Higher

Computing Science

Software Design and Development

Standard Algorithms

Name:____________________________

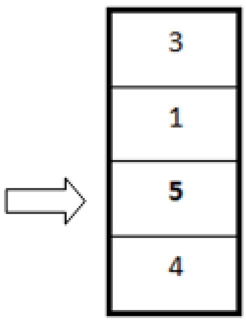
Contents

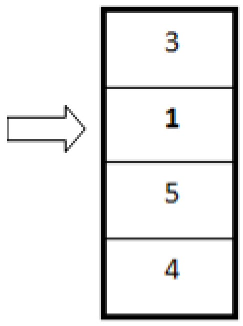
Standard Algorithms

There are four standard algorithms that you have to learn.

- Find Maximum
- Find Minimum
- Count Occurrences
- Linear Search

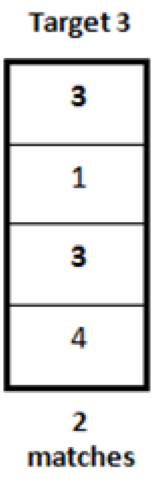
Find Maximum is used to identify the largest number in an array of values.

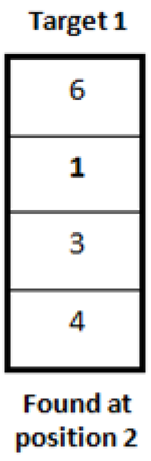




Find Minimum is used to identify the smallest number in an array of values.

Count Occurrences is used to identify the number of times a target value appears in an array of values.

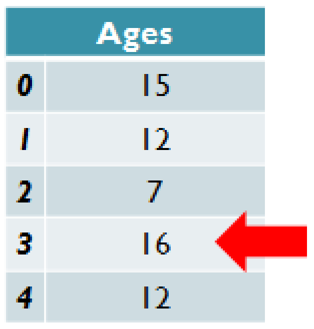




Linear Search is used to identify the position a target value in an array of values.

Find Maximum (Using Arrays)

Find Maximum is used to identify the largest value in an array.



Ages	
0	15
1	12
2	7
3	16
4	12

If used on the Ages array above, Find Maximum would return the value, 16

```
1.  SET max TO ages[0]

2.  FOR counter FROM 1 TO 4 DO
3.          IF ages[counter] > max
4.                  SET max TO ages[counter]
5.          END IF
6.  END FOR

7.  SEND "The highest age is "& max TO DISPLAY
```

1.Max is initialised to match first item in array

2.Repeat for each item in array starting at item 2

3.Check if current array item is higher than Max

4.If true, set Max to match current array item

Find Maximum (Using Record Structure)

Find Maximum is used to identify the largest value in an array of record structure.

If used on the UserDetails record structure below, Find Maximum would return the value, 16.

RECORD *Userdetails* **IS**
    {**STRING** Firstname, **STRING** Surname, **INTEGER** Age, **STRING** House}

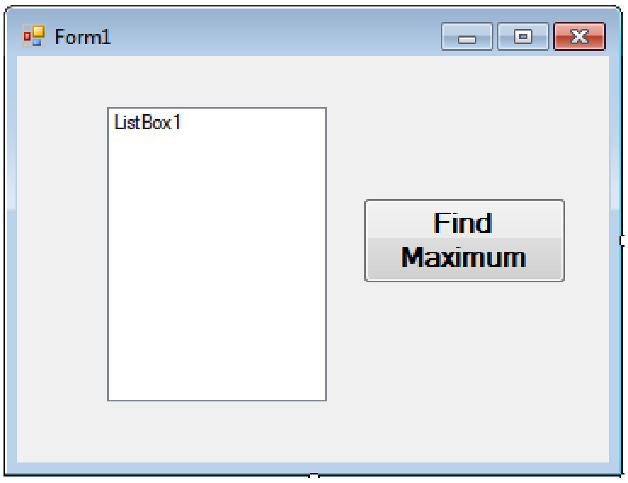
First Name	Surname	Age	House
Harry	Jones	14	Bute
Jenna	White	12	Kintyre
Laura	Cairns	15	Arran
Sam	Kay	16	Arran
Harry	Smith	14	Lomond

```
1.  SET max TO UserDetails [0].ages

2.  FOR counter FROM 1 TO 4 DO
3.          IF UserDetails [counter].ages > max
4.                  SET max TO UserDetails [counter].ages
5.          END IF
6.  END FOR

7.  SEND "The highest age is "& max TO DISPLAY
```

Worked Example 5a – Find Maximum

This example asks the user to enter 10 random numbers.

