Y13 Computer Science (OCR - H446)

Unit 2 - Algorithms & Programming

Sample Past Paper Questions

Topic 1

Elements of Computational Thinking

1 Eve enjoys playing board games. Her favourite board game is called “Pot Luck”. This has a numbered grid of 10 squares by 10 squares. Each square has a number between 1 and 100.

Players place their game counters on square 1. A 30-minute timer is set which counts downwards. Each player rolls two 6-sided dice and then moves their game counter that number of squares. Some squares tell the player to pick up a card. These have instructions on, such as 'Move forward 10 spaces'. If the player lands on one of these squares they move according to the instruction on the card. The first player to land on square 100, is announced as the winner. If no winner is announced before the timer runs out, then it is a draw.

Eve would like to create a computerised version of this game.

i. She has been told that she should make use of abstraction when creating the game.

Describe what is meant by the term 'abstraction'.

____

____

____

____ **[2]**

ii. Give **three** examples of how Eve could use abstraction when creating her game.

1 ____

____

2 ____

____

3 ____

____ **[3]**

iii. Give **two** reasons why Eve should use abstraction when designing the game.

1 ____

2

[2]

2 A group of students are designing a racing car game. The game will allow players to enter their name and then a choice of vehicle. They will then race against other vehicles that will be controlled by the program. Players will use the arrow keys to control their vehicle.

The students use abstraction during the design process.

i. State what is meant by abstraction **and** describe how it can be used to design the racing car game.

Definition

Use

[3]

ii. Explain why it is beneficial to use abstraction when designing a computer program such as a game.

[3]

3 A text-based computer game allows a user to dig for treasure on an island. The island is designed as a grid with 10 rows and 20 columns to store the treasure. Each square is given an x and y coordinate. Some of the squares in the grid store the name of a treasure object. Each treasure object has a value, e.g. 100 and a level, e.g. "Bronze."

The computer game makes use of abstraction.

i. Describe what is meant by the term abstraction and give an example of how abstraction can be used in the treasure game.

Description: ____

____

____

____

Example: ____

____ **[3]**

ii. Give **three** benefits of using abstraction when writing a program.

1 ____

____

2 ____

____

3 ____

____ **[3]**

4 Taylor is designing a program for a client who would like to simulate earthquakes on major cities around the world in 3D. The client would like to be able to view any stage of an earthquake such as:

1. the build-up of the earthquake
2. the earthquake taking place
3. the aftershocks of the earthquake.

The client would also like to be able to play the simulation at different speeds. For example, a slow, normal or fast speed.

Give **three** examples of where abstraction can be used in the design of this program.

1 ____

____

2 ____

____

3 ____

____ **[3]**

5 A programmer is developing an aeroplane simulator. The user will sit in a cockpit and the simulated environment will be displayed on screens around them.

The programmer uses computational methods to design a solution for the program.

i. Complete the table by writing a definition for each computational method.

Computational Method	Definition
Abstraction	
Decomposition	

[2]

ii. Give **three** potential differences between the abstracted aeroplane simulator and reality.

1 ____

____

2 ____

____

3 ____

____

[3]

iii. Identify **two** reasons why abstraction is used when designing a solution to the problem.

1 ____

____

2 ____

____

[2]

6 Kira is creating a computer game where the user can play against the computer.

In each turn, each character can make one move from a selection of possible moves.

Kira uses a tree data structure shown in **Fig. 1** to identify the range of possible moves the computer can make from starting position A. Each connection is a move, with each node representing the result of the move.

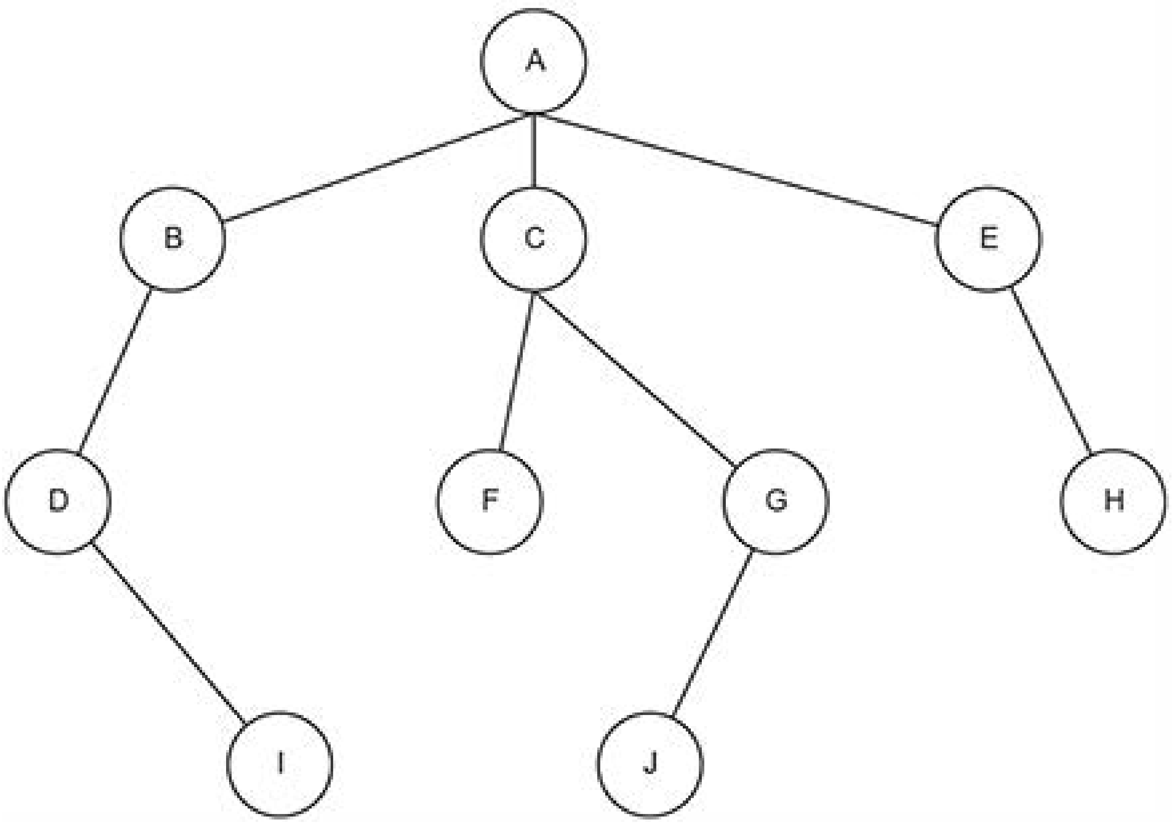


Fig. 1

State what is meant by the term 'abstraction' and describe how Kira has used abstraction in her design of the tree.

[3]

7 A car racing team uses a car simulator to test their drivers in a range of cars on different race tracks.

The car simulator uses an abstraction of the real car and race track. Identify **two** ways in which the simulator could use abstraction.

1

2

--- **[2]**

8(a) DriveSim Tutor is a 3D driving simulator program designed to allow learner drivers to practice following the Highway Code whilst driving through a virtual town.

The simulator's developers study a real town. They then use abstraction on their findings before designing a virtual town.

Explain why it is necessary for the developers to use abstraction.

--- **[2]**

(b) As a result of abstraction there will be similarities and differences between the virtual and real town.

i. State **two** similarities there might be between the virtual and real town. Explain why these similarities exist.

1

2

-- **[4]**

ii. State **two** differences between the virtual and real town.

1

2

-- **[2]**

9 A program is being designed that will allow a user to log into an account on a website using a username and password.

Identify **two** possible inputs and **one** output this program will need.

