Introduction to Programming

SAMPLE

Marking Scheme

This marking scheme has been prepared as a **guide only** to markers. This is not a set of model answers, or the exclusive answers to the questions, and there will frequently be alternative responses which will provide a valid answer. Markers are advised that, unless a question specifies that an answer be provided in a particular form, then an answer that is correct (factually or in practical terms) **must** be given the available marks.

If there is doubt as to the correctness of an answer, the relevant NCC Education materials should be the first authority.

Throughout the marking, please credit any valid alternative point.

Where markers award half marks in any part of a question, they should ensure that the total mark recorded for the question is rounded up to a whole mark.

Complete the following computer programming tasks using Visual Basic.

For each task you must test the programming code you create.

The evidence you **must produce is a Word document** containing the following for each task.

- Required outcomes
- Simple test criteria
- The Visual Basic program code, with relevant comments (i.e. annotated)
- Screenshots of the user interface
- Screenshots of the output generated when the program code is run

It would be a good idea to use the Windows Snipping tool to assemble this Word document.

You do **not** have to provide any general commentary on your work.
You do **not** have to provide any history of what debugging you may have done.

It is very important that the program code is clear, the outputs are designed to be easy to understand and your Word document is well presented.

A formula sheet containing the VB syntax for common programming structures is provided at the back of the examination paper and you may refer to this during the exam. Use of your own notes during the examination is not allowed.

Marks

Question 1

Write and test a program to convert acres to hectares. 1 acre = 0.404686 hectares. **10**
The program should accept user input via a textbox and display the output using a
label. The program should work for real numbers as well as whole numbers (e.g. it
should accept 7.2 acres, 0.5 acres, etc.).

Suitable user interface with textbox, button and labels
- **Award 2 marks for a neat interface with all the required components**
- **Award 1 mark if there are missing components or the layout is very poorly designed**
- **Award 0 marks if not attempted**

Data input and data types
- **Award 1 mark if program handles data input appropriately**
- **Award 1 mark if program stores data as acceptable data type (e.g. double or float)**

Calculating correct answer
- **Award 2 marks for the correct calculation**
- **Award 1 mark if the calculation is correct but there are rounding errors due to using integer data type**
- **Award 0 marks for incorrect formula**

Displaying correct answer
- **Award 1 mark if program successfully displays correct answer**

Testing with valid data
- **Award 1 mark if evidence is provided of testing with valid data**

Testing with invalid data
- **Award 2 marks for testing with two types of invalid data (e.g. with a string and with null input)**
- **Award 1 mark for testing one type of invalid data**
- **Award 0 marks if not attempted**

Total 10 Marks

Introduction to Programming © NCC Education Limited 2017

Question 2

Write and test a program that allows an athletics trainer to store the names of athletes   **50**
and the maximum distance they can throw the javelin in metres, to three decimal
places. The athletics trainer needs to store data about 16 athletes.
- This data should be stored in two separate arrays (one array for the athlete's
  names and another array for maximum distance).
- The athletes' names should be stored as strings.
- The maximum distances should be stored as real numbers.
- The input data should be validated before being entered into the array. Athlete
  names must be at least 2 characters long and cannot contain more than 35
  characters. Only values between 50 and 130 metres should be accepted for
  the maximum distance.
- If an invalid entry is attempted, a message should be displayed informing the
  user that the entry will not be stored and that the athlete name must be
  between 2 and 35 characters long and that only values between 50 and 130
  will be accepted for the maximum distance.
- Once the arrays are full, the 'Input data' button should be disabled and the
  'Show contents' button should be enabled which, when pressed, will print out
  the athlete name and maximum distance for each athlete, in the order entered,
  to a label.
- Complete an Object Definition Sheet for this task.

