Computer Science

A Level | AQA | 7517

AQA PAPER 1 EXAM RESOURCE PACK 2017

Rabbits and Foxes

for A Level AQA Computer Science

zigzageducation.co.uk

POD
7226

Contents

Teacher's Introduction

This pack is designed to help you support your students taking the A Level Computer Science Paper 1 examination. It is based on the 'Rabbits & Foxes' preliminary material (VB .NET) – for examination June 2017.

It consists of the following:

① **Pre-release Commentary** (for teachers)
A detailed overview of the skeleton program, describing all VB code elements and routines.

This section is designed to help you get to grips with the program, so that you can feel confident helping your students. This commentary is not designed to be given to students before they have explored the code for themselves, and if used in this way could lead to misconceptions of how the program works.

② **UML Diagram Activity**
A partially incomplete UML class diagram for students to complete while getting to grips with the skeleton program. Any missing operations and attributes must be added to the diagram. A completed version is provided in the solutions section at the back of the resource.

③ **Programming Theory Questions**
Theory questions test students' understanding of the 'Rabbits & Foxes' code, like Section C in the exam. These are provided in both write-on and non-write-on format.

④ **Programming Exercises**
Modification exercises put students' programming skills to the test, like Section D in the exam.
An Electronic Answer Document (EAD) and the modified VB code are provided on the CD.

Answers and solutions for the UML Diagram activity, theory questions and programming exercises are provided from page 22 onwards. Note that for the programming exercises in particular, these are example solutions and you must use your discretion to award marks accordingly where there are valid alternative solutions.

The **Appendices** contains some additional resources, including:

- Further modifications worksheet: a template for brainstorming further enhancements to the skeleton program. This is suggested as a group activity, so that students (and the teacher) can share their ideas, thus increasing the likelihood of covering every area that will come up in the exam.
- Electronic Answer Document (EAD) printout: hard copy version of the file on CD (for reference).

Enter the URL **zzed.uk/7226** in your web browser to download a folder containing the following:

- **MODIFIED_VB_CODE.txt** – text file containing the new and/or modified program code as shown in the mark scheme for section ④ (from page 25).
- **PAPER1_EAD.docx** – Electronic Answer Document for completing sections ③ and ④

This resource is intended to supplement your teaching only. It is the teacher's responsibility to decide how to use this resource to assist themselves and their students appropriately. You may simply wish to read this material to better inform yourself and to help you prepare your lessons and to give you ideas for your teaching. You may also consider whether it is appropriate to hand out some of the sheets for reference and to use some of the activities for classwork or homework. You may also consider whether it is appropriate to hand out the booklet to be worked through by your students more independently. As with all pre-release material, it is the teacher's responsibility to decide in what way to assist their students, and to decide how this resource in particular can be used to fit into that assistance.

The resources here are provided as an interpretation of the pre-release material. The author does not have any special knowledge of what to expect on any particular exam.

Rabbits and Fox

Description of the Program

The program is a simulation of rabbit population over time and how it is affecte

The world is represented by a grid in which each square can contain a rabbit wa rabbits live) or a fox, or both. F designates a fox, and a number designates a rab rabbits are in the warren).

The menu holds the following options:

- Run simulation with default settings
- Run simulation with custom settings
- Exit

The settings that can be changed in option 2 include:

- Landscape size
- Number or rabbit warrens at start
- Number of foxes at start
- Randomness (as a %)

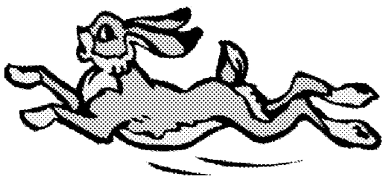
During the simulation you can advance to the next time period showing detail o current state of a fox or rabbit warren.

Each time a period runs, the rabbits can:

- Be eaten by a fox
- Be killed by something other than a fox
- Die of old age
- Increase in number (a number of new baby rabbits are born)

This information is displayed for each warren.

Each time a period runs there is a report on the foxes' age, how much food the f have eaten compared to what they need, and whether they have reproduced. If have reproduced, the location of the new foxes is displayed at the bottom.



Rabbits and Foxes

Description of Program Classes

This program contains multiple classes used to simulate foxes and rabbits in their natural environmen
The classes have been listed below, along with a brief description of their purpose.

Class	Description
Location	A class that creates an object corresponding to a location on the gr
Simulation	The class that drives the main simulation.
Warren	A class that simulates a rabbit warren (where they live).
Animal	An abstract class used for creating foxes and rabbits. It contains al
Fox (inherits Animal)	The class used to model foxes.
Rabbit (inherits Animal)	The class used to model rabbits.

Description of Class Variables

Each class has a number of variables, only accessible in that particular class. Each of the classes abov

Location — Instance variables	Type	Description
Fox	Fox	This value is equal to None when the simulat This value will hold a Fox object, if there is a f
Warren	Warren	This value is equal to None when the simulat This value will hold a Warren object, if there is

Simulation — Instance variables	Type	Description
ViewRabbits	String	Variable that should either have the value 'y' o
TimePeriod	Integer	Counter to store how many iterations of the si
WarrenCount	Integer	Variable that counts the number of warrens.
FoxCount	Integer	Variable that counts the number of foxes.
ShowDetail	Boolean	If this is true, more detail will be shown about
LandscapeSize	Integer	Value that stores the size of the Landscape (t
Variability	Integer	Value that determines how differently the sim other variable values.
FixedInitialLocations	Boolean	If True, the warrens and foxes will start in a fi
Landscape	Array	2D array of locations used to store foxes and w

Warren — Instance variables	Type	Description
MaxRabbitsInWarren	Integer	Constant that stores the maximum number of
RabbitCount	Integer	The value that stores the number of rabbits w
PeriodsRun	Integer	This variable stores how many periods have p
AlreadySpread	Boolean	Boolean value used to determine whether a r [illegible]e).
Variability	Integer	Value that determines how differently the sim other variable values.
Rabbits	Array	An array containing the rabbits that are curre

Animal — Instance variables	Type	Description
NaturalLifespan	Double	Integer value stating how long (in iteratio
ProbabilityOfDeathOtherCauses	Double	Decimal value used for calculating the cha
IsAlive	Boolean	Boolean value that states whether an anim
ID	Integer	Integer value given to uniquely identify th
Age	Integer	Value used to store the age of an animal (
NextID	*Integer*	*Value used to make sure that each new inst* *Note: this is a CLASS VARIABLE, shared by e*

Fox — Instance variables	Type	Description
DefaultLifespan	Integer	Value used for calculating the lifespan of t the variability variable in the Simulation c
DefaultProbabilityDeathOtherCauses	Double	Probability used for calculating the chanc in the Animal class using the variability va
FoodUnitsNeeded	Integer	Number of food units needed to stop the f
FoodUnitsConsumedThisPeriod	Integer	Number of food units that have been cons

Rabbit — Instance variables	Type	Description
DefaultLifespan	Integer	Constant used for calculating the lifespan using the Variability variable in the Simul
DefaultProbabilityDeathOtherCauses	Double	Probability used for calculating the chanc in the Animal class using the variability va
DefaultReproductionRate	Double	Constant used to set the default Reproduc
ReproductionRate	Double	Probability used for calculating the chanc
Genders	Enum	The gender of the rabbit, equal to either M

Description of Class Methods

Along with class variables, each class has a number of methods unique to that class. For each class, its fu

Location — Methods	Description	
New Ⓟ	Input: None Output: None	Creates 1. Ini 2. Ini

Simulation — Methods	Description	
New Ⓟ	Input: Size of landscape (Integer), initial number of warrens (Integer), initial number of foxes (Integer), variability (Integer), whether fixed locations should be used or not (Boolean) Output: None	Creates 1. Cre lar 2. Ad 3. Dr 4. Sta ge
InputCoordinate Ⓕ	Input: Coordinate name ('x' or 'y') Output: Coordinate (Integer)	Asks th coordin Return
AdvanceTimePeriod Ⓟ	Input: None Output: None	Update 1. Fo a. b. c. d. 2. For a. b. c. 3. If n

Simulation — Methods (cont.)	Description	
CreateLandscapeAndAnimals Ⓟ	Input: Initial number of warrens (Integer), initial number of foxes (Integer), whether fixed locations should be used or not (Boolean) Output: None	Creates 1. If th the 2. Oth det
CreateNewWarren Ⓟ	Input: None Output: None	Creates 1. Fin 2. Cre
CreateNewFox Ⓟ	Input: None Output: None	Creates 1. Fin 2. Cre
FoxesEatRabbitsInWarren Ⓟ	Input: Warren's x-coordinate (Integer), warren's y-coordinate (Integer) Output: None	Functio 1. For a. b. c.
DistanceBetween Ⓕ	Input: Two sets of x- and y-coordinates Output: Distance between the points (Double)	Calcula
DrawLandscape Ⓟ	Input: None Output: None	Draws t It check

Warren — Methods	Description	
New Ⓟ	Input: Variability (Integer), number of rabbits in warren (Integer) Output: None	Creates 1. Cre wa 2. If th init var 3. It a
CalculateRandomValue Ⓕ	Input: Base value (Integer), variability (Integer) Output: Random value (Integer)	Provide variabi

Warren — Methods (cont.)	Description	
GetRabbitCount Ⓕ	Input: None Output: Number of rabbits in warren (Integer)	Returns from.
NeedToCreateNewWarren Ⓕ	Input: None Output: Whether a new warren needs to be created (Boolean)	1. Ch bee 2. If th
WarrenHasDiedOut Ⓕ	Input: None Output: Whether a warren is empty or not (Boolean)	This fu 1. If th 2. Oth
AdvanceGen Ⓟ	Input: Whether you should show detail (Boolean) Output: None	Advance 1. If th 2. If th 3. If th con 4. Oth of t
EatRabbits Ⓕ	Input: Number of rabbits that need to be eaten (Integer) Output: Updated number of rabbits to be eaten (Integer)	Remove 1. Fin 2. Rem 3. Rep 4. Co
KillByOtherFactors Ⓟ	Input: Whether you should show detail (Boolean) Output: None	Kills ra random 1. Go 2. Che 3. Rem 4. Co
AgeRabbits Ⓟ	Input: Whether you should show detail (Boolean) Output: None	Makes 1. Go 2. De a.