Input 1 ---

Input 2 ---

Output ---

--- **[3]**

10 Given the following procedure:

```
procedure maths(number)
    a = (number DIV 10) * 10
    b = a + 10
    if (number – a) >= (b – number) then
       print(b)
    else
       print(a)
    endif
endprocedure
```

State the value printed by the procedure `maths` if `number=10` ------------------- **[1]**

11 The following pseudocode procedure performs an insertion sort on the array parameter.

```
procedure insertionSort(dataArray:byRef)
  for i = 1 to dataArray.Length - 1
    temp = dataArray[i]
    tempPos = i – 1
    exit = false
    while tempPos >= 0 and exit == false
       if dataArray[tempPos] < temp then
          dataArray[tempPos + 1] = dataArray[tempPos]
          tempPos = tempPos - 1
       else
          exit = true
       endif
    endwhile
    dataArray[tempPos + 1] = temp
  next i
endprocedure
```

State whether the procedure `insertionSort` sorts the data into ascending or descending order and explain your choice.

[3]

12 Lucas writes a program that makes use of a circular queue. The queue stores the data entered into the program. An array is used to represent the queue.

The program needs two pointers to access and manipulate the data in the queue.

State the purpose of the two pointers and give an appropriate identifier for each.

Pointer 1 purpose ____________________

____

Pointer 1 identifier ____________________

____

Pointer 2 purpose ____________________

____

Pointer 2 identifier ____________________

____

[4]

13 Give **two** reasons why reusable program components are used in programs.

1 ____________________

____

2 ____________________

____________________ **[2]**

14 Eve enjoys playing board games. Her favourite board game is called “Pot Luck”. This has a numbered grid of 10 squares by 10 squares. Each square has a number between 1 and 100.

Players place their game counters on square 1. A 30-minute timer is set which counts downwards. Each player rolls two 6-sided dice and then moves their game counter that number of squares. Some squares tell the player to pick up a card. These have instructions on, such as 'Move forward 10 spaces'. If the player lands on one of these squares they move according to the instruction on the card. The first player to land on square 100, is announced as the winner. If no winner is announced before the timer runs out, then it is a draw.

Eve would like to break the problem down into smaller sub problems so that each sub problem will complete one specific task.

Identify **three** sub problems that Eve can use in her game.

1 --

----

2 --

----

3 --

-- **[3]**

15 Mabel is a software engineer. She is writing a computer game for a client. In the game the main character has to avoid their enemies. This becomes more difficult as the levels of the game increase.

The computer game allows a user to select a character (e.g. name, gender). They can then choose a level for the game (easy, normal, challenging). The user controls their character by moving it left or right. The character can jump using space bar as an input. If the character touches one of the enemies then it loses a life. The character has to make it to the end of the level without losing all their lives.

The game is designed in a modular way.

i. One sub-procedure will handle the user input.

Describe **three** other sub-procedures Mabel could create for the given game description.

1 --

2

3

[6]

ii. Describe the decision that the program will need to make within the user input subprocedure and the result of this decision.

[2]

iii. Define pipelining and give an example of how it could be applied in the program.

[2]

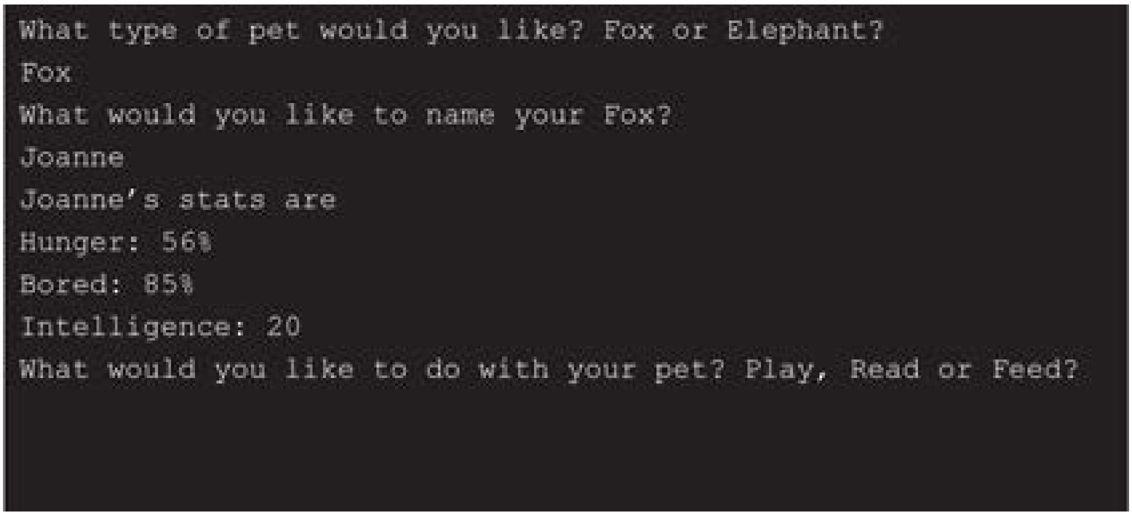
16 A software developer is creating a Virtual Pet game.

The user can choose the type of animal they would like as their pet, give it a name and then they are responsible for caring for that animal. The user will need to feed, play with, and educate their pet.

The aim is to keep the animal alive and happy, for example if the animal is not fed over a set period of time then the pet will die.

- The game tells the user how hungry or bored the animal is as a percentage (%) and the animal's intelligence is ranked as a number between 0 and 150 (inclusive).
- Hunger and boredom increase by 1% with every tick of a timer.
- When the feed option is selected, hunger is reduced to 0.
- When the play option is selected, bored is reduced to 0.
- When the read option is selected, the intelligence is increased by 0.6% of its current value.

An example of the game is shown:



```
What type of pet would you like? Fox or Elephant?
Fox
What would you like to name your Fox?
Joanne
Joanne's stats are
Hunger: 56%
Bored: 85%
Intelligence: 20
What would you like to do with your pet? Play, Read or Feed?
```

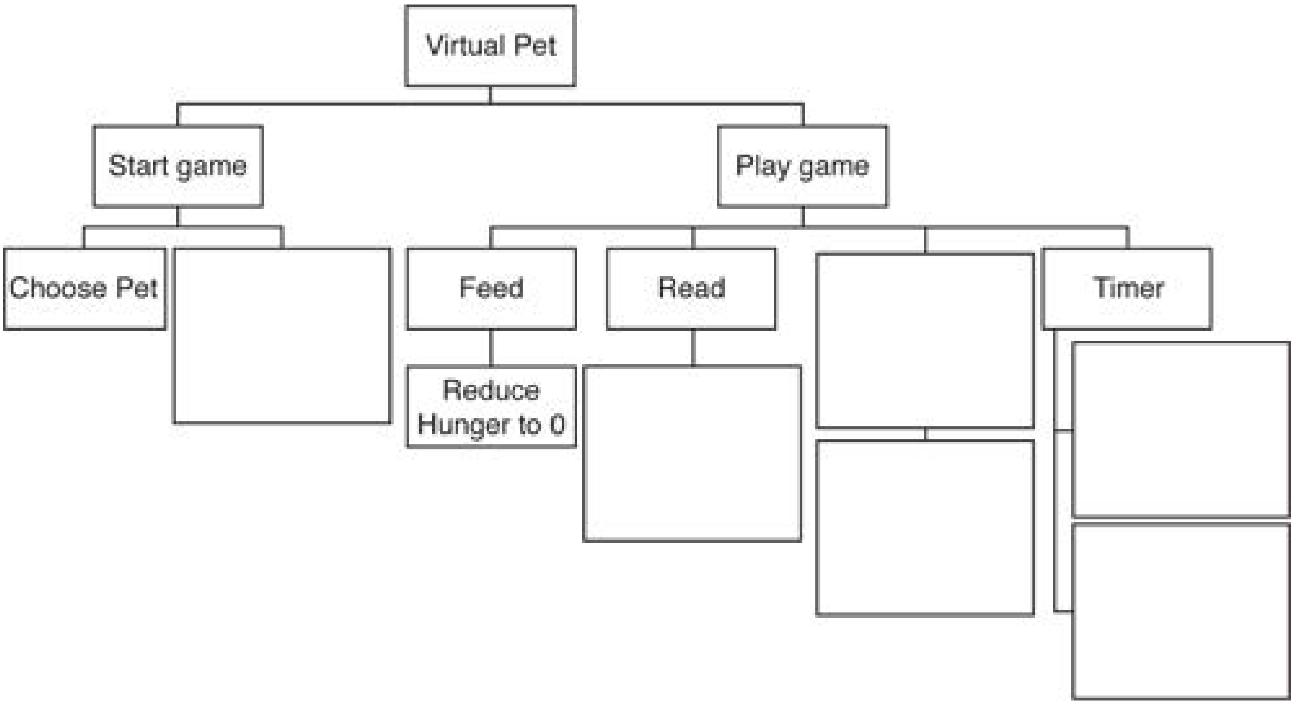
Fig. 1.1

The developer is using decomposition to design the game.

i. Describe the process of decomposition.