Suitable user interface with textbox, buttons and labels
- **Award 2 marks if the interface is well laid out, contains all required
  controls, and controls are disabled/invisible or enabled as appropriate
  (e.g. 'Input data' button and textboxes should be enabled, 'Show
  contents' button should be disabled/hidden at form load time)**
- **Award 1 mark if all components are included on the form but are poorly
  laid out or are not enabled/disabled as requested, or if a few components
  are missing**
- **Award 0 marks if most components missing or not attempted**

**Array of correct size and data type (string) correctly declared for the athlete
names.**
- **Award 2 marks if both size and data type are correct and declaration is
  correct syntax**
- **Award 1 mark if either the size or the data type are inappropriate**
- **Award 0 marks for incorrect declaration that will not compile**

**Array of correct size and data type (float or double) correctly declared for
maximum distance**
- **Award 2 marks if both size and data type are correct and declaration is
  correct syntax**
- **Award 1 mark if either the size or the data type are inappropriate**
- **Award 0 marks for incorrect declaration that will not compile**

Program accepts data input of athlete names
- **Award 1 mark if textbox is enabled**
- **Award 0 marks if not attempted or textbox is disabled**

Program accepts data input of maximum distance.
- **Award 2 marks if textbox is enabled and either the program handles data type conversion (from string to float or double) or the textbox properties are set so that they only accept input of the correct data type**
- **Award 1 mark if textbox is enabled but there is no limiting of data type input or explicit conversion to float or double**
- **Award 0 marks if not attempted**

Appropriate data validation of input for athlete names
- **Award 2 marks if validation successfully excludes athlete names shorter than 2 characters and longer than 35 characters (through selection statement(s))**
- **Award 1 mark if only one of the checks is successful**
- **Award 0 marks if not attempted or does not work for either of the checks**

Appropriate data validation of input for maximum distance
- **Award 2 marks if validation successfully excludes numbers lower than 50, numbers higher than 130 (through selection statement(s)) and data of the wrong type (either through textbox properties or a selection statement)**
- **Award 1 mark if one or two of the three checks are successful**
- **Award 0 marks if not attempted or if none of the checks are successful**

Display appropriate message for invalid input
- **Award 1 mark if an appropriate message is displayed when invalid input is entered**

Program fills arrays successfully
- **Award 1 mark for declaring counter and initialising appropriately to 0**
- **Award 1 mark for incrementing counter whenever the button is clicked and the data passes validation**
- **Award 1 mark for storing the athlete name data in the next vacant element of the athlete names array**
- **Award 1 mark for storing the maximum distance in the next vacant element of the maximum distance array**
- **Award 1 mark for preventing the possibility of trying to write to an out of bounds location (i.e. element 16 of either array, or any element not between 0–15).**
- **Award 1 mark for writing the data validation as a separate function rather than as part of the button click code**

Enabling and disabling controls appropriately
- **Award 1 mark for disabling the 'Input data' button and the textboxes once the array is full and enabling the 'Show contents' button once the arrays are full**
- **Award 0 marks if not attempted**

Use of a loop to step over the arrays and access the contents
- **Award 1 mark for declaring loop counter and initialising to 0**
- **Award 1 mark for correct continuation condition**
- **Award 1 mark for incrementing the loop correctly**
- **Award 1 mark for accessing the contents of the athlete names array**
- **Award 1 mark for accessing the contents of the maximum distance array**

Successfully displaying the contents of the array
- **Award 2 marks for writing the code to access and display the contents as a separate function rather than as part of the button click code**
- **Award 1 mark for successfully adding data from the athlete names array to a label**
- **Award 1 mark for successfully adding data from the maximum distance array to a label**
- **Award 2 marks for successfully pairing up the athlete names and maximum distance line by line on the label output**
- **Award 2 marks if all 16 athlete names and maximum distances are added and displayed in order**

Evidence provided of testing with valid data
- **Award 2 marks for two or more tests with valid data**
- **Award 1 mark if only one test with valid data**
- **Award 0 marks if not attempted**

Evidence provided of testing athlete names with extreme data (e.g. "AB", "ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJ")
- **Award 1 mark if there is evidence of testing with extreme data**
- **Award 0 marks if not attempted**

Evidence provided of testing maximum distance with extreme data (e.g. 49.999, 50.0, 130.0, 130.001)
- **Award 2 marks if both boundaries of the valid range of maximum distance are tested**
- **Award 1 mark if only one boundary is tested**
- **Award 0 marks if not attempted**

