Higher

Computing Science

Software Design and Development

Data Types & Structures

Name:_______________________________

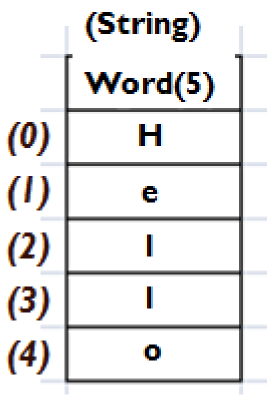
Contents

Data Types (Revision)

There are four main data types you need to know about:

Data Type	Contents	Example
CHARACTER	Single Letter	“A”, “B”, “C”
INTEGER	Whole Number	2, 15, 18, 100
SINGLE (REAL)	Real Number	2.45, 3.9, 12.994
BOOLEAN	True or False	TRUE / FALSE

Data Structures: String



A string is a special sort of array that contains characters. A string is actually a just a list of single characters.

Strings can be joined using concatenation or extracted using substrings.

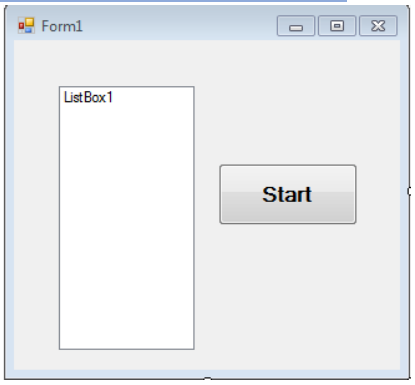
1D Arrays (revision)

A 1D array is an ordered sequence of simple data types, all of the same type.

	Names
(0)	Rose
(1)	Jack
(2)	Laura
(3)	James

Worked Example 1a – 1D Arrays (Revision)

This is a simple example to refresh your memory on how arrays work. This program allows five bands to be entered into different indexes in the array. The contents of each position are then added to the listbox. Finally, the user can choose which index to display.



```
Public Class Form1


Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim bands(5) As String
        Dim choice As Integer

        'input data into an array
        For index = 1 To 5
            bands(index) = InputBox("Enter the name of a band")
        Next


        'output all data from an array
        For index = 1 To 5
            ListBox1.Items.Add(bands(index))
        Next



        'output selected index from array
        choice = InputBox("Enter the position to display (1-5)")

        If choice >= 1 And choice <= 5 Then
            MsgBox("The band at position " & choice & " is " & bands(choice))
        Else
            MsgBox("The array does not have " & choice & " positions.")
        End If


End Sub
End Class
```

Parallel 1D Arrays

Parallel 1D arrays allows related data to be stored together

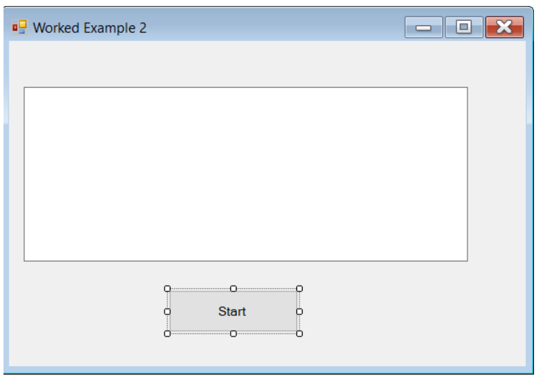
	Names		Ages		Houses
(0)	Rose	*(0)*	12	*(0)*	Arran
(1)	Jack	*(1)*	14	*(1)*	Bute
(2)	Laura	*(2)*	11	*(2)*	Bute
(3)	James	*(3)*	15	*(3)*	Cumbrae

In the example above, all the information about Rose is stored in index position 0 **in each array**.

It is important, to keep the data together, that related information is entered into the same index position for each array.

Worked Example 1b – Parallel 1D Arrays

This example allows 3 golfers to enter their first name and scores in two rounds of a golf tournament. To qualify, a golfer must have a combined total for both rounds of below the qualifying score. The program should ask for the qualifying score and calculate whether each golfer has qualified. All details for each golfer should then be displayed.



```
Public Class Form1

Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim golferName(3) As String
        Dim round1(3) As Integer
        Dim round2(3) As Integer
        Dim qualified(3) As String
        Dim qualScore As Integer
```