Warren — Methods (cont.)	Description	
MateRabbits Ⓟ	Input: Whether you should show detail (Boolean) Output: None	Func 1. G 2. If a b c
CompressRabbitList Ⓟ	Input: Number of dead rabbits (Integer) Output: None	Shifts
ContainsMales Ⓕ	Input: None Output: Whether a warren contains males (Boolean)	Check 1. It 2. If
Inspect Ⓟ	Input: None Output: None	Prints
ListRabbits Ⓟ	Input: None Output: None	Prints

Animal — Methods	Description	
New Ⓟ	Input: Average lifespan (Integer), average probability of dying from other causes (Double), variability (Integer) Output: None	Cons
CalculateNewAge Ⓟ	Input: None Output: None	Incre
CheckIfDead Ⓕ	Input: None Output: Boolean	Whet
Inspect Ⓟ	Input: None Output: None	Prints
CheckIfKilledByOtherFactor Ⓕ	Input: None Output: Boolean	Deter
CalculateRandomValue Ⓕ	Input: Base value (Integer), variability (Integer) Output: Double	Calcu

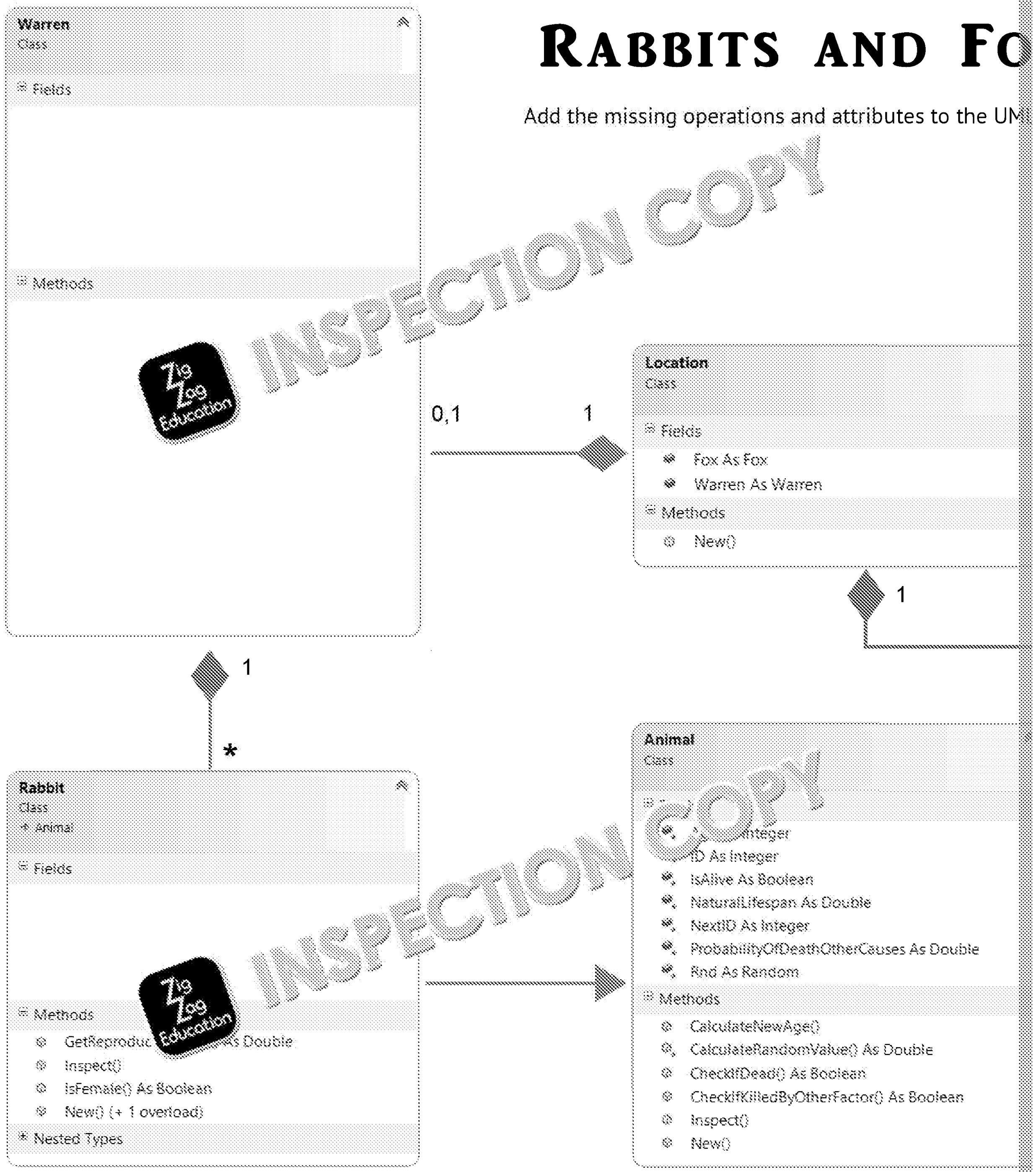
Fox — Methods	Description	
New Ⓟ	Input: Variability (Integer) Output: None	Constructor
AdvanceGeneration Ⓟ	Input: Whether detail should be shown (Boolean) Output: None	Determines
ResetFoodConsumed Ⓟ	Input: None Output: None	Resets this
ReproduceThisPeriod Ⓕ	Input: None Output: Boolean	Determines
GiveFood Ⓟ	Input: Number of food units (Integer) Output: None	Adds the nu
Inspect Ⓟ	Input: None Output: None	Prints out th

Rabbit — Methods	Description	
New Ⓟ	Input: Variability (Integer), parents reproduction rate (Double) Output: None	Constructor
Inspect Ⓟ	Input: None Output: None	Print out the
IsFemale Ⓕ	Input: None Output: Boolean	Returns whe
GetReproductionRate Ⓕ	Input: None Output: Reproduction rate (Double)	Returns the

In addition to the functions and procedures found in the classes, there is also the main program.

Rabbits and Fo

Add the missing operations and attributes to the UM



Programming Theory Questio

These questions refer to the Preliminary Material and require you to load the Skelet but do not require any additional programming.

1. Give an example of instantiation from the skeleton program.

   ....

2. State the name of an identifier(s) for the following:

   a. An array variable

   ....

   b. A subclass

   ....

   c. A parent class

   ....

   d. A class variable

   ....

   e. An accessor method

   ....

   f. A mutator method

   ....

   g. A variable used to store a whole number

   ....

   h. A Boolean variable

   ....

   i. Four constants that store a float

   ....

   ....

3 a. Two classes that have a composition aggregation relationship.

   ....

   b. Why is Warren to Rabbit not an example of association aggregation?

   ....

   ....

4. Are there any examples of polymorphism in the skeleton code?

....

5. State the name of an identifier for a procedure or function that is overridde

....

6. Look at the EatRabbits subroutine in the Warren class in the skeleton progr
Why does the generation of a random rabbit need to be inside a repetition s

....

7. Look at the Warren class. Why has a named constant been used instead of a

....

....

....

8. State the name of an identifier for an enumerated data type.

....

9. How could the Fox class be changed to make the foxes live longer?

....

10. What is the purpose of the variable AlreadySpread in the Warren class and h

....

....

....

....

....

11. What is the purpose of the method CompressRabbits?

....

....

....