Evidence provided of testing athlete names with invalid data
- **Award 2 marks if suitable testing with two types of invalid data is attempted (e.g. a null value and a real number)**
- **Award 1 mark if only one type of invalid data is tested**
- **Award 0 marks if not attempted**

Evidence provided of testing maximum distance with invalid data
- **Award 2 marks if suitable testing with two types of invalid data is attempted (e.g. a null value and a string)**
- **Award 1 mark if only one type of invalid data is tested**
- **Award 0 marks if not attempted**

Code suitably commented
- **Award 3 marks if the code is fully commented for all variable declarations, functions, loops and selection statements**
- **Award 2 marks for a good level of commenting but one or two sections not fully commented**
- **Award 1 mark if the code is sparsely commented, or there are lots of comments but the comments are confusing or inaccurate**
- **Award 0 marks if there are no comments, or no helpful comments**

Object Definition Sheet
- **Award 4 marks for fully complete Object Definition Sheet with all non-default property values and all candidate-written event procedures**
- **Award 3 marks if one or two property values or procedures are omitted**
- **Award 2 marks if only partially completed (three or more entries missing but at least two correct entries)**
- **Award 1 mark if at least one correct entry**
- **Award 0 marks if not attempted**

Total 50 Marks

Question 3

The athletics trainer wishes to find the names of all athletes whose maximum javelin **15** throwing distance is at least 90 metres in order to invite these promising javelin throwers to participate in a more intensive training regime with the aim of entering them in national competitions.

Extend the program from Question 2 so that it includes a button which, when pressed, will identify the names of all athletes with a maximum javelin throwing distance of 90 metres or higher. The program should store, in a new array, the names of all the athletes who meet the throwing distance criteria, so that they can be contacted. The list of promising athletes should be displayed on a label.

Create the required GUI components
- **Award 2 marks for adding a suitably named button, and a label to display the list of athletes with a maximum javelin throw of 90 metres or greater**
- **Award 1 mark if only part of the additional required components are added**
- **Award 0 marks if no additional GUI components are added**

Loop over the arrays, find all athletes with a maximum distance of 90 or greater, and add them to the new array
- **Award 1 mark for declaring a string array to store the names of athletes with a maximum javelin throw of 90 metres or greater**
- **Award 1 mark for choosing a suitable size for the promising javelin throwers array (16 or lower, but not too low)**
- **Award 2 marks for looping over the maximum distances array**
- **Award 2 marks for a suitable selection statement to check if the maximum distance is greater than or equal to 90 metres**
- **Award 2 marks for accessing the athlete names array at the current counter value of the loop if the maximum distance is greater than or equal to 90 metres**
- **Award 2 marks for adding the athlete name to the new string array if the maximum distance is greater than or equal to 90 metres**

Loop over the sub arrays for promising athletes and print out to a label
- **Award 3 marks for successfully adding data from the promising athletes array to a label**

Total 15 Marks

Question 4

The athletics trainer wants a search facility added to the program to enable them to **10** search for an athlete and find their maximum javelin throw distance. Extend the program from Question 3 to add a textbox to enter an athlete name, a button which, when pressed, will run the search, and a label to display their maximum distance.

Create the required GUI components
- **Award 2 marks for adding a suitably named button, textbox, and a label to display the list of athletes with a maximum javelin throw of 90 metres or greater.**
- **Award 1 mark if only part of the additional required components are added**
- **Award 0 marks if no additional GUI components are added**

Linear search
- **Award 1 mark for declaring an integer variable to store the location that the athlete's name was found and initialising it to a suitable value e.g. -1**
- **Award 3 marks for looping over the athlete names array and comparing each name in turn against the search term.**
- **Award 1 mark for storing the identified element number**
- **Award 2 marks for accessing the maximum distance stored at the correct location**
- **Award 1 mark for displaying the maximum distance for the named athlete**

Total 10 Marks

Question 5

The athlete names and maximum distances for all athletes need to be stored **15**
permanently. Extend the program from Question 4 so that it writes out the athlete
name and corresponding maximum distance. These should be written out to a text file
called 'JavelinDistance'.

