QUESTION 2: OBJECT-ORIENTED PROGRAMMING

Animals are transferred to the zoo from time to time. The administrator of the zoo requires software to assist in placing these animals in suitable available enclosures.

The animals at the zoo are classified into three categories according to their size:

- L – Large animal
- M – Medium-sized animal
- S – Small animal

Different animal species are housed in different sized enclosures.

The files required for this question can be found in the folder named **Question2_XXXX** where XXXX refers to the programming language you have studied. You have been provided with a text file named **DataQ2.txt** and an incomplete program that consists of:

- A class unit (Delphi)/object class (Java) which describes the attributes of an enclosure and contains some methods
- A main form unit (Delphi)/test class (Java)

The text file contains the data of an unknown number of enclosures in the zoo.

The details of each enclosure appear on one line with the data items separated by semicolons (;) and hash (#) characters in the following format:

```
<Type of animal>;<Number of animals currently in the enclosure>#<Size of the enclosure in square metres>;<Category of animals based on size>#
```

Example of the data for the first five enclosures in the text file **DataQ2.txt**:

```
Cheetah;3#80.2;L#
Ratel;7#50;S#
XXX;0#20;X#
Serval;5#80.75;M#
XXX;0#36;X#
```

NOTE: XXX denotes an empty enclosure and therefore the category of the animals is unknown and is denoted by an X.

Do the following:

- Rename the given folder for QUESTION 2 by replacing the name of the programming language you have studied with your examination number.

- **Delphi programmers:**

  - Open the given incomplete program file **Question2P.dpr**.
  - Add your examination number as a comment in the first line of both the class unit (**uQuest2**) and the main form unit (**Question2U**).

- **Java programmers:**
  - o Open the given incomplete object class **Quest2** and the test class **TestQuestion2**.
  - o Add your examination number as a comment in the first line of both the object class (**Quest2**) and the test class (**TestQuestion2**).

- Compile and execute the program. The interface displays three menu options as indicated in the section labelled **QUESTION 2** in **ANNEXURE B (Delphi)/ANNEXURE C (Java)**.

2.1 **Do the following to complete the code in the class unit (Delphi)/object class (Java):**

The given **uQuest2** unit (Delphi)/**Quest2** class (Java) contains the declaration of four attributes for an enclosure object and the set (mutator) and get (accessor) methods for these attributes.

Write code for additional methods as described below.

2.1.1 Write code for a **constructor** method using parameter values to initialise the following attributes:

- Type of animal (fAType/type)
- Total number of this animal type in the enclosure (fNumber/number)
- Size of the enclosure in square metres (fSize/size)
- The category of the animals in the enclosure according to their size, that is L, M or S. (fCat/cat) (4)

2.1.2 Write code for a method named **isSuitable** to determine whether an enclosure is suitable to house a specific group of animals. The method receives as parameters the number of animals the group consists of and their category based on their size. It returns a Boolean value.

The following applies:

- Animals can only be housed in an empty enclosure. An empty enclosure is indicated with "XXX" as animal type.

- The enclosure must be large enough to house the group of animals. The criteria for the size of the enclosures are (on the next page):

- o A large animal (L) needs a minimum space of 18 square metres.
- o A medium-sized animal (M) needs a minimum space of 12 square metres and a maximum space of less than 18 square metres.
- o A small animal (S) needs a minimum space of 7 square metres and a maximum space of less than 12 square metres. (7)

2.1.3 Write code for a **toString** method that will construct and return a string which includes labels and information about the object in the following format:

```
<Animal type>...<Category of the animals in the enclosure>
Enclosure size: <Enclosure size>
Number of animals:  <Number of animals in the enclosure>
<Blank line>
```

Example of the output of the first two objects when their strings are returned by the **toString** method and displayed:

```
Cheetah...L
Enclosure size: 80.2
Number of animals: 3

Ratel...S
Enclosure size: 50.0
Number of animals: 7

```

(4)

2.2 **Do the following to complete the code in the main form unit (Delphi)/test class (Java):**

2.2.1 Declare an array capable of storing 30 enclosure objects and a counter variable to keep track of the number of objects in the array. (2)

2.2.2 Write code to test whether the text file exists.

If the file exists, write code to read lines of text from the text file. For each line of text, extract the data, create an enclosure object and assign the object to the array.

If the text file does not exist, display a suitable message and terminate the program.

NOTE: The objects have to be assigned to the array before the menu options are displayed. (15)

2.2.3 Complete each menu option as follows:

Menu Option A

Write code to display a numbered list of all the information for all the enclosures using the **toString** method.

Example of the output of some of the enclosures:

```
List of all the enclosures
==========================
Enclosure number: 1
Cheetah...L
Enclosure size: 80.2
Number of animals: 3

Enclosure number: 2
Ratel...S
Enclosure size: 50.0
Number of animals: 7

Enclosure number: 3
XXX...X
Enclosure size: 20.0
Number of animals: 0

:
```

(4)

Menu Option B

A number of animals need to be transferred to the zoo and a suitable empty enclosure needs to be identified to house them.

Write code to allow the user to enter the following:

- The type of animal, for example Tiger
- The number of this type of animal
- The category of the animals based on size (for example L, M or S)

Use a conditional loop and the **isSuitable** method to search for a suitable empty enclosure in the array.

If a suitable empty enclosure is found, the attributes of the empty enclosure object in the array must be updated using the relevant set (mutator) methods. A message must be displayed indicating the enclosure number.

HINT: Use menu option A to display all enclosures to see whether the empty enclosure referred to in the message has been updated with the relevant information.

If a suitable empty enclosure is not found, an appropriate message must be displayed.

Test your program with the following test data:

Information for test data set 1:

Animal type: Tiger
Number of animals: 2
Size of the animals: L

Example of the output:

```
These animals were placed in enclosure number 5.

List of all the enclosures
==========================
:

Enclosure number: 5
Tiger...L
Enclosure size: 36.0
Number of animals: 2

:
```

Information for test data set 2:

Animal type: Meerkat
Number of animals: 12
Size of the animals: S

Example of the output:

```
No suitable enclosure was found.
```

(11)

- Make sure that your examination number is entered as a comment in the first line of the class unit (Delphi)/object class (Java) as well as the main form unit (Delphi)/test class (Java).
- Save all the files.
- A printout of the code will be required.
- Print both the class unit (Delphi)/object class (Java) and the main form unit (Delphi)/test class (Java).

[47]