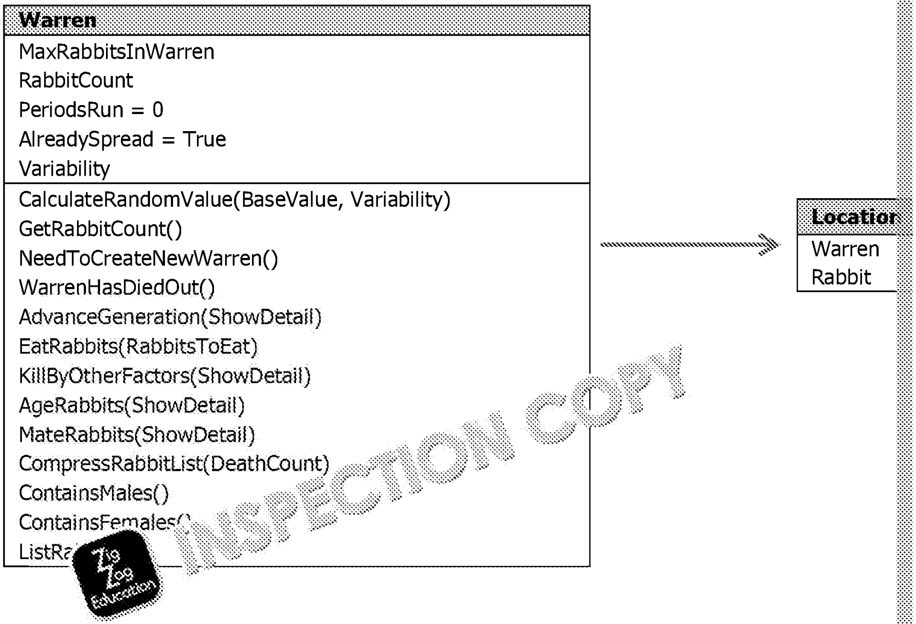
12. Why is it necessary to store the gender of the rabbits?

....

....

....

13. Identify six errors in the section of UML diagram below.



1 ....

2 ....

3 ....

4 ....

5 ....

6 ....

14. Create a UML diagram to show the relationship between rabbits, foxes and

All variables and methods must be shown.

15. What conditions are needed for a new warren to be created? [2 marks]

....

....

....

Programming Theory Questio

These questions refer to the Preliminary Material and require you to load
but do not require any additional programming.

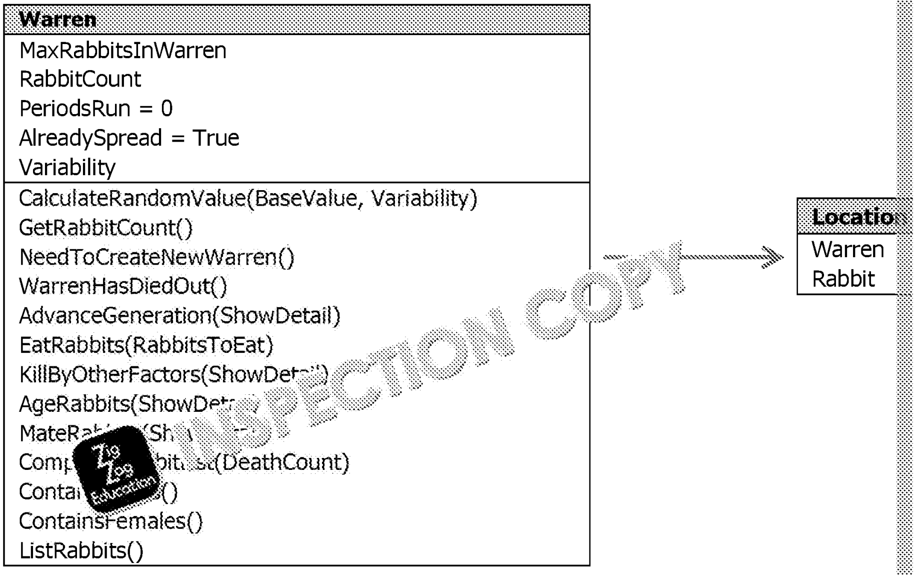
1. Give an example of instantiation from the skeleton program.
2. State the name of an identifier(s) for the following:

    a. An array variable [1 mark]
    b. A subclass [1
    c. A parent class [1 mark]
    d. A class varia
    e. An accessor method [1 mark]
    f. A mutator m
    g. A variable used to store a whole number [1 mark]
    h. A Boolean va
    i. Four constants that store a float [4 marks]

3 a. Two classes that have a composition aggregation relationship.

  b. Why is Warren to Rabbit not an example of association aggregation?

4. Are there examples of polymorphism in the skeleton code?
5. State the name of an identifier for a procedure or function that is overridde
6. Look at the EatRabbits subroutine in the Warren class in the skeleton progr
   Why does the generation of a random rabbit need to be inside a repetition s
7. Look at the Warren class. Why has a named constant been used instead of a
8. State the name of an identifier for an enumerated data type.
9. How could the Fox class be changed to make the foxes live longer?
10. What is the purpose of the variable AlreadySpread in the Warren class and h
11. What is the purpose of the method CompressRabbitList?
12. Why is it necessary to store the gender of the rabbits?
13. Identify six errors in the section of UML diagram below.



Warren

MaxRabbitsInWarren
RabbitCount
PeriodsRun = 0
AlreadySpread = True
Variability

CalculateRandomValue(BaseValue, Variability)
GetRabbitCount()
NeedToCreateNewWarren()
WarrenHasDiedOut()
AdvanceGeneration(ShowDetail)
EatRabbits(RabbitsToEat)
KillByOtherFactors(ShowDetail)
AgeRabbits(ShowDetail)
MateRabbits(ShowDetail)
CompressRabbitList(DeathCount)
ContainsMales()
ContainsFemales()
ListRabbits()

Locatio

Warren
Rabbit

14. Create a UML diagram to show the relationship between rabbits, foxes and
    All variables and methods must be shown.
15. What conditions are needed for a new warren to be created?

Programming Exercises

The following require you to open the skeleton program and make modifications. Th
and illustrate how you should prepare your answer

Question 1

This task refers to the Main procedure

Alter how the menu displays so that:

- There is a new option '3. Rabbit Paradise'
- The 'Exit' option is now numbered 4

Evidence you need to provide:

- Copy of your amended code
- Screen capture of it executing

Question 2

This task refers to the Main procedure

Code option 3 so that when it is selected the simulation is run with the followin

- A landscape size of 20
- 20 warrens
- 0 foxes
- Locations are not fixed
- Variability is 1

Evidence you need to provide:

- Copy of your amended code
- Screen capture of it executing

Question 3

This task refers to the Simulation class

Add an option to the game menu:
'0. Advance time periods hiding detail'

Code this option.

Evidence you need to provide:

- Copy of your amended code
- Screen capture of it executing

Question 4

This task refers to the Rabbit class

Change *Rabbit's* constructor so that it receives in an extra variable that will allow rabbits to be altered. Use the identifier *genderRatio* for the new variable.

Set the default value to 50 so that the constructor can be called without specifyi

Evidence you need to provide:

- Copy of your amended code

Question 5

This task refers to the Fox class

Add *Gender* to the *Fox* class.

Make the ratio of males to females 1 : 2.

Alter the *Inspect* method so that the gender of a fox is reported.

Change *ReproduceThisPeriod* so that only female foxes can reproduce.

Evidence you need to provide:

- Copy of your amended code
- Screen capture of an inspection of the Fox at 2,10

Question 6

A new subclass must be created for this task, as well as changes to the createLan in Simulation

Create a subclass of *Warren* called a *GiantWarren*.

- A giant warren has a maximum capacity of 200 and can always spawn a ne already.
- A giant warren has a default rabbit.
- Add a giant warren to the default game at position (11,4) with a starting p

Evidence you need to provide:

- Copy of your amended code
- Screen capture of a default simulation executing

Question 7

A new subclass must be created for this task, as well as changes to the Location c
createLandscapeAndAnimals, drawLandscape and AdvanceTimePeriod procedu

Create a *Den* class that can exist in a location.

- The den will spawn 1 new fox per 3 time periods.
- The den will store how many foxes it has created as a private instance vari
- The fox will appear at a random position.
- If there is already a fox in this location, it is replaced by the new fox.
- Position the den at (2,3) in a default game.
- The den will be displayed on the map as a D plus the number of foxes it ha

Evidence you need to provide:

- Copy of your amended code
- Screen capture at time period 3 of a default game running

Question 8

This tasks refers to the Fox class

The average age of death of foxes needs to be known.

- Create a class variable called *_TotalDeadFoxes* to store the total foxes wh
- Create a class variable called *_TotalFoxAge* to store the sum of the ages o
- When a fox dies, the *_TotalDeadFoxes* needs to be incremented and its ag
- An accessor method in Fox called *getLifeExpect* will return the average ag
- A message stating 'The average life expectancy of a fox stands at X' should each time it is displayed.
- If no foxes have yet died, the default lifespan should be returned.

Evidence you need to provide:

- Copy of your amended code
- Screen capture of default simulation at time period 0
- Screen capture of default simulation at time period 4

Question 9

This task refers to the Simulation class

Create a menu option in the simulation: '6. Find biggest warren′ .

The coordinates of the biggest warren will then be displayed: 'Biggest warren at

Create a new procedure called findBiggest to search the warren array in a linear message.

Evidence you need to provide:

- Copy of your amended code
- Screen capture of option 6 running

Question 10

This task refers to the Rabbit class

Make rabbit death probability go up by 10% with age.

Evidence you need to provide:

- Copy of your amended code
- Screen capture of a warren inspected (showing individual rabbits) at time p

Question 11

This task requires changes to Warren and Simulation classes

Create a menu option: '7. Inspect all rabbits'.

It should display a list of all rabbits in all warrens, showing their details.

An accessor method to get the rabbits list out of a warren must be created.