- The athlete names and their maximum distances should be paired on the
  same line, with the first athlete name followed by their maximum distance on
  the same line with no spaces between them, and so on for all athletes in the
  array.
- There should be a 'Save results' button, which will write the data to file when
  pressed. This button should not be enabled until the 'Find promising javelin
  throwers' button (or whatever name was given to the button for Question 3)
  has been pressed.
- Test the extended functionality of the program.
- **Award 1 mark for correct import statement for System.IO**
- **Award 1 mark for adding required GUI component ('Save results' button)**
- **Award 1 mark for having the 'Save results' button disabled when the
  program is started and enabling it after the 'Find promising javelin
  throwers' button has been pressed**
- **Award 2 marks for a variable to store the file-path and data validation to
  ensure it is not null**
- **Award 1 mark for correctly declaring a StreamWriter**
- **Award 1 mark for initialising StreamWriter passing in the filepath**
- **Award 2 marks for looping over both arrays and appending the relevant
  maximum distance to each athlete name for writing out to file in the
  required format**
- **Award 1 mark for using the WriteLine method to write the contents out to
  file**
- **Award 1 mark for closing the connection to the file**
- **Award 1 mark for evidence of creating the file**
- **Award 1 mark for use of a function or functions rather than placing the
  write to file code in the button click Sub**
- **Award 2 marks for evidence of testing (e.g. with null filename, with
  suitable filename but empty array, with suitable filename and full array
  with valid values)**

Total 15 Marks

Submission guidelines

Once you have completed the tasks, carefully check the contents of the Word file so that it is a complete record of your work and is presented according to the guidance notes above.

Submit this Word file for assessment via suitable media.

End of paper

Declaring variables

Variable declarations have the following format:

Dim [variable_name] As [data-type]

Code examples

```
Dim myWholeNumber As Integer
Dim myRealNumber as Double
Dim myName As String
```

For loops

For loops have the following format:

Dim [loop_counter_name] As Integer

```
For [loop_counter_name] = [start_num] To [end_num]
    'do the code here
Next
```

Code example

```
Dim counter As Integer

For counter = 0 To 5
    Console.WriteLine("The counter is at: " & counter)
Next
```

While loops

While loops have the following format:

```
While [condition]
    'do the code here
Next
```

Code example

```
Dim myNum As Integer

While myNum < 12
    Console.WriteLine("The value of myNum is: " & myNum)
    myNum=myNum+1
End While
```

If statements

If statements have the following format:

If [condition]
    'do the code here
End If

Code example

```
Dim myNum As Integer

If myNum < 12
    Console.WriteLine("The value of myNum is: " & myNum)
End While
```

Arrays

Arrays are declared and the number of elements set as follows:

Dim [array_name]([number_of_elements]) As [data_type]

Code example

```
Dim myNums(12) As Integer
```

To store a value in an array we need to indicate which element we want to store the value in:

```
myNums[0] = 15              'stores the value 15 in element 0 of the array
```

To retrieve a value from an array we need to indicate which element we wish to access:

```
temp = myNums[2]           'copy contents of element 2 of myNums to temp
```

To loop over an array using a for loop:

```
Dim counter As Integer
For counter = 0 to 11
    temp = myNums[counter]
    Console.WriteLine("The counter is at: " & counter)
Next
```

Learning Outcomes matrix

Question	Learning Outcomes assessed	Marker can differentiate between varying levels of achievement
1	2, 3	Yes
2	1, 2, 3, 4, 6	Yes
3	2, 3, 4, 6	Yes
4	2, 3, 4, 6	Yes
5	2, 3, 4, 5, 6	Yes

Grade descriptors

Learning Outcome	Pass	Merit	Distinction
Create project documentation	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses data capture and validation	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses sequential programming with different data types	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses iteration and selection constructs	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses file I/O	Demonstrate ability to perform the task	Demonstrate ability to perform the task consistently well	Demonstrate ability to perform the task to the highest standard
Implement a program that uses arrays	Demonstrate ability to perform the task	Demonstrate ability to perform the task consistently well	Demonstrate ability to perform the task to the highest standard