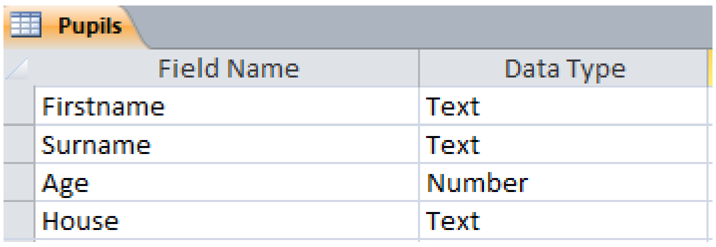
Parallel arrays to store four items of data about three golfers

```
        For index = 0 To 2
            golferName(index) = InputBox("Please enter your name")
            round1(index) = InputBox("Please enter round 1 score")
            round2(index) = InputBox("Please enter round 2 score")
        Next


        qualscore = InputBox("Please enter the qualifying score (cut)")



        For index = 0 To 2
            If round1(index) + round2(index) < qualscore Then
                qualified(index) = "Qualified"
            Else
                qualified(index) = "Not Qualified"
            End If
        Next


        txtOutput.AppendText("Golfer Name" & vbTab & "Round 1" & vbTab & "Round 2" &
        vbTab &
                              "Qualfied?" & vbNewLine)

        For index = 0 To 2

            txtOutput.AppendText(golferName(index) & vbTab & round1(index) & vbTab &
                                  round2(index) & vbTab & qualified(index) &
vbNewLine)

        Next

End Sub
End Class
```

Record Structure/Array of Records

Records are **customised data types** created by the programmer. They can contain **several variables** which can be of **different data types**.

When you create a record structure, you are essentially creating a database structure.



Pupils

Field Name	Data Type
Firstname	Text
Surname	Text
Age	Number
House	Text

Record Structure

A **record structure** is created by giving the structure a name and defining the 'fields' required.

```
RECORD  recordname  IS
        {datatype fieldname1, datatype fieldname2, datatype fieldname3…}
```

Array of Records

An **array of records** is then declared which specifies the size of the array and the record structure to use (as the data type):

```
DECLARE arrayname(indexes) AS recordname
```

Notice, instead of declaring the array using a data type such as integer or string, we have used the name of the record structure as the data type.

Benefits over Parallel 1D arrays

- Records make sure that related data always stays together. Because there is only one array, each index position has fields to enter all the related information.
- When passing parameters in a program, only one array has to be passed in or out of a subprogram instead of multiple arrays (one for each piece of data being stored).

Example:

Create a record structure to store the information below for 10 pupils:

First Name	Surname	Age	House
Harry	Jones	14	Bute
Jenna	White	12	Kintyre
Laura	Cairns	15	Arran

a) Defining the **record structure**:

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

b) Declare an **array of records**.

```
DECLARE UserRecord(10) As Userdetails
```

Record values can now be initialised or updated as a complete record

```
SET UserRecord[1] TO {“Harry”, “Jones”, 37}
```

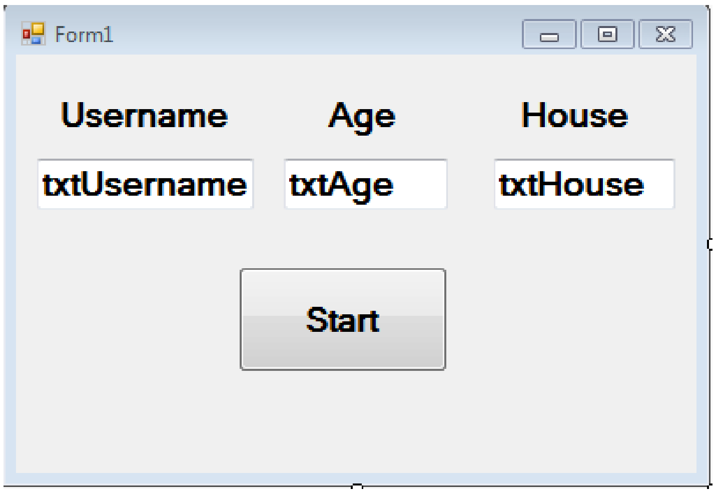
Or by referring to individual values

```
SET UserRecord[1].Firstname TO “Harry”

SET UserRecord[1].Surname TO “Jones”
```

Worked Example 2 – Record Structure

Records allow for a customised data type to be created that contains several variables of different types.

A variable or array is then declared which uses the records structure as its data type.



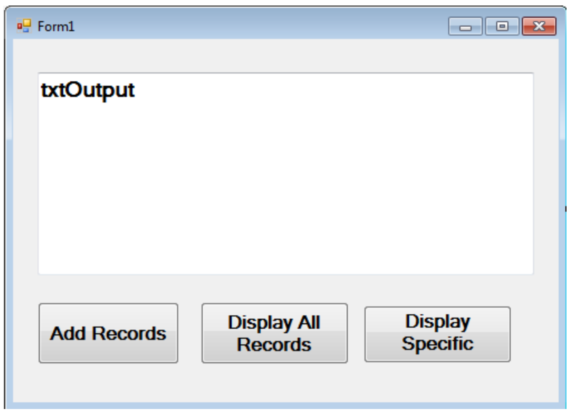
```
Public Class Form1

    Public Structure RecordDetails

        Dim username As String
        Dim age As Integer
        Dim house As String

    End Structure
```

```
Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click

        Dim mydetails As RecordDetails


        mydetails.username = InputBox("Enter a username")
        mydetails.age = InputBox("Enter your age")
        mydetails.house = InputBox("Enter your house")



        txtUsername.Text = mydetails.username
        txtAge.Text = mydetails.age
        txtHouse.Text = mydetails.house


    End Sub

End Class
```

Worked Example 3 – Array of Records

An array of records allows for more than one set of details to be stored as a record.