Evidence you need to provide:

- Copy of your amended code
- Screen capture of option 7 running

13 marks

Question 12

This task requires changes to Simulation as well as creation of new classes

Beneath the warrens are secret tunnels connecting them. Not every warren is c warren is connected to more than two other warrens. This data must be stored i

WarrenGraph
-nodes[]

+addNode(theNode)
+adjList()

Node
-selfX -selfY -leftBranch -leftBranchY -rightBranchX -rightBranchY
+getCoord(l/r/s)

Each warren connected to another has the coordinates of itself and its connecti *WarrenGraph* contains a list of all nodes. The procedure *getCoord* returns the x- based on arguments (l)eft, (r)ight and (s)elf.

The *adjList* method displays an adjacency list and should be executed by a new

The following data should be used to initially populate the graph.

self	left	right
(1,1)	(2,8)	(9,7)
(2,8)	(13,4)	(1,1)
(9,7)	(1,1)	(13,4)
(13,4)	(9,7)	(2,8)

Evidence you need to pr
- Copy o a code
- Screen e of option 8 running

Question 13

This task requires changes to Simulation and WarrenGraph

Create a new procedure in *WarrenGraph* called *adjMatrix*. It will display the gra

will be executed by '9. Display adjacency matrix'. A 1 should be used to indicate

Evidence you need to provide:

- Copy of your amended code
- Screen capture of option 9 running

Question 14

This task requires changes to WarrenGraph

Amend your solution for task 13 to replace the '1' with the actual distance betw

Use Pythagoras' theorem to calculate the distance between the two points.

Distances should be rounded to 1 decimal place.

Evidence you need to provide:

- Copy of your amended code for adjMatrix
- Screen shot of option 9 running

Question 15

This task requires changes to Simulation and WarrenGraph

Create a procedure to find whether there is a route between two warrens.

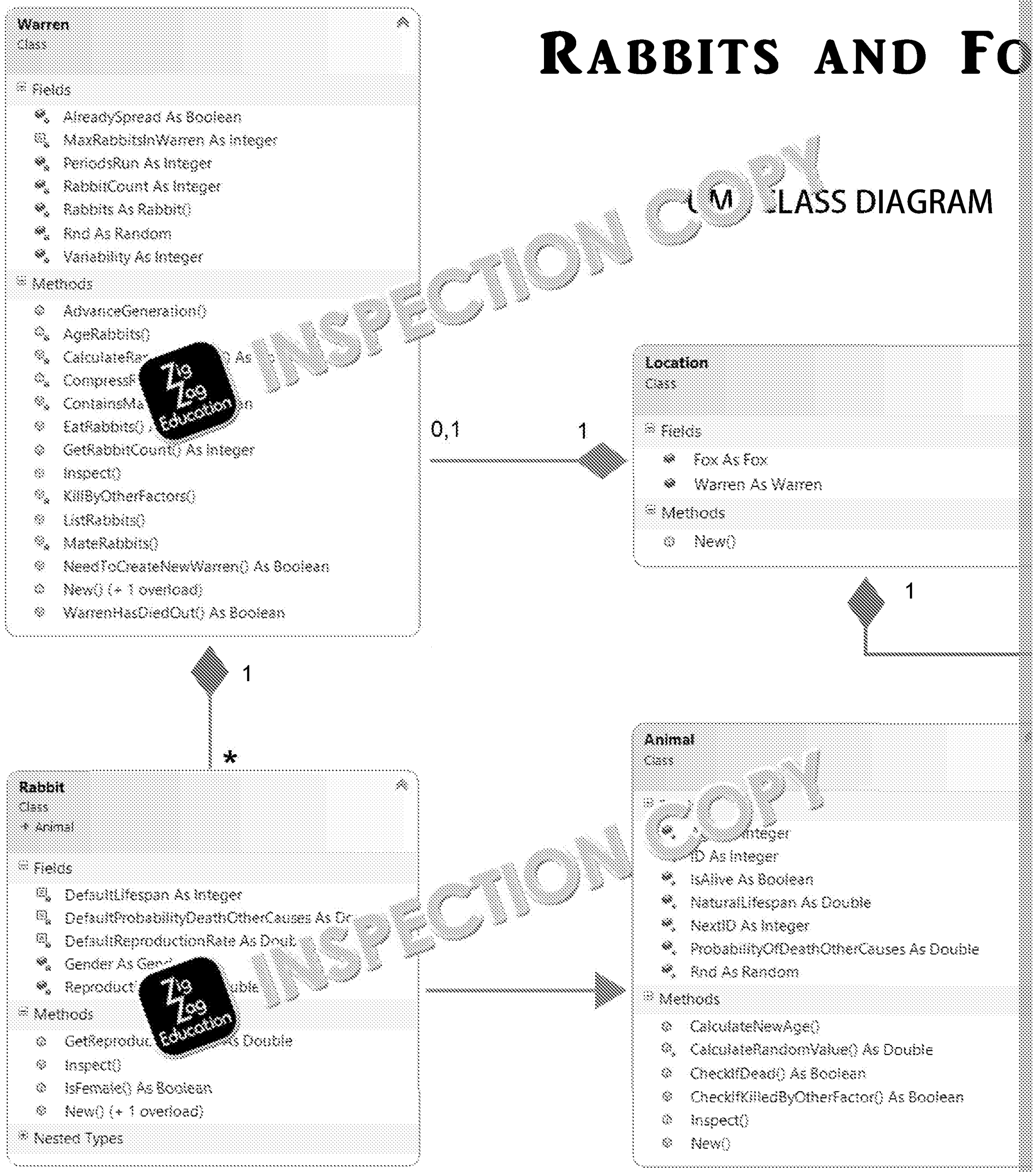
It will be executed by Option 10.

Evidence you need to provide:

- Copy of your code
- Screen capture of option 10 running showing no route between warrens
- Screen capture of option 10 running showing a route between warrens

RABBITS AND FO

UML CLASS DIAGRAM

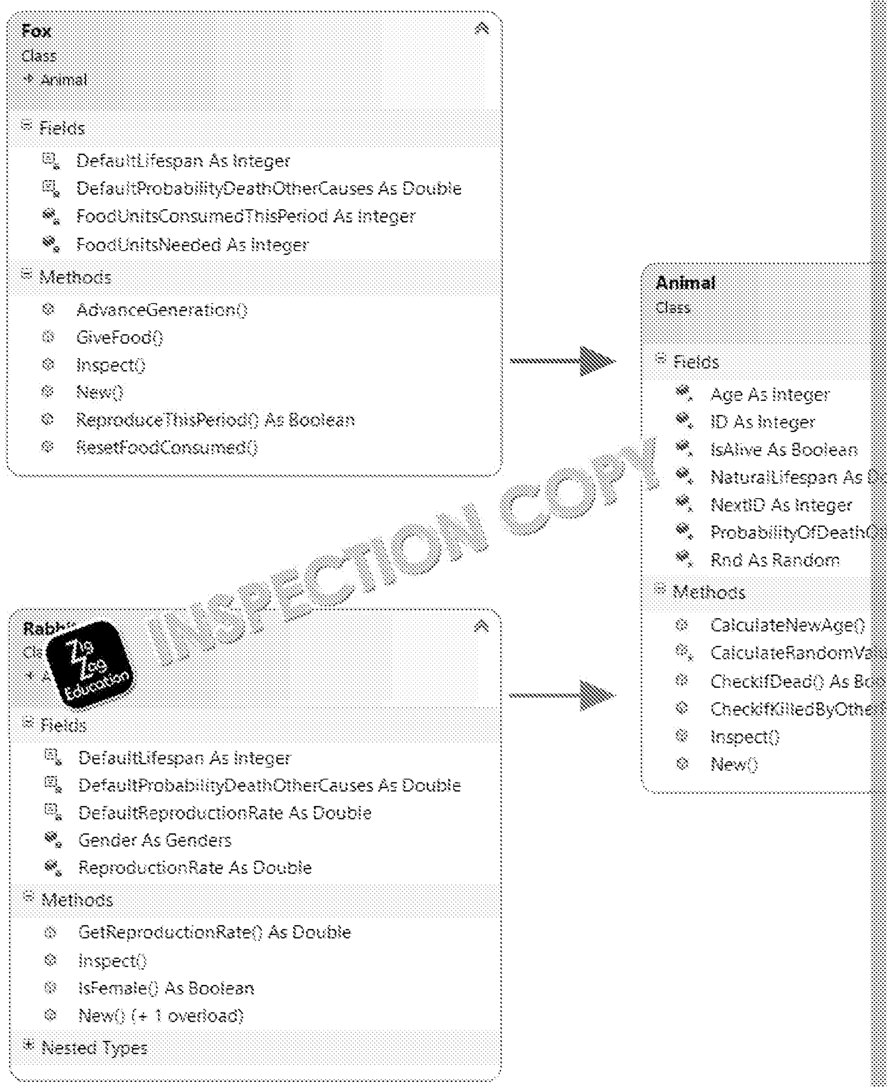


Programming Theory Questions (Suggested Answers)

Q	Marking Guidance

1	Dim Sim As New Simulation(LandscapeSize, InitialWarrenCount, InitialFoxCount, Vari
	Landscape(x, y).Warren = New Warren(Variability)
	Landscape(x, y).Fox = New Fox(Variability)
	Rabbits = New Rabbit(MaxRabbitsInWarren)