[2]

ii. The developer has produced the following structure diagram for the game:



Complete the structure diagram for the Virtual Pet game by filling in the empty boxes.

[6]

17 A card game uses a set of 52 standard playing cards. There are four suits; hearts, diamonds, clubs and spades. Each suit has a card with a number from; 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.

The card game randomly gives 2 players 7 cards each. The unallocated cards become known as the deck.

The players then take it in turns to turn over a card. A valid move is a card of the same suit or the same number as the last card played.

The winner is the first player to play all of their cards.

One component of the game is checking if a move is valid.

Identify **three** other components of the game.

1 ____

2 ____

3 ____

[3]

18 A programmer has designed a program that includes a reusable program component.

The reusable program component is a function called `isInteger()`. This will take a string as an argument and then check that each digit is between 0 and 9. For example if 103 is input, it will check that the digits 1, 0 and 3 are each between 0 and 9.

The `asc()` function returns the ASCII value of each digit. For example `asc("1")` returns 49.

The ASCII value for 0 is 48. The ASCII value for 9 is 57.

```
01  function isInteger(number)
02    result = true
03    for count = 0 to number.length-1
04      asciiValue = asc(number.substring(count, 1))
05        if not(asciiValue >= 48 and asciiValue <= 57) then

06          result = false
07      endif
08    next count
09    return result
10  endfunction
```

i. Identify **one** identifier used in the function `isInteger()`.

[1]

ii. Give the line number where the branching (selection) construct starts in the function `isInteger()`.

[1]

iii. Give the line number where the iteration construct starts in the function `isInteger()`.

[1]

19 A recursive pseudocode function, `recursiveAlgorithm()`, is shown.

```
01  function recursiveAlgorithm(value)
02    if value <= 0 then
03      return 1
04    elseif value MOD 2 = 0 then
05      return value + recursiveAlgorithm(value – 3)
06    else
07      return value + recursiveAlgorithm(value – 1)
```

```
        endif
    endfunction
```

Trace the recursive function, `recursiveAlgorithm()`, and give the final return value when called with `recursiveAlgorithm(10)`. You may choose to use the table below to give your answer.

Function call	`value`	`return`

Final return value ....

[5]

20 A program uses a bubble sort to sort data into ascending numerical order.

The data is stored in a 0-indexed 1-dimensional array.

Show each stage of a bubble sort to sort this data into ascending numerical order:

1	5	3	9	2	7

You should clearly show and label each pass in your answer.

[4]

21 Given the following procedure:

```
procedure generate(number)
        a = 0
        while number > 0
             if number MOD 2 == 0 then
                  a = a + 2
                  print(a)
                  number = number – 2
             else
                  a = a + 1
                  print(a)
                  number = number – 1
             endif
        endwhile
endprocedure
```

State the values printed by the procedure `generate` when `number` = 7.

[1]

22(a) A printer buffer is a storage area that holds the data, known as jobs, that are to be printed by a printer.

A simulation of the printer buffer uses a queue data structure to store jobs that are waiting to be printed. The queue is not circular.

The printer buffer is represented as a zero-indexed 1D array with the identifier `buffer`.

Fig. 2 shows the current contents of the queue `buffer` and its pointers.

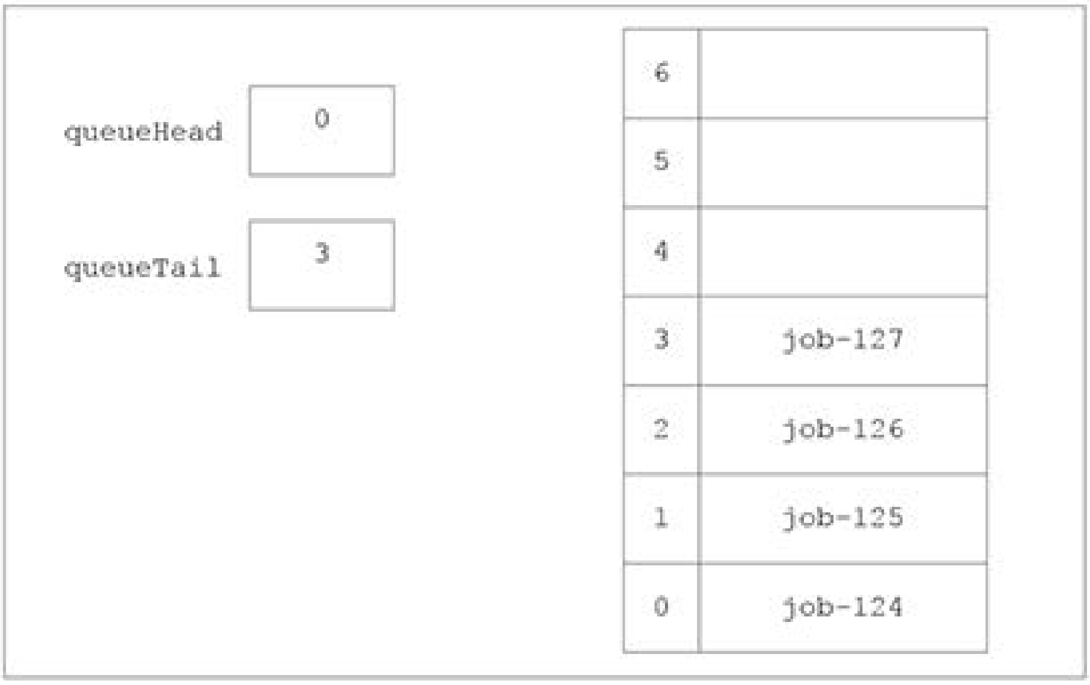


Fig. 2

State the purpose of the pointers `queueHead` and `queueTail`.

`queueHead` ____

____

`queueTail` ____

____

[2]

(b) The function `dequeue` outputs and removes the next data item in the queue.

The procedure `enqueue` adds the job passed as a parameter to the queue.

Show the final contents of the queue and pointer values after the following instructions have been run on the queue `buffer` shown in **Fig. 2**.

```
dequeue()

dequeue()

enqueue(job-128)

dequeue()

enqueue(job-129)
```

queueHead □

queueTail □

6	
5	
4	
3	
2	
1	
0	

[5]

(c) The array, `buffer` and pointer values are declared with global scope.

i. The function `dequeue` returns `null` if the array is empty, and the contents of the next element if not empty. The queue is not circular.

Write an algorithm, using pseudocode or program code, for the function `dequeue()`.

[5]

ii. The function `enqueue` returns -1 if there is no space at the end of the queue to add data, and returns 1 if the parameter was added to `buffer`. The array `buffer` contains a maximum of 100 elements.

Write an algorithm, using pseudocode or program code, for the function `enqueue()`.

[6]

iii. In the main program of the simulation the user is asked whether they want to add an item to the queue or remove an item.

If they choose to add an item they have to input the job name, and the function `enqueue` is called.

If they choose to remove an item, the function `dequeue` is called and the job name is output.

Appropriate messages are output if either action cannot be run because the queue is either empty or full.

Write, using pseudocode or program code, an algorithm for the main program of the simulation.

[8]

(d) The queue is changed to make it a circular queue.

Describe how the functions `enqueue` and `dequeue` will need to be changed to allow `buffer` to work as a circular queue.

[3]

(e) Some print jobs can have different priorities. The higher the priority the sooner the job needs to be printed.