The Find Maximum algorithm is then used to identify the highest number entered.



```
Public Class Form1


Private Sub btnFindMax_Click(sender As Object, e As EventArgs) Handles
btnFindMax.Click


        Dim myList(10) As Integer
        Dim max As Integer


        For index = 1 To 10
            myList(index) = InputBox("Please enter number " & index)
            ListBox1.Items.Add(myList(index))
        Next index




        max = myList(1)

        For index = 2 To 10

            If myList(index) > max Then
                max = myList(index)
            End If
        Next index


        MsgBox("The largest number in the list is " & max)


End Sub




End Class
```

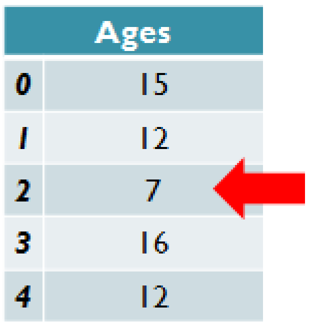
Fill array with values

Find Maximum Algorithm

Find Minimum (Using Arrays)

Find Minimum is used to identify the smallest value in an array.

If used on the Ages array (*right*), Find Minimum would return the value, 7



	Ages
0	15
1	12
2	7
3	16
4	12

```
1.  SET min TO ages[0]

2.  FOR counter FROM 1 TO 4 DO
3.          IF ages[counter] < min
4.                  SET min TO ages[counter]
5.          END IF
6.  END FOR

7.  SEND "The lowest age is "& min TO DISPLAY
```

1.Min is initialised to match first item in array

2.Repeat for each item in array starting at item 2

3.Check if current array item is lower than Min

4.If true, set Min to match current array item

Find Minimum (Using Record Structure)

Find Minimum is used to identify the smallest value in an array of record structure.

If used on the UserDetails record structure below, Find Minimum would return the value, 12.

RECORD *Userdetails* **IS**
{**STRING** Firstname, **STRING** Surname, **INTEGER** Age, **STRING** House}

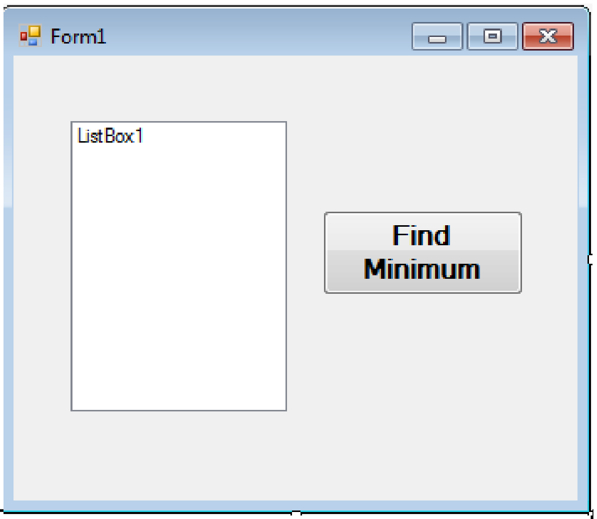
First Name	Surname	Age	House
Harry	Jones	14	Bute
Jenna	White	12	Kintyre
Laura	Cairns	15	Arran
Sam	Kay	16	Arran
Harry	Smith	14	Lomond

```
1.  SET min TO UserDetails[0].ages

2.  FOR counter FROM 1 TO 4 DO
3.          IF UserDetails [counter].ages < min
4.                  SET min TO UserDetails [counter].ages
5.          END IF
6.  END FOR

7.  SEND "The lowest age is "& min TO DISPLAY
```

Worked Example 5b – Find Min

This example asks the user to enter 10 random numbers.

The Find Minimum algorithm is then used to identify the lowest number entered.