2a	Landscape / Rabbits
2b	Fox / Rabbit
2c	Animal
2d	NextID
2e	Any procedures/functions with Get at the start of the identifier
2f	Any procedures with Set at the start of the identifier
2g	MenuOption / LandscapeSize / InitialWarrenCount / InitialFoxCount / Variabilit Or any other local variable
2h	FixedInitialLocations OR ShowDetail OR AlreadySpread OR Males OR IsAlive
2i	DefaultProbabilityDeathOtherCauses ReproductionProbability DefaultReproductionRate DefaultProbabilityDeathOtherCauses
3a	Location to Fox or Location to Warren or Warren to Rabbit (any correct pair for
3b	Rabbit objects cannot exist unless they have an associated Warren
4	Yes – the constructor for Rabbit
5	Inspect
6	To keep selecting a different rabbit at random until the required number of rab
7	Makes the program code easier to understand / improves readability Makes it easier to update the program Makes it easier to change the maximum number of rabbits in a warren ANY 2
8	Gender
9	The DefaultLifeSpan constant needs to be increased from 7
10	It stores whether or not the warren has already created a new warren It stops the warren creating more than 1 new warren It is set to False by default It is set to True when a new warren is created
11	When rabbits are eaten or die they are removed from random positions in the l Compressing rabbits list removes the gaps
12	Only female rabbits can reproduce This therefore affects the calculation for how many new baby rabbits are born
13	Type and direction of arrow wrong Warren should not inherit from Location Location is associated to Warren Location stores warrens and/or foxes Location cannot store rabbits AlreadySpread should be set to False as default The constant MaxRabbitsInWarren has a default value of 99 Warren should contain a list of rabbits The Inspect() procedure is missing There is no function called ContainsFemales() in Warren ANY 6

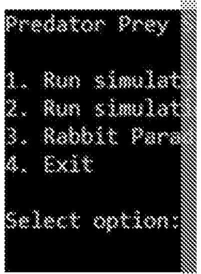
Q	Marking Guidance
14	 Fox Class → Animal Fields DefaultLifespan As Integer DefaultProbabilityDeathOtherCauses As Double FoodUnitsConsumedThisPeriod As Integer FoodUnitsNeeded As Integer Methods AdvanceGeneration() GiveFood() Inspect() New() ReproduceThisPeriod() As Boolean ResetFoodConsumed() Animal Class Fields Age As Integer ID As Integer IsAlive As Boolean NaturalLifespan As D NextID As Integer ProbabilityOfDeathO Rnd As Random Methods CalculateNewAge() CalculateRandomVa CheckIfDead() As Bo CheckIfKilledByOthe Inspect() New() Rabb Cl → A Fields DefaultLifespan As Integer DefaultProbabilityDeathOtherCauses As Double DefaultReproductionRate As Double Gender As Genders ReproductionRate As Double Methods GetReproductionRate() As Double Inspect() IsFemale() As Boolean New() (+ 1 overload) Nested Types 1 mark for correct class name (×3) 1 mark for correct instance variables (×3) 1 mark for correct methods (×3) 1 mark for correct inheritance arrows (×2)
15	The number of rabbits in the warren must have reached the maximum allowed The warren cannot have already created a new warren

Programming Exercises (Solutions)

Q	Example Solution

1

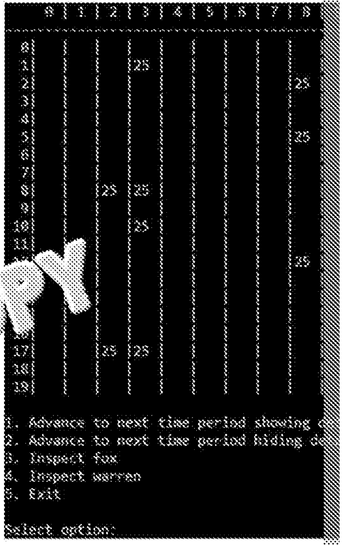
```
Sub Main()
    ...
      ...
      Console.WriteLine("1. Run simulation with default settings")
      Console.WriteLine("2. Run simulation with custom settings")
      Console.WriteLine("3. Rabb[illegible]
      Console.WriteLin[illegible]
      Console.W[illegible]



    Loop While MenuOption <> 4
    Console.ReadKey()
  End Sub
```



```
Predator Prey

1. Run simulat
2. Run simulat
3. Rabbit Para
4. Exit

Select option:
```

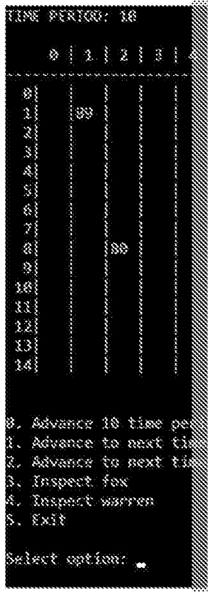
2

```
    If MenuOption = 1 Or MenuOption = 2 Or MenuOption = 3 Then
      ...

          ...
          FixedInitialLocations = True
        ElseIf MenuOption = 3 Then
          LandscapeSize = 20
          InitialWarrenCount = 20
          InitialFoxCount = 0
          Variability = 1
          FixedInitialLocations = False
        Else
```



```
      0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8
-------------------------------------------
  0|   |   |   |   |   |   |   |   |
  1|   |   |   |25 |   |   |   |   |
  2|   |   |   |   |   |   |   |   |25
  3|   |   |   |   |   |   |   |   |
  4|   |   |   |   |   |   |   |   |
  5|   |   |   |   |   |   |   |   |25
  6|   |   |   |   |   |   |   |   |
  7|   |   |   |   |   |   |   |   |
  8|   |   |25 |25 |   |   |   |   |
  9|   |   |   |   |   |   |   |   |
 10|   |   |   |25 |   |   |   |   |
 11|   |   |   |   |   |   |   |   |
   |   |   |   |   |   |   |   |   |25



 17|   |   |25 |25 |   |   |   |   |
 18|   |   |   |   |   |   |   |   |
 19|   |   |   |   |   |   |   |   |

1. Advance to next time period showing d
2. Advance to next time period hiding de
3. Inspect fox
4. Inspect warren
5. Exit

Select option:
```

Q	Example Solution

3	Class Simulation
	  ...
	    ...
	    Do
	      Console.WriteLine()
	      Console.WriteLine("0. Advance 10 time periods hiding detail")
	      Console.WriteLine("1. Advance to next time period showinc
	      ...
	      MenuOption = CInt(Console.ReadLine())
	      If MenuOption = 0 Then
	        ShowDetail = False
	          For x = 1 To 1
	            ePe
	            nce ..nePeriod()
	      Enu If
	      If MenuOption = 1 Then
	        ...
	
	TIME PERIOD: 10
	
	      0 | 1 | 2 | 3 | 4
	----------------------- 	
	   0| | | | |
	   1| |89 | | |
	   2| | | | |
	   3| | | | |
	   4| | | | |
	   5| | | | |
	   6| | | | |
	   7| | | | |
	   8| | |80 | |
	   9| | | | |
	  10| | | | |
	  11| | | | |
	  12| | | | |
	  13| | | | |
	  14| | | | |
	
	0. Advance 10 time pe
	1. Advance to next ti
	2. Advance to next ti
	3. Inspect fox
	4. Inspect warren
	5. Exit
	
	Select option:

4	Public Sub New(ByVal Variability As Integer, **ByVal GenderRatio As Integer**)
	      MyBase.New(DefaultLifespan, DefaultProbabilityDeathOtherCauses, Variability)
	      ReproductionRate = DefaultReproductionRate * MyBase.CalculateRandomValue(100, Varia
	      If Rnd.Next(0, 100) < **GenderRatio** Then
	        Gender = Genders.Male
	      ...

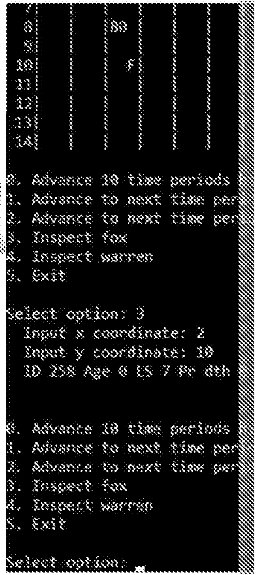
5	Class Fox
	    Inherits Animal
	    Enum Genders
	      Male
	      Female
	    End Enum
	    ...
	    ...
	    en As Genders
	    New(ByVal Variability As Integer)

Q	Example Solution
5 (cont.)	

```
    MyBase.New(DefaultLifespan, DefaultProbabilityDeathOtherCauses, Variability)
    FoodUnitsNeeded = CInt(10 * MyBase.CalculateRandomValue(100, Variability) / 100)
    If Rnd.Next(1, 3) = 1 Then
      Gender = Genders.Male
    Else
      Gender = Genders.Female
    End If
  End Sub
  ...
```

```
  Public Function ReproduceThisPeriod() As Boolean
    Const ReproductionProbability As Double = 0.25
    If Rnd.Next(0, 100) < ReproductionProbability * 100 And Gender = Genders.Female Then
      Return True
    ...
```

```
  Public Overrides Sub Inspect()
    ...
    ...
    Console.Write("Food eaten " & FoodUnitsConsumedThisPeriod & " ")
    If Gender = Genders.Female Then
      Console.WriteLine("Gender Female")
    Else
      Console.WriteLine("Gender Male")
    End If
    Console.WriteLine()
  ...
```