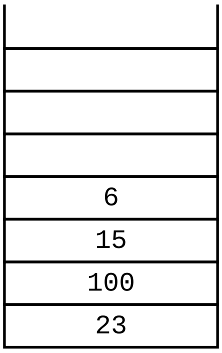
Describe how the program could be changed to deal with different priorities.

[3]

23 A computer program stores data input on a stack named `dataItems`. The stack has two subprograms to add and remove data items from the stack. The stack is implemented as a 1D array, `dataArray.`

Sub-program	Description
`push()`	The parameter is added to the top of the stack
`pop()`	The element at the top of the stack is removed

The current contents of `dataItems` are shown:



The main program asks a user to push or pop an item from the stack. If the user chooses ‘push’, the data item is added to the stack. If the user chooses “pop”, the next item is removed from the stack, multiplied by 3 and output.

The main program is shown:

```
01  userAnswer = input("Would you like to push or pop an item?")
02  if userAnswer == "push" then
03    push(input("Enter data item"))
04  else
05    print(pop() * 3)
06  endif
```

i. Before the sub-programs, `push()` and `pop()`, can add or remove items from the stack, a selection statement is used to decide if each action is possible.

Describe the decision that needs to be made in each sub-program and how this impacts the next process.

`push()` --

`pop()` --

----

[4]

ii. The algorithm does not work when the user enters "`PUSH`" or "`Push`". The algorithm needs to be changed in order to accept these inputs.

Identify the line number to be changed and state the change that should be made.

Line number --

Change --

----

[2]

24 State the **three** basic programming constructs used to control the flow of execution, giving your own example of each.

1

Example

2

Example

3

Example

[6]

25 A supermarket uses a stock control system.

Details of products are stored on a stock database.

Explain how the system used in the supermarket can control the quantity of tins of beans in stock so that the chance of running out is minimised.

[6]

26 * A company enforces standard rules about writing functions on its programmers. Discuss the reasons why this might be the case.

[9]

27 DriveSim Tutor is a 3D driving simulator program designed to allow learner drivers to practice following the Highway Code whilst driving through a virtual town.

The simulator's developers study a real town. They then use abstraction on their findings before designing a virtual town.

A road in the town has a "no overtaking" sign.

Describe how the simulator would check the driver obeys this sign whilst on this road.

-- **[3]**

28 A program designer is investigating the use of concurrent processing.

i. Describe what is meant by the term 'concurrent processing'.

-- **[2]**

ii. Give **two** benefits of using concurrent processing.

1 --

----

2 --

-- **[2]**

29 Taylor has written a game which uses concurrent processing.

i. Describe what is meant by the term ‘concurrent processing’.

--- **[2]**

ii. Explain why concurrent processing is needed to allow multiple users to log in and interact with game elements at the same time.

--- **[3]**

30 A programmer is developing an ordering system for a fast food restaurant. When a member of staff inputs an order, it is added to a linked list for completion by the chefs.

The programmer is considering using concurrent programming.

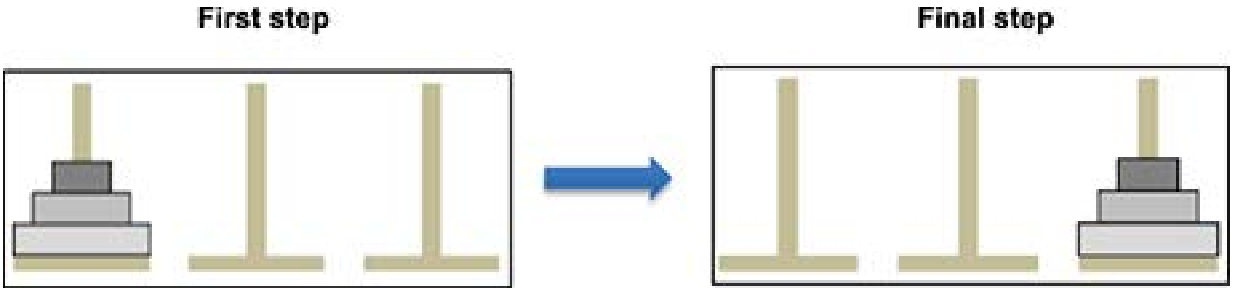
Discuss how concurrent programming can be applied to the food ordering system and the benefits and limitations of doing so. **[9]**

31 *A programmer is interested in using concurrent processing to perform a searching algorithm.

Explain how concurrent processing could be used in searching algorithms, and evaluate the benefits and trade-offs from implementing concurrent processing in a searching algorithm.

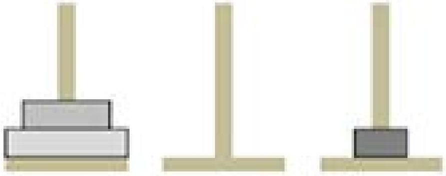
[9]

32(a) The Towers of Hanoi is a classic puzzle. Disks are placed in order on a pole, the biggest disc at the bottom of the pole, the smallest disk at the top of the pole, on the first of three poles. The challenge is to get them to the third pole in the same order.

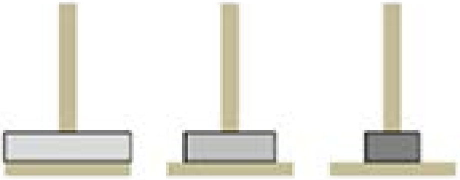


The disks can only be moved under the following rules:

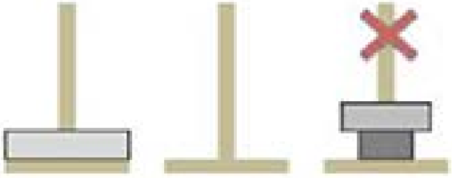
- only one disk can be moved at a time
- a disk can only ever be placed on an empty pole or on top of a larger disk
- a larger disk can never be placed on a smaller disk.



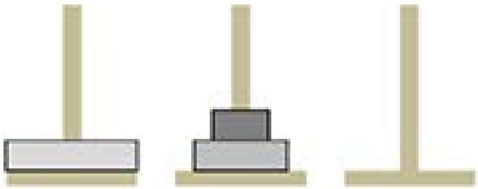
This is a valid move.



This would be a valid second move.



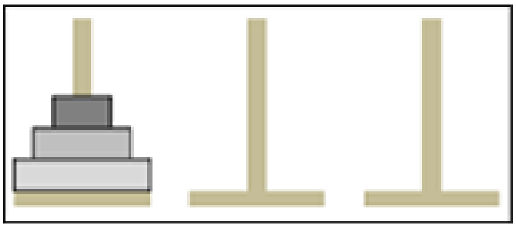
This would **not** be a valid third move (you can't put a bigger disk on a smaller one.)



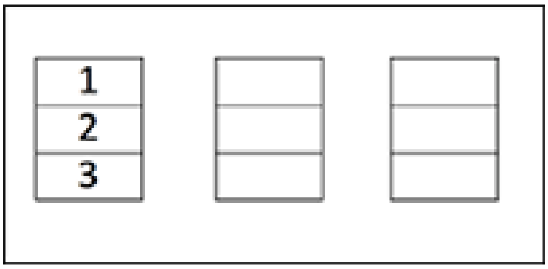
This **would** be a valid third move.

Each disk can be represented by an integer denoting its size.

So



Can be represented by

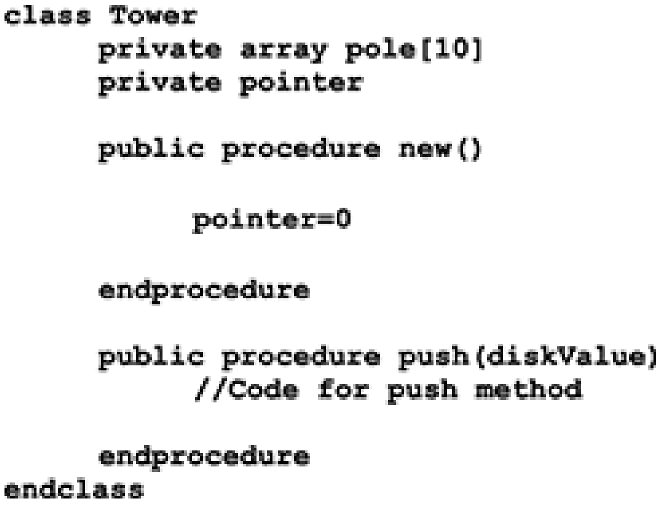


i. Explain why you would use a *stack* rather than a *queue* to store the configuration of disks at each pole.