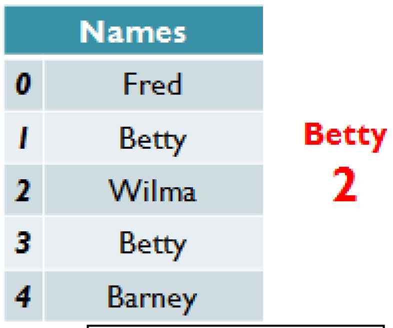
```
Public Class Form1

Private Sub btnFindin_Click(sender As Object, e As EventArgs) Handles btnFindMin.Click

        Dim myList(10) As Integer
        Dim min As Integer

        For index = 1 To 10
            myList(index) = InputBox("Please enter number " & index)
            ListBox1.Items.Add(myList(index))
        Next index



        min = myList(1)

        For index = 2 To 10

            If myList(index) < min Then
                min = myList(index)
            End If
        Next index


        MsgBox("The smallest number in the list is " & min)

End Sub


End Class
```

Fill array with values

Find Minimum Algorithm

Count Occurrences (Using Arrays)

Count Occurrences is used to identify how many times a particular value appears in an array



Names	
0	Fred
1	Betty
2	Wilma
3	Betty
4	Barney

If used on the Names array above for the name "Betty", Count Occurrences would return the value, 2

```
1. RECEIVE target FROM KEYBOARD

2. SET numFound TO 0

3. FOR counter FROM 0 TO 4
4.        IF names[counter] = target
5.                 SET numFound TO numFound + 1
6.        END IF
7. END FOR

8. SEND "The number found is "& numFound TO DISPLAY
```

1. Ask user to enter target value to count

2. Initialise numFound to 0

3. Repeat for each item in array

4. Check if current name matches target

5. If true, **increment** numFound by 1

Count Occurrences (Using Record Structures)

Count Occurrences is used to identify how many times a particular value appears in an array of record structure.

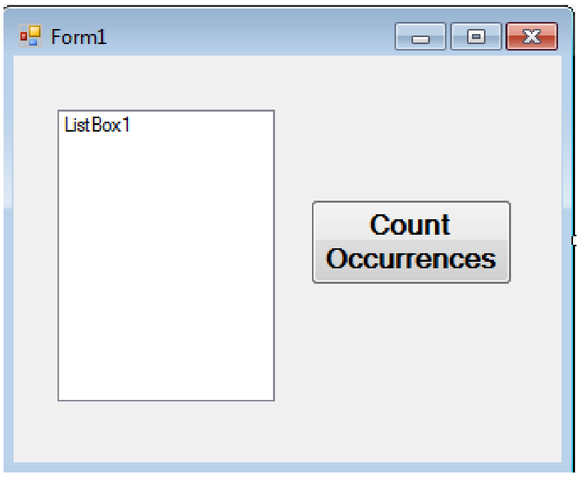
If used on the UserDetails record structure below for the name “Harry”, Count Occurrences would return the value, 2

RECORD *Userdetails* **IS**
{**STRING** Firstname, **STRING** Surname, **INTEGER** Age, **STRING** House}

First Name	Surname	Age	House
Harry	Jones	14	Bute
Jenna	White	12	Kintyre
Laura	Cairns	15	Arran
Sam	Kay	16	Arran
Harry	Smith	14	Lomond

```
1. RECEIVE target FROM KEYBOARD

2. SET numFound TO 0

3. FOR counter FROM 0 TO 4
4.      IF UserDetails [counter].names = target
5.           SET numFound TO numFound + 1
6.      END IF
7. END FOR

8. SEND “The number found is ”& numFound TO DISPLAY
```

Worked Example 5c – Count Occurrences



This example asks the user to enter 10 random names.

One of the names is then entered as the target to count.

The Count Occurrences algorithm is used to identify the number of times the target name appears in the list.

```
Public Class Form1

Private Sub btnCount_Click(sender As Object, e As EventArgs) Handles btnCount.Click


        Dim myList(10) As String
        Dim target As String
        Dim counter As Integer



        For index = 1 To 10
            myList(index) = InputBox("Please enter name  " & index)
        Next index





        target = InputBox("Please enter name to fine  ")

        counter = 0

        For index = 1 To 10

            If myList(index) = target Then
                counter = counter + 1
            End If
        Next index




        MsgBox("The name " & target & " appears  " & counter & " times ")


End Sub


End Class
```

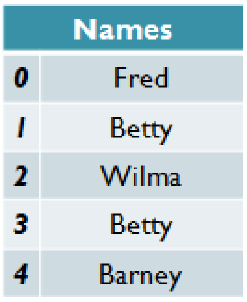
Fill array with names

Count Occurrences algorithm

Linear Search (Using Arrays)

Linear Search is used to identify whether or not an item is in a list, and which position it occupies.