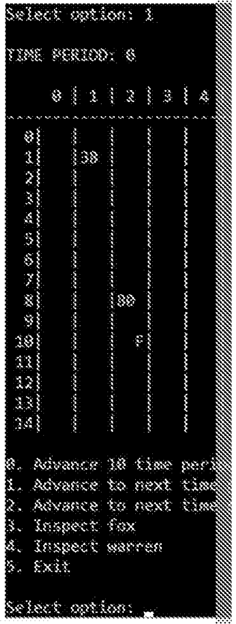


```
  7|     |     |     |     |     |
  8|     |     |S9   |     |     |
  9|     |     |     |     |     |
 10|     |     |    F|     |     |
 11|     |     |     |     |     |
 12|     |     |     |     |     |
 13|     |     |     |     |     |
 14|     |     |     |     |     |

0. Advance 10 time periods
1. Advance to next time per
2. Advance to next time per
3. Inspect fox
4. Inspect warren
5. Exit

Select option: 3
  Input x coordinate: 2
  Input y coordinate: 10
  ID 258 Age 0 LS 7 Pr dth


0. Advance 10 time periods
1. Advance to next time per
2. Advance to next time per
3. Inspect fox
4. Inspect warren
5. Exit

Select option: _
```

Q	Example Solution
6	

```
Class GiantWarren
        Inherits Warren
        Public Sub New(ByVal Variability As Integer, ByVal RabbitCount As Integer)
            MyBase.New(Variability)
            Rabbits = New Rabbit(200) {}
             RabbitCount = GiantRabbitCount
            For r = 0 To RabbitCount - 1
                Rabbits(r) = New Rabbit(Variability)
            Next
        End Sub
```

```
Private Sub CreateLandscapeAndAnimals(ByVal InitialWarrenCount As Integer, ByVal InitialFoxCou
FixedInitialLocations As Boolean)
        ...
            ...
            Landscape(10, 3).Warren = New Warren(Variability, 52)
            Landscape(11, 4).Warren = New GiantWarren(Variability, 115)
            Landscape(13, 4).Warren = New Warren(Variability, 67)
            ...
```



```
Select option: 1

TIME PERIOD: 6

      0 | 1 | 2 | 3 | 4
-------------------------
  0|    |    |    |    |
  1|    |38  |    |    |
  2|    |    |    |    |
  3|    |    |    |    |
  4|    |    |    |    |
  5|    |    |    |    |
  6|    |    |    |    |
  7|    |    |    |    |
  8|    |    |80  |    |
  9|    |    |    |    |
 10|    |    |   F|    |
 11|    |    |    |    |
 12|    |    |    |    |
 13|    |    |    |    |
 14|    |    |    |    |

0. Advance 10 time peri
1. Advance to next time
2. Advance to next time
3. Inspect fox
4. Inspect warren
5. Exit

Select option: _
```

Plus: *Warren instance variables need to be protected and not private:*

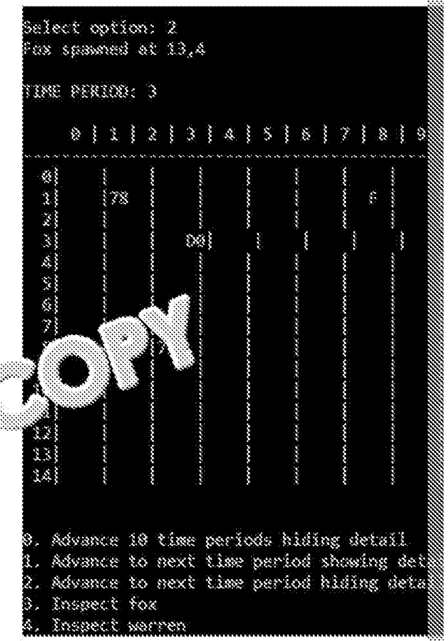
```
    Protected Const MaxRabbitsInWarren As Integer = 99
    Protected Rabbits() As Rabbit
    Protected RabbitCount As Integer = 0
    Protected PeriodsRun As Integer = 0
    Protected AlreadySpread As Boolean = False
    Protected Variability As Integer
    Protected Shared Rnd As New Random()
```

Q **Example Solution**

7

```
Class Den
    Private FoxesSpawned As Integer

    Public Sub New()
        FoxesSpawned = 0
    End Sub

    Public Function Spaw[illegible]
        Return N[illegible]
    End Function

    Public Function GetSymbol() As String
        Return "D" + FoxesSpawned.ToString
    End Function
End Class
```

```
Class Location
    Public Fox As Fox
    Public Warren As Warren
    Public Den As Den

    Public Sub New()
        Fox = None
        Warren = None
        Den = None
    End Sub
End Class
```



```
Select option: 2
Fox spawned at 13,4

TIME PERIOD: 3

     0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
-------------------------------------------------
  0|    |    |    |    |    |    |    |    |
  1|    |78  |    |    |    |    |    F |    |
  2|    |    |    |    |    |    |    |    |
  3|    |    |  D0|    |    |    |    |    |
  4|    |    |    |    |    |    |    |    |
  5|    |    |    |    |    |    |    |    |
  6|    |    |    |    |    |    |    |    |
  7|    |    |    |    |    |    |    |    |
 12|    |    |    |    |    |    |    |    |
 13|    |    |    |    |    |    |    |    |
 14|    |    |    |    |    |    |    |    |


0. Advance 10 time periods hiding detail
1. Advance to next time period showing det
2. Advance to next time period hiding deta
3. Inspect fox
4. Inspect warren
```

Q	Example Solution
7 (cont.)	

```
Private Sub DrawLandscape()
  ...
      ...
      If Not Landscape(x, y).Fox Is None Then
        Console.Write("F")
      Else
        Console.Write(" ")
      End If
      If Not Landscape(x, y).Den Is None Then
        Console.Write(Landscape(x, y).Den.GetSymbol())
      Else
        Console.Write(" ")
      End If
      Console.Write("|")
    Next
```

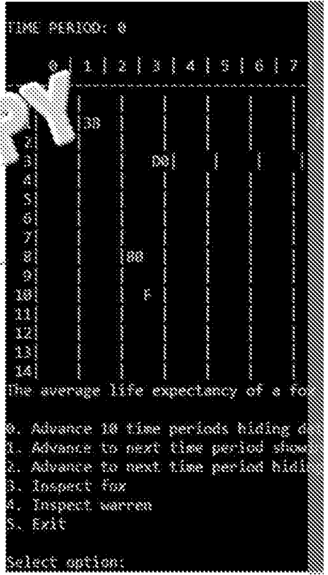
```
Private Sub CreateLandscapeAndAnimals(ByVal InitialWarrenCount As Integer, ByVal InitialFo
FixedInitialLocations As Boolean)
  ...
    ...
    FoxCount = 5
    Landscape(2, 3).Den = New Den()
```

```
Private Sub AdvanceTimePeriod()
  Dim NewFoxCount As Integer = 0
  If TimePeriod Mod 3 = 0 Then
    Dim x, y As Integer
    x = Rnd.Next(1, LandscapeSize - 1)
    y = Rnd.Next(1, LandscapeSize - 1)
    Landscape(x, y).Fox = Landscape(2, 3).Den.Spawn()
    Console.WriteLine("Fox spawned at " + x.ToString + "," + y.ToString)
  End If
```

Q	Example Solution
8	

```
Class Fox
    ...
    ...
    Private Gender As Genders
    Private Shared TotalDeadFoxes As Double = 10
    Private Shared TotalAge As Double = 70
```

```
    Public Sub AdvanceGeneration(ByVal ShowDetail As Boolean)
        ...
        If FoodUnitsConsumedThisPeriod = 0 Then
            ...
            ...
        End If
        If Not IsAlive Then
            TotalDeadFoxes +=
            TotalAge = TotalAge + Age
        End If
    End Sub

    Public Function GetLifeExpect() As Double
        Return TotalAge / TotalDeadFoxes
    End Function
```



```
TIME PERIOD: 0

     0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
   --------------------------------
     |     |38 |     |     |     |     |
    2|     |     |     |     |     |     |
    3|     |     |   D0|     |     |     |
    4|     |     |     |     |     |     |
    5|     |     |     |     |     |     |
    6|     |     |     |     |     |     |
    7|     |     |     |     |     |     |
    8|     |     |80   |     |     |     |
    9|     |     |     |     |     |     |
   10|     |     |  F  |     |     |     |
   11|     |     |     |     |     |     |
   12|     |     |     |     |     |     |
   13|     |     |     |     |     |     |
   14|     |     |     |     |     |     |
The average life expectancy of a fo

0. Advance 10 time periods hiding d
1. Advance to next time period show
2. Advance to next time period hidi
3. Inspect fox
4. Inspect warren
5. Exit

Select option:
```