-- [2]

ii. The tower class has the method push. It takes in the value of the disk to be pushed. It adds it to the top of the stack if it is a valid move. If it is not a valid move, the value of the disc is not added and the message 'Invalid move' is printed to the screen.

The stack is implemented using an array called pole and an integer called pointer. Pointer represents the index of the array position at the top of the stack.



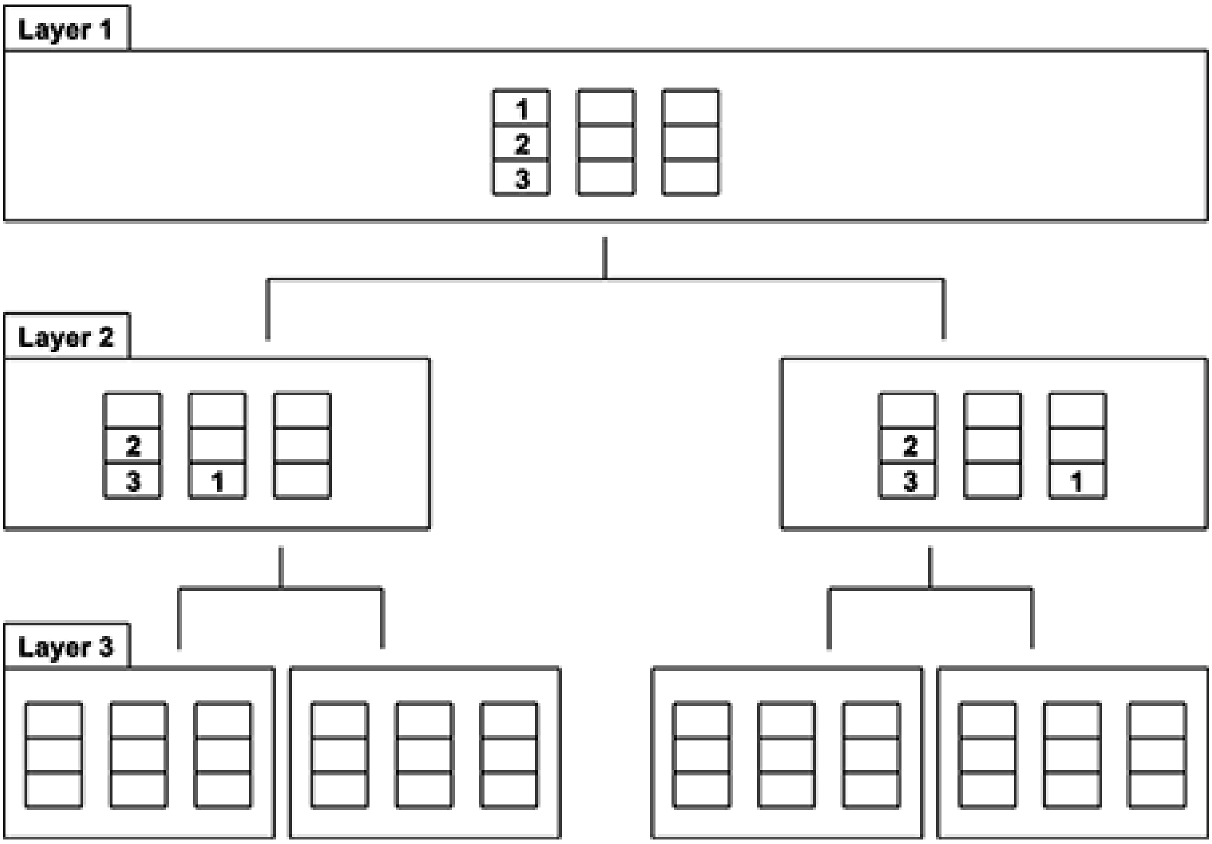
```
class Tower
     private array pole[10]
     private pointer

     public procedure new()

          pointer=0

     endprocedure

     public procedure push(diskValue)
          //Code for push method

     endprocedure
endclass
```

Write the pseudocode to go in the push method. Annotate your pseudocode with comments to show how it solves the problem. You are not expected to test for overflow.

[6]

(b) One way to try to find a solution would be to generate a tree of possible moves until a solution is found.

i. A tree has been started below. Complete **Layer 3** to show 4 possible moves.



[4]

ii. The search space represented by the tree could be searched using a depth first or breadth first search.

Describe **one** advantage and **one** disadvantage of depth-first search compared with breadth-first search.

Advantage:

Disadvantage:

--- [4]

(c) Rather than using a tree, the following iterative algorithm can be used to play the perfect game where the number of disks is odd. A similar algorithm exists for an even number of disks. A program is required to solve the Towers of Hanoi puzzle using the iterative algorithm below.

Cycle through the following three steps until the puzzle is solved (which may be after any of the steps):

- make the valid move between tower1 and tower3
- make the valid move between tower1 and tower2
- make the valid move between tower3 and tower2

NB: The valid move might be in either direction but there will only be one possibility each time.

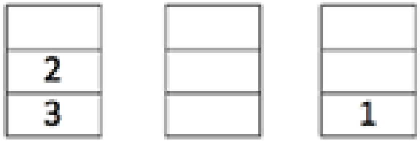
In this program there are three objects of the class Tower; `tower1, tower2` and `tower3`

The Tower class has the methods push, peek and pop

The method `push` adds a disk to the tower, for example `tower1.push(3)` adds a disk of size 3 to tower1.

The method peek returns the value of the disk on top of the tower but does not remove it. It returns the value 999 if the tower is empty.

The method `pop` removes a disk from a tower and returns the value of that disk.

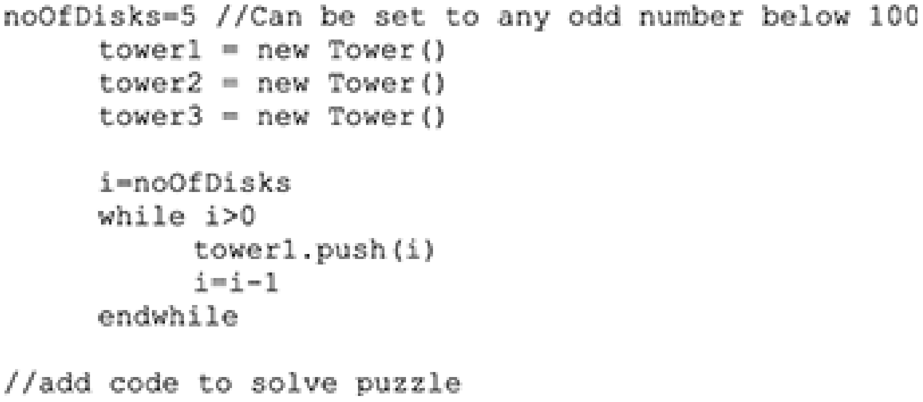


e.g. `x=tower1.peek()` would make x equal to 2 and the towers stay the same.

`x=tower1.pop()` would make x equal to 2 and remove 2 from tower1.

Iterative algorithm to solve Towers of Hanoi for an odd number of disks.

Complete the pseudocode program below so when given an odd number of disks, below 100, on tower1 they will be moved to tower3 using the iterative algorithm. Annotate your pseudocode with comments to show how it solves the problem.



```
noOfDisks=5 //Can be set to any odd number below 100
      tower1 = new Tower()
      tower2 = new Tower()
      tower3 = new Tower()

      i=noOfDisks
      while i>0
            tower1.push(i)
            i=i-1
      endwhile

//add code to solve puzzle
```

[10]

(d) The complexity of solving the Towers of Hanoi can be expressed in Big O notation as O(2^n) where n is the number of disks.

i. A given computer takes 8 milliseconds (ms) to solve a 3 disk problem. Calculate how long the computer takes to solve a 5 disk problem.