If used on the Names array above for the name, "Betty", Linear Search would return the position, 3



	Names
0	Fred
1	Betty
2	Wilma
3	Betty
4	Barney

Betty
FOUND
at position 3

```
1. RECEIVE target FROM KEYBOARD
2. SET found TO FALSE
3. SET positionTO 0

4.   FOR counter FROM 0 TO 4
5.      IF names[counter] = target
6.          SET found TO TRUE
7.          SET position TO counter
8.      END IF
9.   END FOR

10. IF found = TRUE
11.     SEND "Found at position "& position TO DISPLAY
12. ELSE
13.     SEND "Not found"
14. END IF
```

1. Ask user to enter target value to find

2-3. Initialise found **"flag"** to False and position to 0

4. Repeat for each item in array

5. Check if current name matches target

6-7. If true, set found **"flag"** to true and position to current loop value

Linear Search (Using Record Structures)

Linear Search is used to identify whether or not an item is in a list, and which position it occupies.

If used on the UserDetails record structure below for the name, “Harry”, Linear Search would return the position, 4

RECORD *Userdetails* **IS**
{**STRING** Firstname, **STRING** Surname, **INTEGER** Age, **STRING** House}

First Name	Surname	Age	House
Harry	Jones	14	Bute
Jenna	White	12	Kintyre
Laura	Cairns	15	Arran
Sam	Kay	16	Arran
Harry	Smith	14	Lomond

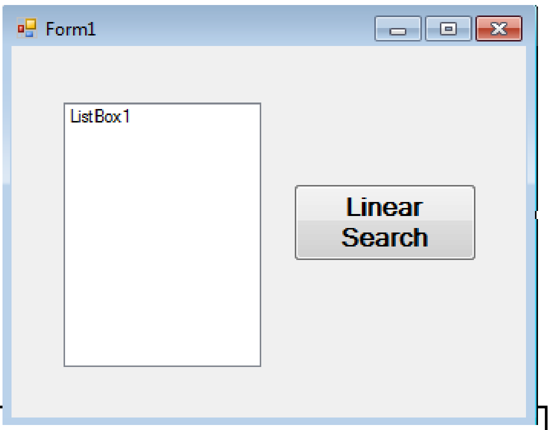
```
1. RECEIVE target FROM KEYBOARD
2. SET found TO FALSE
3. SET positionTO 0

4.   FOR counter FROM 0 TO 4
5.       IF UserDetails [counter].names = target
6.           SET found TO TRUE
7.           SET position TO counter
8.       END IF
9.   END FOR

10. IF found = TRUE
11.     SEND “Found at position ”& position TO DISPLAY
12. ELSE
13.     SEND “Not found”
14. END IF
```

Worked Example 5d – Linear Search

This example asks the user to enter 10 random names.

One of the names is then entered as the target to count.

The Linear Search algorithm is used to identify the position of the target name in the list.



```
Public Class Form1

    Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles
btnSearch.Click


        Dim myList(10) As String
        Dim target As String
        Dim found As Boolean
        Dim position As Integer


        For index = 1 To 10
            myList(index) = InputBox("Please enter name  " & index)
        Next index


        target = InputBox("Please enter name to fine  ")


        found = False

        For index = 1 To 10

            If myList(index) = target Then
                found = True
                position = index
            End If
        Next index



        If found = True Then

            MsgBox("Item was found at position " & position)
        Else
            MsgBox("Item was not found")
        End If


    End Sub
End Class
```

Fill array with names

Linear Search algorithm

Practise Tasks

1. Create a program that reads in from file the times (in seconds) for 6 athletes in a 100m sprint. The program should then identify the fastest time and the slowest time. Both these values should be written to a different file for later use.

2. A program is required that reads in from file the number of goals scored in each of 8 football matches on a Saturday. The program should identify the number of games where fewer than 3 goals were scored and the number of games where more than 6 goals were scored. Both these values should be written to a different file for later use.

3. Create a program that will read in from file a list of 7 schools and the town they are in. The program should then allow the user to enter the name of a town and the schools in that town will be displayed and written to file. If no schools are in the town entered, an appropriate message should be displayed.

Practise Questions

A 1D array stores a list of 8 scores as shown below.

Index	0	1	2	3	4	5	6	7
Scores	16	12	19	20	17	8	13	19

(a) Write an algorithm to identify and display the highest score in the list

(b) Write an algorithm to identify and display the lowest score in the list

(c) Write an algorithm to identify and display the number of scores over 15 in the list

(d) Write an algorithm to identify and display the position of the value 20 in the list.