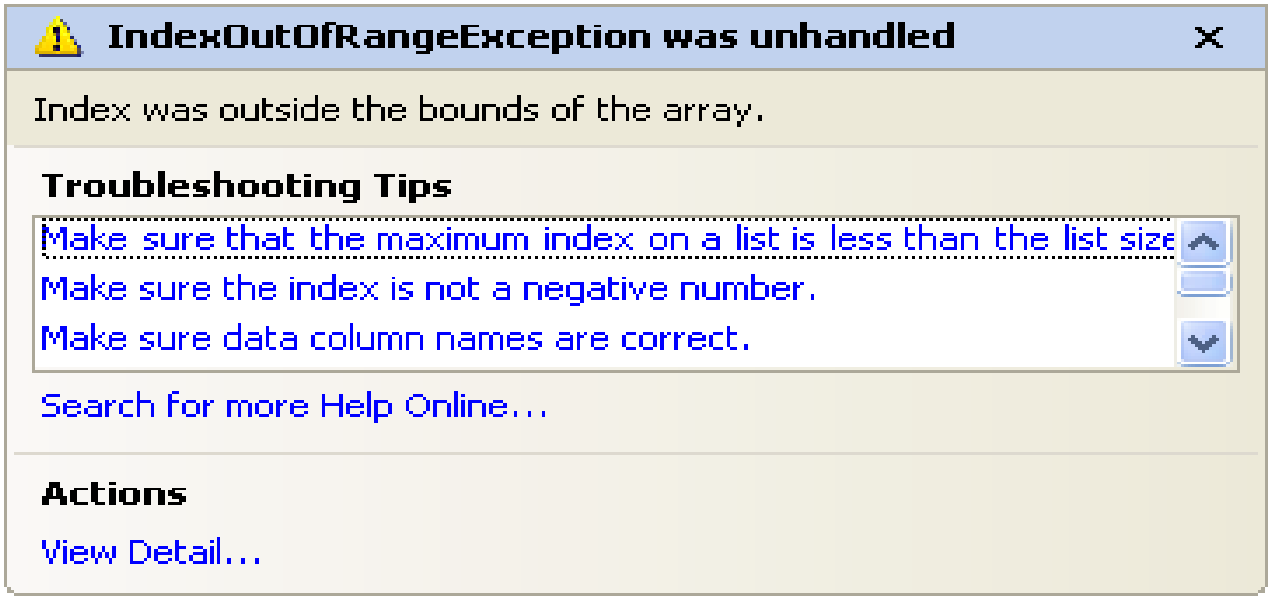
`ReDim Preserve arrayName(m)`

resizes the array and retains as many values as possible.

Out of Bounds Error

The following code references an array element that doesn't exist. This will cause an error.

```
Dim trees() As String = {"Sequoia", _
                         "Redwood", "Spruce"}
txtBox.Text = trees(5)
```



Passing Arrays to Procedures

- An array declared in a procedure is local to that procedure
- An entire array can be passed to a Sub or Function procedure
- The call statement uses the name of the array without parentheses.
- The header of the Sub of Function procedure uses the name with empty set of parentheses.

Example 4

- This example uses a Function procedure to add up the numbers in an array. The GetUpperBound method is used to determine how many numbers are in the array.

Example 4

```
Private Sub btnCompute_Click(...) Handles btnCompute.Click
  Dim score() As Integer = {85, 92, 75, 68, 84, 86, _
                            94, 74, 79, 88}
  txtAverage.Text = CStr(Sum(score) / 10)
End Sub

Function Sum(ByVal s() As Integer) As Integer
  Dim total As Integer = 0
  For index As Integer = 0 To s.GetUpperBound(0)
    total += s(index)
  Next
  Return total
End Function
```