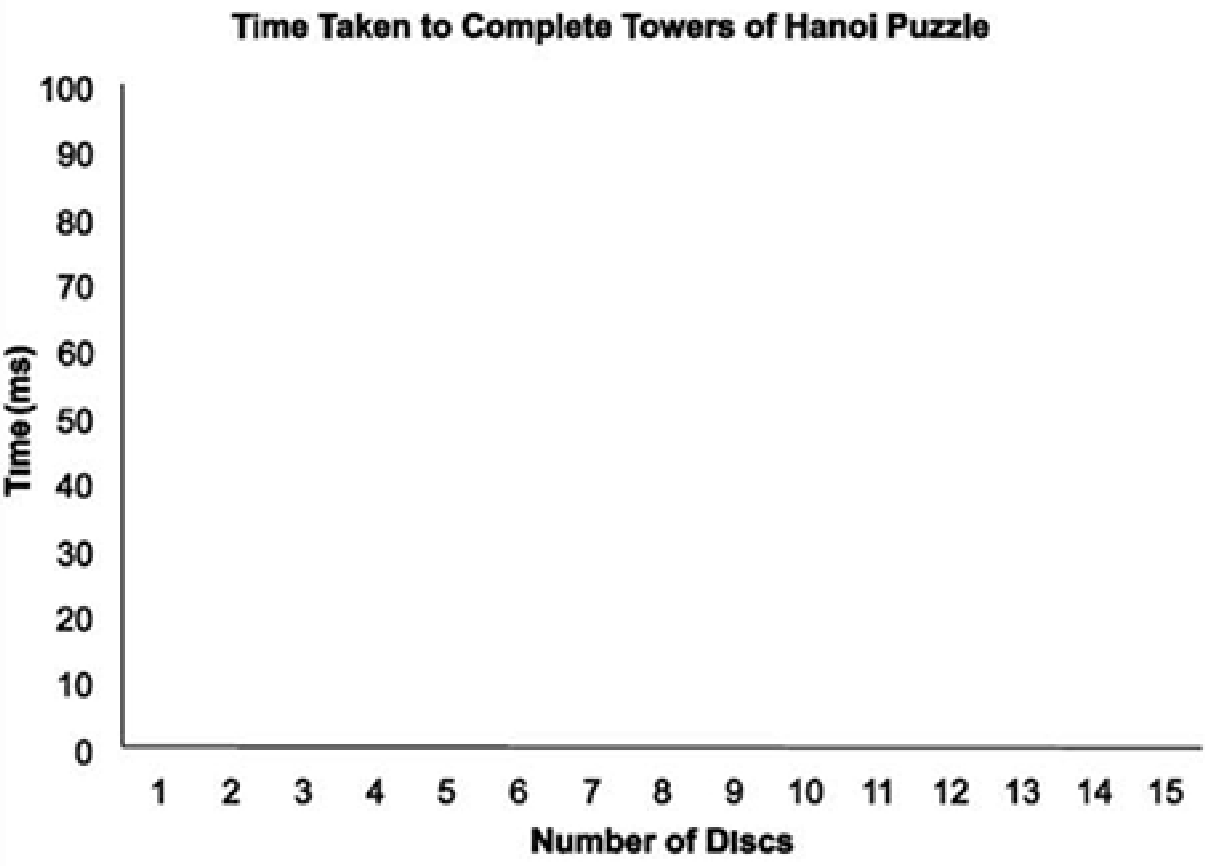
-- **[1]**

ii. State **one** reason why the answer given for part **(i)** may only be an estimate.

-- **[1]**

iii. Complete the graph below to show an estimate of how long a computer would take to solve the Towers of Hanoi with a variable number of discs.

[2]



(e) * Concurrency can be used to speed up the processing of some problems.

Discuss to what extent concurrency might be of use when solving the Towers of Hanoi puzzle. You should consider both the search space tree and iterative algorithm approaches in your answer.

[9]

33 A flight simulator allows a user to take control of a simulated aeroplane. The user can fly the plane in an environment that can simulate different weather conditions and additional planes in the sky.

Explain what is meant by 'concurrent processing' and describe **one** example of how the simulator could make use of it.

Concurrent processing

Example

[4]

34 Describe **two** examples of where pipelining is used in any computer system.

1

2

---

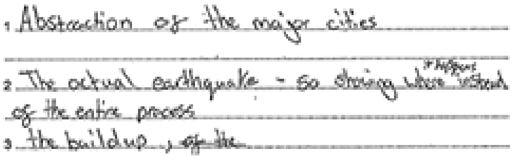
[4]

END OF QUESTION PAPER

Question			Answer/Indicative content	Marks	Guidance
1		i	1 mark per bullet to max 2 • Removing unnecessary detail • Simplifying a complex problem • Focussing on only the necessary/main parts	2	<u>Examiner's Comments</u> Many candidates could give a definition for abstraction with many specifying that it removed unnecessary detail to simplify a problem.
		ii	1 mark per bullet to max 3 e.g. • The physical game **board/squares** can be replaced with a grid • Physical **dice** can be replaced with two numbers • Physical player **counters** can be replaced with simple 2D graphics • Physical instruction **cards** can be replaced with text messages • Physical **timer** can be replaced with a simple text based timer	3	Allow other suitable examples that are relevant to the scenario. <u>Examiner's Comments</u> Some candidates struggled to use the context of the question to identify components such as the dice, counters, squares, cards or the timer. Candidates need to be able to apply knowledge of abstraction to give suitable responses within context.
		iii	1 mark per bullet to max 2 e.g. • Reduce programming time/cost • Simpler to solve the problem • Program requires less memory // computational power // faster program execution • Reduces complexity of programming code • Allows Eve to focus on core aspects	2	<u>Examiner's Comments</u> Many candidates successfully identified at least one reason why abstraction should be used, such as decreasing the computational processing demands for the solution. Some candidates did however repeated definitions of abstraction, rather than giving the advantages of using it. Another common error was to give an example of how it would benefit a player of the game rather than focusing on the design stage.
			Total	7	
2		i	1 mark for definition, 1 mark each for each example of use to max 2 (3 overall) **Definition:** • Removal of unnecessary detail **Example use:** • E.g. simplifying scenery • E.g. removing internal features of a vehicle that are not needed • E.g. simplify physics for vehicle movement • E.g. The vehicles may not be drawn to scale	3	Allow any reasonable examples for this scenario For the example use, allow 2 marks for stating a valid example of abstraction with an expansion. For example,"simplify track (1) by taking out the bumps in the road (1)" would be given two marks. <u>Examiner's Comments</u> Many candidates scored two marks for defining abstraction and giving one example of its application to the game, but fewer managed to identify a second example for full marks.