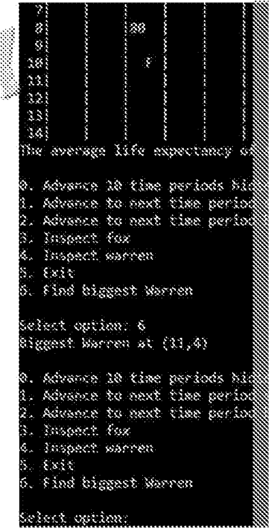
Add to end of DrawLandscape in Simulation:

```
Dim lifeExpect As Double
Dim theFox = New Fox([illegible])
lifeExpect = theFox.GetLifeExpect
Console.WriteLine("The average life expectancy of a fox stands at " + lifeExpect.ToString)
```

Q	Example Solution

9

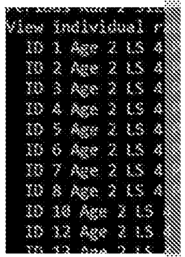
```
Public Sub New(ByVal LandscapeSize As Integer, ByVal InitialWarrenCount As Integer, ByVal In
Variability As Integer, ByVal FixedInitialLocations As Boolean)
  ...
    Console.WriteLine("5. Exit")
    Console.WriteLine("6. Find biggest Warren")
    ...
    ...
    If MenuOption = 6 Then
      findBiggest()
    End If

Private Sub findBiggest()
  Dim biggestX, biggestY, biggestSize As Integer
  biggestSize = -1
  For x = 0 To LandscapeSize - 1
    For y = 0 To LandscapeSize - 1
      If Not Landscape(x, y).Warren Is None Then
        If Landscape(x, y).Warren.GetRabbitCount > biggestSize Then
          biggestX = x
          biggestY = y
          biggestSize = Landscape(x, y).Warren.GetRabbitCount
        End If
      End If
    Next
  Next
  Console.WriteLine("Biggest Warren at (" + biggestX.ToString + "," + biggestY.ToString + ")")
End Sub
```



```
  7|   |   |   |   |   |
  8|   |   |20 |   |   |
  9|   |   |   |   |   |
 10|   |   |  F|   |   |
 11|   |   |   |   |   |
 12|   |   |   |   |   |
 13|   |   |   |   |   |
 14|   |   |   |   |   |
The average life expectancy o

0. Advance 10 time periods hid
1. Advance to next time period
2. Advance to next time period
3. Inspect fox
4. Inspect warren
5. Exit
6. Find biggest Warren

Select option: 6
Biggest Warren at (11,4)

0. Advance 10 time periods hid
1. Advance to next time period
2. Advance to next time period
3. Inspect fox
4. Inspect warren
5. Exit
6. Find biggest Warren

Select option:
```

10

```
Public Overrides Sub CalculateNewAge()
  MyBase.CalculateNewAge()
  ProbabilityOfDeathOtherCauses = ProbabilityOfDeathOtherCauses * 1.1
```



```
View individual r
   ID 1 Age 2 LS 4
   ID 2 Age 2 LS 4
   ID 3 Age 2 LS 4
   ID 4 Age 2 LS 4
   ID 5 Age 2 LS 4
   ID 6 Age 2 LS 4
   ID 7 Age 2 LS 4
   ID 8 Age 2 LS 4
   ID 10 Age 2 LS
   ID 12 Age 2 LS
```

Q | **Example Solution**

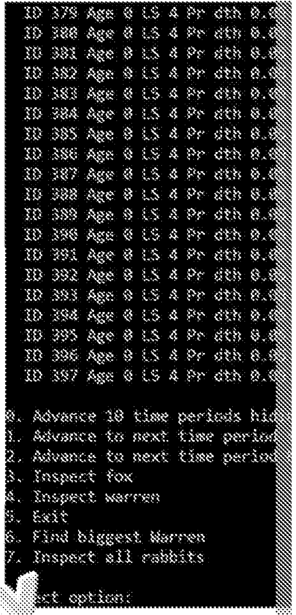
11

Add to class Warren:

```
Public Function getRabbits() As Rabbit()
    Return Rabbits
End Function
```

Add to class Simulation:

```
    Console.WriteLine("7. Inspect all rabbits")
    Console.WriteLine()
    Console.Write("Select option: ")
    MenuOption = CInt(Console.ReadLine())
    If MenuOption = 7 Then
      Dim AllRabbits() As Rabbit
      'get all rabbits
      For x = 0 To LandscapeSize - 1
        For y = 0 To LandscapeSize - 1
          If Not Landscape(x, y).Warren Is None Then
            AllRabbits = Landscape(x, y).Warren.getRabbits()
            'display all rabbits
            For i = 0 To AllRabbits.Length - 1
              Try
                AllRabbits(i).Inspect()
              Catch ex As Exception
                'catch null rabbits
              End Try
            Next
          End If
        Next
      Next
    End If
```



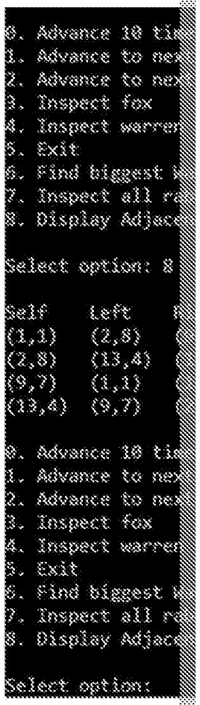
```
 ID 379 Age 0 LS 4 Pr dth 0.0
 ID 380 Age 0 LS 4 Pr dth 0.0
 ID 381 Age 0 LS 4 Pr dth 0.0
 ID 382 Age 0 LS 4 Pr dth 0.0
 ID 383 Age 0 LS 4 Pr dth 0.0
 ID 384 Age 0 LS 4 Pr dth 0.0
 ID 385 Age 0 LS 4 Pr dth 0.0
 ID 386 Age 0 LS 4 Pr dth 0.0
 ID 387 Age 0 LS 4 Pr dth 0.0
 ID 388 Age 0 LS 4 Pr dth 0.0
 ID 389 Age 0 LS 4 Pr dth 0.0
 ID 390 Age 0 LS 4 Pr dth 0.0
 ID 391 Age 0 LS 4 Pr dth 0.0
 ID 392 Age 0 LS 4 Pr dth 0.0
 ID 393 Age 0 LS 4 Pr dth 0.0
 ID 394 Age 0 LS 4 Pr dth 0.0
 ID 395 Age 0 LS 4 Pr dth 0.0
 ID 396 Age 0 LS 4 Pr dth 0.0
 ID 397 Age 0 LS 4 Pr dth 0.0

0. Advance 10 time periods hid
1. Advance to next time period
2. Advance to next time period
3. Inspect fox
4. Inspect warren
5. Exit
6. Find biggest Warren
7. Inspect all rabbits

ect option:
```

Q	Example Solution
12	

```
Class WarrenGraph
    Private Nodes As Node()

    Public Sub New()
        Dim n1 As New Node(1, 1, 2, 8, 9, 7)
        Dim n2 As New Node(2, 8, 13, 4, 1, 1)
        Dim n3 As New Node(9, 7, 1, 1, 13 [illegible]
        Dim n4 As New Node(13 [illegible] 8
        Nodes = New No [illegible] n4}
    End [illegible]

    [illegible] AdjList()
        [illegible]ole.WriteLine()
        Console.WriteLine("Self" + vbTab + "Left" + vbTab + "Right")
        For index = 0 To Nodes.Length - 1
            Console.WriteLine(Nodes(index).getCoord("s") + vbTab + Nodes(index).getCoord("l") + v
        Next
    End Sub
End Class

Class Node
    Private selfX As Integer
    Private selfY As Integer
    Private leftX As Integer
    Private leftY As Integer
    Private rightX As Integer
    Private rightY As Integer

    Public Sub New(sx As Int [illegible] y As [illegible] teger, lx As Integer, ly As Integer, rx As Integer, ry As Int
        selfX = sx
        [illegible] = sy
        [illegible] lx
        [illegible] ly
        rightX = rx
        rightY = ry
    End Sub
```

Q	Example Solution
12 (cont.)	

```
Public Function getCoord(ByVal branch As String) As String
  If branch.Equals("l") Then
    Return ("(" + leftX.ToString + "," + leftY.ToString + ")")
  ElseIf branch.Equals("r") Then
    Return ("(" + rightX.ToString + "," + ri[illegible]ing[illegible])")
  Else
    Return ("(" + s[illegible] selfY.ToString + ")")
  End If
[illegible]tio[illegible]
En[illegible]
```



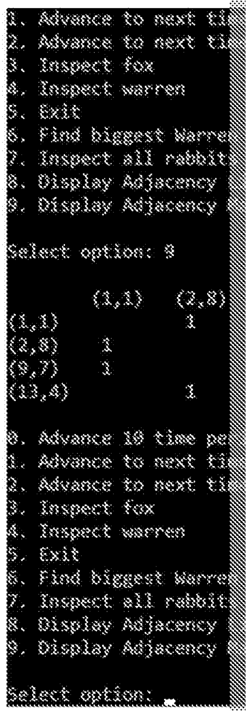
```
0. Advance 10 ti
1. Advance to ne
2. Advance to ne
3. Inspect fox
4. Inspect warre
5. Exit
6. Find biggest
7. Inspect all r
8. Display Adjac

Select option: 8

Self    Left    
(1,1)   (2,8)   
(2,8)   (13,4)  
(9,7)   (1,1)   
(13,4)  (9,7)   

0. Advance 10 ti
1. Advance to ne
2. Advance to ne
3. Inspect fox
4. Inspect warre
5. Exit
6. Find biggest
7. Inspect all r
8. Display Adjac

Select option:
```