This example allows the user to display all records or specify an array index position to display only a chosen record.



```
Public Class Form1

    Public Structure myDatabase

        Dim firstname As String
        Dim surname As String
        Dim age As Integer

    End Structure

    Dim userdetails(4) As myDatabase
```

```
Private Sub btnAdd_Click(sender As Object, e As EventArgs) Handles btnAdd.Click


        For index = 1 To 4

            userdetails(index).firstname = InputBox("Enter your first name")
            userdetails(index).surname = InputBox("Enter your surname")
            userdetails(index).age = InputBox("Enter your age")

        Next


    End Sub
```

```
Private Sub btnDisplayAll_Click(sender As Object, e As EventArgs) Handles
btnDisplayAll.Click

        For index = 1 To 4

            txtOutput.AppendText(userdetails(index).firstname & vbTab &
                           userdetails(index).surname & vbTab &
userdetails(index).age
                     & vbNewLine)

        Next

    End Sub
```

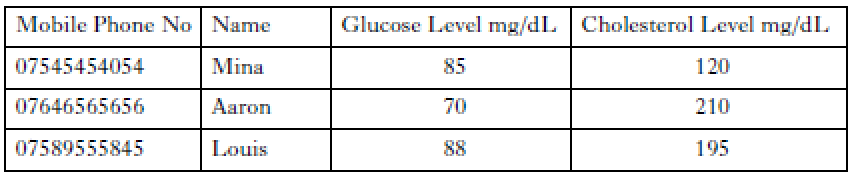
All goes on one line

```
Private Sub btnDisplayOne_Click(sender As Object, e As EventArgs) Handles
btnDisplayOne.Click

        Dim choice As Integer

        choice = InputBox("Enter record number to display (1-4)")

        txtOutput.AppendText(userdetails(choice).firstname & vbTab &
                     userdetails(choice).surname & vbTab & userdetails(choice).age
                     & vbNewLine)


End Sub


End Class
```

$\}$ **All goes on one line**

Practise Task

1. Create a record structure that allows the following details for 3 patients in a hospital to be stored.



Mobile Phone No	Name	Glucose Level mg/dL	Cholesterol Level mg/dL
07545454054	Mina	85	120
07646565656	Aaron	70	210
07589555845	Louis	88	195

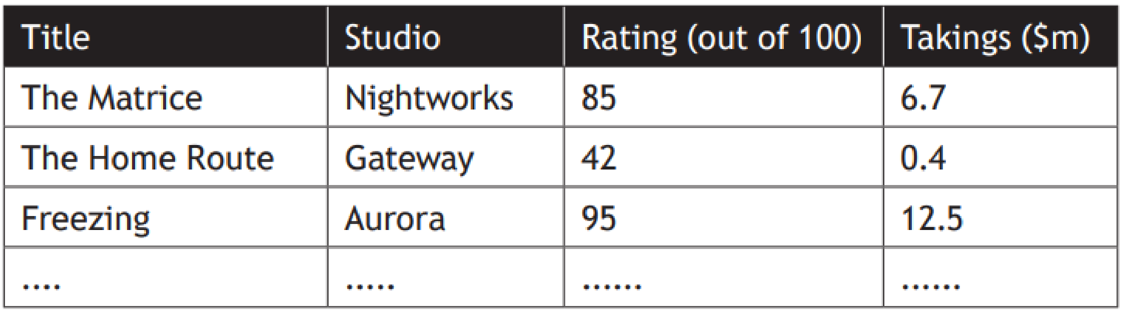
The program should allow:

- all details to be added at once
- a patient number (1 to 3) to be entered and their details displayed on screen

Practise Questions

Question 1 (2018 Qu 12b)

The app will have information on the top 100 movies of all time including the studio that made the movie, fan ratings and takings at the box office. For example



Title	Studio	Rating (out of 100)	Takings ($m)
The Matrice	Nightworks	85	6.7
The Home Route	Gateway	42	0.4
Freezing	Aurora	95	12.5
....	.....	......	......

a) Using pseudocode or a programming language of your choice, define a suitable record data structure for the movie data above. (2)

b) Using pseudocode or a programming language of your choice, declare the variable which can store the details of the top 100 movies. Your answer should use the record data structure created in part ().

Question 2 (2019 Qu 15a)

Two hundred competitors entered a regional orienteering competition in either the Junior or Senior category. Each competitor received a score based on their performance. The names, categories and scores are stored in a csv file called ‘competitors.csv’. Part of the file is shown below.

Senga Jones,Senior,67

Agnes Adam,Junior,88 …

A program is required to read the data from the csv file and then offer a menu of different options.

The data will be stored in parallel 1D arrays. Using a programming language of your choice, declare parallel 1D arrays that can store the data for the 200 competitors. (2)