Question			Answer/Indicative content	Marks	Guidance
		ii	1 mark each to max 2 • Simplifies the problem / algorithm / programming code • Faster to create the program code • Final program uses less memory/processor time • Programmer can focus on core aspects of the game • Completed game will be simpler for end users to understand / play	3	Do not accept a reiteration of a definition of abstraction. Examiner's Comments Some candidates reiterated a definition of abstraction and did not focus on the benefits of abstraction as the question required. Most candidates struggled to give more than one or two benefits. There were also a number of unqualified responses such as 'saves time' that did not specify 'saving coding development time'.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
3		i	1 mark for each description to max 2 and 1 mark for example e.g. • Removal of unnecessary detail… • ….to allow programmers to focus on core aspects of the problem…. • ….simplifies a complex problem Examples, e.g: • Treasure objects are replaced with text labels // no images of treasure are used • Island is set of coordinates and no info as to environment/layout and other objects	3	Allow other suitable examples that are relevant to the treasure game. Examiner's Comments Many candidates identified that abstraction simplified or removed unnecessary detail to gain some marks, but then found it harder to give an example relevant to the scenario. Many candidates gave examples that related to graphics whereas the scenario explicitly stated that the game was text based.
		ii	1 mark each to max 3 e.g. • Reduces programming time • Reduces complexity of code (through abstraction by generalisation) • Reduces amount of memory required / computational power • Simplifies the problem so it's easier to solve / understand (by recognising common patterns) • Allows programmers to focus on core aspects of the problem	3	Examiner's Comments Many candidates gained some marks, but most found it difficult to identify three distinct reasons, and repetition was often observed in responses. Again, some candidate answers did not provide suitable levels of qualification for the points given.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
4			1 mark per bullet to max 3 Any reasonable abstraction e.g. • will not be to scale not life size • will exclude features e.g. people, road markings etc • will only show what is relevant e.g. buildings	3	<u>Examiner's Comments</u> Most candidates demonstrated that they knew what abstraction was, but a significant number struggled to make clear qualified points. Some candidates gave definitions of abstraction as the removal of unnecessary detail, but this did not answer the question. This was a contextual question that required relevant and qualif ied points. Responses such as 'the weather' were insufficient, whereas 'removal of the effects of weather' was a valid qualified point. Exemplar 2  1 Abstraction of the major cities 2 The actual earthquake - so showing where it happens instead of the entire process 3 the buildup , ~~of the~~ This exemplar shows a response that has unqualified points that do not show how abstraction can be applied. It highlights factors that could be abstracted, but does not tell us how.
			Total	3	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>5</td><td></td><td>i</td><td>Max 1 mark for each definition
e.g.<table><tr><th>Term</th><th>Definition</th></tr><tr><td>Abstraction</td><td>Removal of unnecessary components // focus on only necessary components</td></tr><tr><td>Decomposition</td><td>Breaking down a problem into subproblems</td></tr></table></td><td>2</td><td><u>Examiner's Comments</u>
While many candidates accurately recited definitions for abstraction and decomposition there were equally many that presented very vague and unqualified responses. E.g. For abstraction: 'Simplifying a problem' without specifying how it was simplified was insufficient. For decomposition: 'Breaking a problem into smaller parts' – without specifying that the 'smaller parts' are sub-problems rather than saying sequences of instructions or loops was insufficient.</td></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark for each
e.g.
• Removal of visual elements such as buildings on the ground
• Simplification of controls
• Focus on important elements such as weather, height, speed</td><td>3</td><td><u>Examiner's Comments</u>
Many candidates gave detailed and relevant examples that were suitable within the context of a flight simulator. However, there were many unqualified responses that were too vague. Unqualified responses such as 'terrain' did not go far enough to explain how the terrain in the simulation would differ from reality. To gain marks candidates had to make it clear exactly how the difference identified differed between reality and the simulation.</td></tr>
<tr><td></td><td></td><td>iii</td><td>1 mark for each to max 2
e.g.
• Reduce memory requirements
• Reduce processing requirements
• Simplify the problem being solved</td><td>2</td><td><u>Examiner's Comments</u>
It was noticeable that many candidates reiterated definitions of abstraction rather than identifying the actual reasons why abstraction is used, thus scoring no marks for not answering the question. Again, there were also many vague and unqualified responses. Responses such as 'focus on key aspects' did not identify the reason why abstraction is used in the design, whereas clearing specifying 'focus on key aspects allows coding/development time to be saved' would make the reason clear.</td></tr>
<tr><td></td><td></td><td></td><td>Total</td><td>7</td><td></td></tr>
</table>

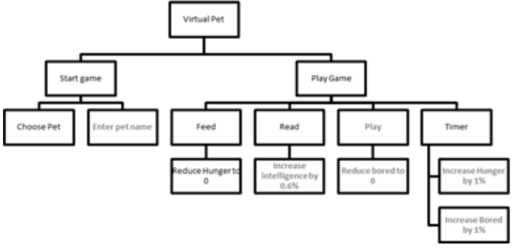
Question			Answer/Indicative content	Marks	Guidance
6			1 mark for definition - Removal of unnecessary detail // Simplification to allow development of a program more easily 1 mark to max 2 for application e.g. - The actual movements are represented by vertices/lines - State of the move is represented by a letter/symbol rather than the actual move position - Tree does not show details about what the moves are	3 AO1.1 (1) AO2.1 (1) AO2.2 (1)	Allow other suitable examples that are relevant to the scenario in the question.
			Total	3	
7			e.g. - Reduces track scenery - Limited functionality on car dashboard - Simplified controls - Simplified physics - Simplified / removed weather	2	Accept any reasonable answer **Examiner's Comments** Many candidates confused the concept of abstraction (simplification) with the requirement to make a genuinely realistic simulation.
			Total	2	

Question			Answer/Indicative content	Marks	Guidance
8	a		• A real town contains things that aren't relevant to the simulation (1) which would require unnecessary programming / design effort (1). • … would require extra computational resources … (1). • … could detract from the main purpose of the program (1).	2	Up to 2 marks for a valid explanation.
	b	i	• Road signs / road markings (1) – so the user can practise obeying these when driving (1). • Traffic Lights (1) – so user can practise obeying traffic light signals (1). • Zebra crossing (1) – so user can practise slowing down / stopping at zebra crossing (1). • Cars / vehicles (1) – so user can practice driving with other cars on the road (1). • Pedestrians (1) – so user can practice looking out for and avoiding pedestrians (1).	4	1 mark for each correct identification up to a maximum of two identifications plus up to a further 1 mark for each of two valid explanations.
		ii	• Scenery may be simplified (1). • Smaller roads may be removed (1). • Potholes may be removed (1). • Buildings may be simplified (1). • Imperfections / wear / damage in road markings and signs will be ignored (1). • No need to worry about sounds of real town (1).	2	1 mark for each correct identification up to a maximum of two identifications.
			Total	8	
9			1 mark for each input to max 2 • Username • Password 1 mark for output e.g. • Message to request input • Message to state login successful • Message to say login unsuccessful	3	<u>Examiner's Comments</u> This question was generally well answered by most candidates, but answers had to be given in the context of the scenario. Some less successful responses mistakenly identified input/output devices rather than specific examples of inputs and outputs.
			Total	3	

Question			Answer/Indicative content	Marks	Guidance
10			• 10	1 A03.2 (1)	
			Total	**1**	
11			1 mark pet bullet to max 3 • Descending order • Line 07 `(dataArray[tempos]<temp)` has the comparison… • …that checks if current position is less than item to insert and… • …breaks out of loop when current position is less than or equal to item to insert	3 AO1.2 (1) AO2.2 (2)	
			Total	**3**	
12			1 mark for the purpose and 1 mark for matching appropriate name (4 marks total), e.g: • Pointer to the first element in the queue • firstElement // any other meaningful name • Pointer to the last element in the queue // Pointer to the first free element in the queue • lastElement / any other meaningful name	4 AO1.2 (4)	Must cover purpose and name for 2 marks for each pointer.
			Total	**4**	

Question			Answer/Indicative content	Marks	Guidance
13			1 mark each to max 2: • One piece of code can be used many times / in multiple places / makes code more efficient • No need to write the same code multiple times • Takes less time to plan/design/code the program • Easier error detection as fix once and it corrects in each place // less likely to have errors as code is not written multiple times • Makes it easier to maintain the program	2	Examiner's Comments Many less successful responses gave vague generalities such as 'saves time' or 'more efficient' without specifying why or how. Points given must be qualified in some way at A Level. For example, 'saves development time as pre-written routines are available'. Pre-written or pre-tested, and saving development time due to already being written, were the most popular answers.
			Total	2	
14			1 mark per bullet to max 3 e.g. • Initialising a new board with 100 squares • Placing players on square 1 at the start • Setting a new 30-minute timer • Rolling two dice and displaying the total • Moving the counters the number of places on the dice • Choosing an instruction card and displaying the message • Checking if a player has landed on square 100 • Checking if the timer is 0	3	Allow other suitable examples that are relevant to the scenario. Examiner's Comments Some candidates struggled to apply decomposition to identify parts of the scenario that could be broken down into sub problems. Examples needed to relate to the gameplay for the given scenario and needed to be qualified in some way. For example, 'Dice' on its own was insufficient, whereas 'Generating dice score' was sufficient.
			Total	3	