Changes to class Simulation:

```
    ...
    Console.WriteLine("8. Display Adjacency List")
    Console.WriteLine()
    Console.Write("Select option: ")
    MenuOption = CInt(Console.ReadLine())
    If MenuOption = 8 Then
      Dim theGraph As New WarrenGraph()
      theGraph.AdjList()
    End If
```

Q	Example Solution
13	***Changes to the menu:***

```
...
Console.WriteLine("9. Display Adjacency Matrix")
Console.WriteLine()
Console.Write("Select option: ")
MenuOption = CInt(Console.ReadLine[illegible]
If MenuOption = 9 Then
   Dim theGraph As [illegible]raph()
   [illegible]Gr[illegible]()
```



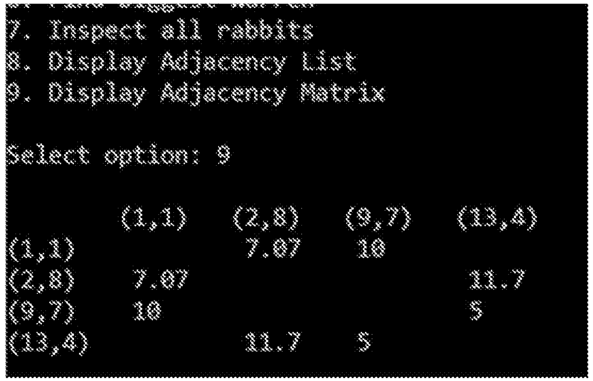
```
1. Advance to next ti
2. Advance to next ti
3. Inspect fox
4. Inspect warren
5. Exit
6. Find biggest Warre
7. Inspect all rabbit
8. Display Adjacency
9. Display Adjacency

Select option: 9

        (1,1)   (2,8)
(1,1)           1
(2,8)   1
(9,7)   1
(13,4)          1

0. Advance 10 time pe
1. Advance to next ti
2. Advance to next ti
3. Inspect fox
4. Inspect warren
5. Exit
6. Find biggest Warre
7. Inspect all rabbit
8. Display Adjacency
9. Display Adjacency

Select option: _
```

adjMatrix procedure in WarrenGraph:

```
Public Sub AdjMatrix()
   Dim theHeadings(Nodes.Length) As String
   Console.WriteLine()
   Console.Write(vbTab)
   For index = 0 To Nodes.Length - 1
      Console.Write(Nodes(index).getCoord("s") + vbTab)
      theHeadings(index) = Nodes(index).getCoord("s")
   Next
   Console.WriteLine()
   For index1 = 0 To Nodes.Length - 1
      Console.Write(Nodes(index1).getCoord("s") [illegible] vb[illegible]ab)
      For index2 = 0 To Nodes.Length - 1
         If (Nodes(index2).getCo[illegible]adings(index1)) Or (Nodes(index2).getCoord("r")
            Console.Wri[illegible]
         Else
            Co[illegible]rite(vbTab)
         [illegible]d If
      [illegible]ext
      Console.WriteLine()
   Next
End Sub
```

Example Solution

```
Public Sub AdjMatrix()
    Dim theHeadings(Nodes.Length) As String
    Console.WriteLine()
    Console.Write(vbTab)
    For index = 0 To Nodes.Length - 1
        Console.Write(Nodes(index).getCoord("s") + vbTab)
        theHeadings(index) = Nodes(index).getCoord("s")
    Next
    Console.WriteLine()
    For index1 = 0 To Nodes.Length - 1
        Console.Write(Nodes(index1).getCoord("s") + vbTab)
        For index2 = 0 To Nodes.Length - 1
            If (Nodes(index2).getCoord("l") = theHeadings(index1)) Or (Nodes(index2).getCoord("r") = theHeadings(index1)) Then
                Dim distance As Double
                Dim x1, x2, y1, y2 As Double
                x1 = theHeadings(index1).IndexOf(",")
                y1 = Double.Parse(theHeadings(index1).Substring(x1 + 1, ((theHeadings(index1).Length - (x1 + 2)))))
                x1 = Double.Parse(theHeadings(index1).Substring(1, x1 - 1))
                Dim coord2 As String = Nodes(index2).getCoord("s")
                x2 = coord2.IndexOf(",")
                y2 = Double.Parse(coord2.Substring(x2 + 1, ((coord2.Length - x2 - 2))))
                x2 = Double.Parse(coord2.Substring(1, x2 - 1))
                distance = (Math.Sqrt(Math.Pow(Math.Abs(x2 - x1), 2) + Math.Pow(Math.Abs(y2 - y1), 2)))
                distance = Math.Round(distance, 2)
                Console.Write(" " + distance.ToString + vbTab)
            Else
                Console.Write(vbTab)
            End If
        Next
        Console.WriteLine()
    Next
End Sub
```



```
7. Inspect all rabbits
8. Display Adjacency List
9. Display Adjacency Matrix

Select option: 9

        (1,1)   (2,8)   (9,7)   (13,4)
(1,1)            7.07    10
(2,8)    7.07                    11.7
(9,7)    10                      5
(13,4)           11.7    5
```

Suggested Marks

9 marks

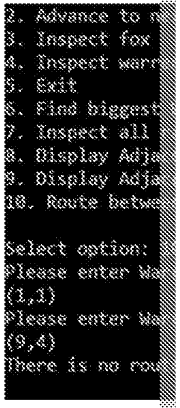
- *1 mark for getting the x,y coordinates of the starting point*
- *4 marks for IF statement to distinguish between whether node is left or right branch and getting the cords (must be inside IF statement already there)*
- *2 marks for applying Pythagoras correctly (there are several ways to do this, doesn't need to match example; award 1 mark for a good attempt)*
- *1 mark for rounding to 1dp*
- *1 mark for screen capture*

Q	Example Solution
15	*Added to class Simulation:*

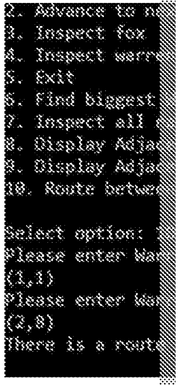
```
        Console.WriteLine("10. Route between warrens?")
        Console.WriteLine()
        Console.Write("Select option: ")
        MenuOption = CInt(Console.ReadLine())
        If MenuOption = 10 Then
          Dim theGraph As New WarrenGraph()
          theGraph.isRoute()
        End If
```

Added to class WarrenGraph:

```
  Public Sub isRoute()
      Dim route As Boolean = False
      Dim coord1, coord2 As String
      Console.WriteLine("Please enter Warren 1 coordinates in format (x,y)")
      coord1 = Console.ReadLine
      Console.WriteLine("Please enter Warren 2 coordinates in format (x,y)")
      coord2 = Console.ReadLine
      For index = 0 To Nodes.Length - 1
        If Nodes(index).getCoord("s") = coord1 Then
          If Nodes(index).getCoord("l") = coord2 Then
            route = True
          ElseIf Nodes(index).getCoord("r") = coord2 Then
            route = True
          End If
        End If
      Next

      If route = True Then
        Console.WriteLine("There is a route between the 2 warrens")
      [illegible]
        Console.WriteLine("There is no route between the 2 warrens")
      End If

    End Sub
```



```
2. Advance to n
3. Inspect fox
4. Inspect warr
5. Exit
6. Find biggest
7. Inspect all
8. Display Adja
9. Display Adja
10. Route betwe

Select option:
Please enter Wa
(1,1)
Please enter Wa
(9,4)
There is no rou
```



```
2. Advance to ne
3. Inspect fox
4. Inspect warre
5. Exit
6. Find biggest
7. Inspect all
8. Display Adjac
9. Display Adjac
10. Route betwee

Select option: 1
Please enter War
(1,1)
Please enter War
(2,8)
There is a route
```

Ideas for modifications	How to i

Name	

ZigZag Education supporting

A Level AQA Computer Science Pap

Summer 2017: Rabbits and Foxe

Electronic Answer Document (EAD)

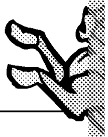
Instructions

- Enter your name in the box at the top of this page
- Answer **all** questions by entering your answers into this document
- Remember to **save** this document regularly
- Save and print this document and any additional pages

- Answer **all** questions
- The marks available for each question are shown in brackets

- You will need:
  - □ access to a computer
  - □ access to a printer
  - □ access to appropriate software
  - □ electronic copies of the required skeleton code
  - □ EAD (Electronic Answer Document)

Total marks:	



Programming Theory Qu

Answer all questions.
Remember to save this document re

Q		Answer
1		
2	(a)	
	(b)	
	(c)	
	(d)	
	(e)	
	(f)	
	(g)	
	(h)	
	(i)	
3	(a)	
	(b)	
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		

Programming Exerci

Answer all questions.
Remember to save this document re

Q	Answer
1	
2	
3	
5	
6	
7	
8	
9	
10	
11	
12	
13	
15	