Standard logic for developing a coded search routine.
Dim wkNumOrd As Integer
If Not EOF(1) Then
    Input(1, wkOrderNo)
    Input(1, wkItemNo)
    Input(1, wkNumOrd)
txtOrderNo.Text = wkOrderNo
txtItemName.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 As String)
    Dim itemSub As Integer = 0
    Dim matchInd As String = "NO"
    Do Until itemSub > 8 Or matchInd = "YES"
        If wkItemNo = itemNoArray(itemSub) Then
            matchInd = "YES"
        Else
            itemSub = itemSub + 1
        End If
    Loop
    If matchInd = "YES" Then
        Return itemNameArray(itemSub)
    Else
        Return "Match Not Found"
    End If
End Function

Function SearchWhile(ByVal wkItemNo)
    Dim itemSub As Integer = 0

2 parallel arrays

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    ' Code to initialize and populate arrays
End Sub

Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnClear.Click
    txtOrderNo.Clear()
    txtItemNo.Clear()
    txtNumOrd.Clear()
End Sub

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnExit.Click
    FileClose(1)
End Sub
```vbnet
Private Sub frmSoupCreate_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles frmSoupForm_Load
    itemNoArray(0) = 3
    itemNoArray(1) = 12
    itemNoArray(2) = 15
    itemNoArray(3) = 17
    itemNoArray(4) = 24
    itemNoArray(5) = 25
    itemNoArray(6) = 27
    itemNoArray(7) = 28
    itemNoArray(8) = 45
    itemNameArray(0) = "SEAFOOD CHOWDER"
    itemNameArray(1) = "CORN CHOWDER"
    itemNameArray(2) = "CLAM CHOWDER"
    itemNameArray(3) = "TOMATO SOUP"
    itemNameArray(4) = "CHICKEN SOUP"
    itemNameArray(5) = "VEGETABLE SOUP"
    itemNameArray(6) = "ONION SOUP"
    itemNameArray(7) = "GREEN PEA SOUP"
    itemNameArray(8) = "WONTON SOUP"
    FileOpen(1, "soup", OpenMode.Input)
End Sub

Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnClear.Click
txtOrderNo.Clear()
txtItemNo.Clear()
txtNumOrd.Clear()
End Sub

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnExit.Click
FileClose(1)
End Sub
```
Looking at why it was important to have the add to itemSub in the else rather than a stand alone. It will be added to even if a match is found so the itemSub will be 1 bigger than when the match was found.
2 ways to end the Do until - need a check to determine which one was the reason.
Logic telling you when to get out of the loop.

Function SearchArray(ByVal wkItemNo)
    Dim itemSub As Integer = 0
    Dim matchInd As String = "NO"
    Do Until itemSub > 8 Or matchInd = "YES"
        If wkItemNo = itemNoArray(itemSub) Then
            matchInd = "YES"
        Else
            itemSub = itemSub + 1
        End If
    Loop
    If matchInd = "YES" Then
        Return itemNameArray(ItemSub)
    Else
        Return "Match Not Found"
    End If
End Function

Function SearchWhile(ByVal wkItemNo)
    Dim itemSub As Integer = 0
    Dim matchInd As String = "NO"
    Do While itemSub < 8 And matchInd = "NO"
        If wkItemNo = itemNoArray(itemSub) Then
            matchInd = "YES"
        Else
            itemSub = itemSub + 1
        End If
    Loop
    If matchInd = "YES" Then
        Return itemNameArray(ItemSub)
    Else
        Return "Match Not Found"
    End If
End Function
First 2 characters will be used to search for a match and the characters after that are the name.
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 = Microsoft.VisualBasic.Left(itemArray(itemSub), 2) Then
    If wkItemNo = itemArray(itemSub).Substring(0, 2) Then
        matchInd = "YES"
    Else
        itemSub = itemSub + 1
    End If
Loop
If matchInd = "YES" Then
    Return Microsoft.VisualBasic.Mid(itemArray(itemSub), 3)
    Return itemArray(itemSub).Substring(2)
Else
    Return "Match Not Found"
End If
End Function
End Class
Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnExit.Click
    Me.Close()
End Sub

Private Sub btnRead_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnRead.Click
    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) As Integer
    Dim itemSub As Integer = 0
    Dim matchInd As String = "NO"
    Do Until itemSub > 0 Or matchInd = "YES"
        'If wkItemNo = Microsoft.VisualBasic.Left(itemArray(itemSub), 2) Then
        If wkItemNo = itemArray(itemSub).Substring(0, 2) Then
            matchInd = "YES"
        Else
            itemSub = itemSub + 1
        End If
    Loop
    Return itemSub
End Function
Bubble sort logic - see Powerpoint presentations

Bubble sort - PASS #2

Before numbers:
5
2
3
4
6

SUB1
1

SUB2
2

HOLD
5

END-PT
4

FLIP-CT
0
1

Compare 5 which is what SUB1 is pointing to and 2 which is what SUB2 is pointing to.
In this instance, 5 > 2 so a flip is done and 1 is added to flip-ct.

After numbers:
2
5
3
4
6

Note again that with the bubble sort, the heavy number sinks to the bottom.
Private Sub btnSort_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSort.Click
    Dim i As Integer
    Dim endPt As Integer = 4
    Dim flipCt As Integer = 0
    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
    Loop
    For i = 0 To 4
        lstSort.Items.Add(numArray(i))
    Next
End Sub
End Class
Your assignment is to code the topdown sort using my bubble as a starting point.