Question			Answer/Indicative content	Marks	Guidance
15		i	1 mark per bullet, max 2 per sub-procedure e.g. • Select character (name, gender) • Gives the user options for choosing a character • Choose level • Give the user the choice of level (easy, normal, challenging) and take the user input • Touch enemy • Called to determine if the character touches an enemy • Lose life • Remove a life, if <0 then game over • End level • Move onto next level One mark for identifying sensible subroutine, 1 mark for description	6 AO2.1 (2) AO2.2 (2) AO3.2 (2)	Do not award any user *input* related procedures e.g. Left/Right input (but character movement *output* on screen left/right would be valid) Allow other reasonable responses from the scenario e.g. generate enemy() **Examiner's Comments** Most candidates presented reasonable procedures within the context of the question. Relatively few made the mistake of reiterating procedures to handle character movement that was specified as an example in the question.
		ii	1 mark per bullet • Decision based on what the user has input • E.g. If they click left move the character left // if they click right move the character right // if they click space bar make the character jump	2 AO2.1 (1) AO2.2 (2)	**Examiner's Comments** A number of answers lacked the clarity of a clear decision and subsequent action that would be performed. A number of answers given did not relate to the user input sub-procedure as specified.
		iii	1 mark per bullet • The result from one process / procedure feeds into the next • E.g. the result of detecting a character touching an enemy feeds into reducing the number of lives	2 AO1.2 (1) AO2.2 (1)	Note: 1 Mark Max for a generic description of pipelining **Examiner's Comments** Many candidates could define the concept of pipelining. Fewer were able to apply it within the context of the given scenario.
			Total	10	

Question			Answer/Indicative content	Marks	Guidance
16		i	1 mark per bullet to max 2 - Splitting a problem down - Into its component parts/sub-procedures/modules	2 AO1.1 (2)	**Examiner's Comment:** Nearly all candidates scored full marks for factual recall of the required definition.
		ii	1 mark per box  Virtual Pet Start game Play Game Choose Pet Enter pet name Feed Read Play Timer Reduce Hunger to 0 Increase intelligence by 0.6% Reduce bored to 0 Increase Hunger by 1% Increase Bored by 1%	6 AO2.2 (6)	Calculations must be correct **Examiner's Comment:** Nearly all candidates achieved three or more marks after analysing the requirements in the stem of the question. A number gave incorrect multiplying factors for some of the required elements and thus lost marks where mathematical accuracy was required.
			Total	8	
17			1 mark for each component e.g. • Allocating cards to each player • Generating the deck • Managing whose turn it is to play • Checking won	3	Accept any reasonable component **Examiner's Comments** The specification requires candidates to be able to identify elements of computational thinking. As such, candidates are expected to be able to think procedurally and to be able to identify the components of a problem. While analysis of the problem given in context led most candidates to identify valid components, many struggled to read the scenario and to give relevant points. For instance, many reiterated aspects of checking if a move was valid, which was already given in the question.
			Total	3	

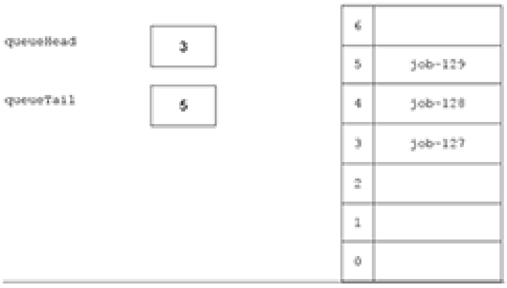
Question			Answer/Indicative content	Marks	Guidance
18		i	1 mark for: - `isInteger` - `number` - `result` - `count` - `asciiValue`	1	Penalise excessive spaces in identifiers such as *ascii Value* instead of *asciiValue* Examiner's Comments This question was generally well done (although slightly less well done than parts (a) (ii) and (a) (iii)), but many candidates did very well. The most common erroneous responses were giving the names of predefined functions/properties or giving relational operators.
		ii	(0)5	1	Examiner's Comments This question required an exact answer only and was answered correctly by the majority of candidates.
		iii	(0)3	1	Examiner's Comments This question also required an exact answer only and was answered correctly by the majority of candidates.
			Total	3	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>19</td><td></td><td></td><td>1 mark for final return value 29 (award in working or answer space) 1 mark each for working
<ul><li>First call with 10 and second call with 7</li><li>Remainder of calls 6, 3, 2</li><li>Final call value -1</li><li>Adding/showing return values (1 + 2 + 3 + 6 + 7 + 10)</li></ul>e.g.
<table>
<tr><th>Function call</th><th><code>value</code></th><th><code>return</code></th></tr>
<tr><td><code>recursiveAlgorithm(10)</code></td><td>10</td><td>29</td></tr>
<tr><td><code>recursiveAlgorithm(7)</code></td><td>7</td><td>19</td></tr>
<tr><td><code>recursiveAlgorithm(6)</code></td><td>6</td><td>12</td></tr>
<tr><td><code>recursiveAlgorithm(3)</code></td><td>3</td><td>6</td></tr>
<tr><td><code>recursiveAlgorithm(2)</code></td><td>2</td><td>3</td></tr>
<tr><td><code>recursiveAlgorithm(–1)</code></td><td>–1</td><td>1</td></tr>
</table></td><td>5</td><td>The table is given as guidance, but actual process may be presented in different ways.

<u>Examiner's Comments</u>

While many candidates continue to find recursion a challenging topic there were many who encouragingly achieved full marks. Weaker candidates traced the initial sequence of calls but found it harder to identify the last call to <code>recursiveAlgorithm(-1)</code> that triggered the base case and then found it harder again to calculate the unwind sequence.</td></tr>
<tr><td></td><td></td><td></td><td>Total</td><td>5</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>20</td><td></td><td></td><td>1 mark per bullet
- 1st swap of 5 and 3
- Remainder of first pass
- Pass 2
- Pass 3
<table>
<tr><td>1</td><td>5</td><td>3</td><td>9</td><td>2</td><td>7</td><td></td></tr>
<tr><td>1</td><td>**3**</td><td>**5**</td><td>9</td><td>2</td><td>7</td><td></td></tr>
<tr><td>1</td><td>3</td><td>5</td><td>**2**</td><td>**9**</td><td>7</td><td></td></tr>
<tr><td>1</td><td>3</td><td>5</td><td>2</td><td>**7**</td><td>**9**</td><td>End of pass 1</td></tr>
<tr><td>1</td><td>3</td><td>2</td><td>5</td><td>7</td><td>9</td><td>End of pass 2</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>5</td><td>7</td><td>9</td><td>End of pass 3</td></tr>
</table></td><td>4</td><td>Candidates do not need to show each swap, so if the candidate has clearly shown the end of pass 1, they have met the first two marking points.
Marks can be awarded for correctly showing the results of each pass.
Examiner's Comments
Most candidates clearly showed the steps that would take place in a bubble sort for the date given, with most achieving full marks. Some candidates did not explicitly label each pass as required in the question, but marks were given where the passes could be implied.
Few candidates described the principles of a bubble sort instead of applying it to the data given.</td></tr>
<tr><td></td><td></td><td></td><td>Total</td><td>4</td><td></td></tr>
<tr><td>21</td><td></td><td></td><td>- 1,3,5,7</td><td>1
(AO3.3)
(1)</td><td>Examiner's Comments
Candidates who answered the previous part of the question generally went on to score full credit for a second trace of the algorithm with a different calling argument value.</td></tr>
<tr><td></td><td></td><td></td><td>Total</td><td>1</td><td></td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
22	a		1 mark per pointer • queueHead: Point to the first element in the queue // next element to remove • queueTail: Point to the last element in the queue	2 AO1.2 (2)	
	b		1 mark per bullet up to max 5 • first 3 jobs removed • 128 and 129 added in positions 4 and 5 respectively • no additional jobs • queueHead being 3 (FT errors) • queueTail being 5 (FT errors)  queueHead 3 queueTail 5 6 5 job-129 4 job-128 3 job-127 2 1 0	5 AO2.1 (2) AO2.2 (3)	The underlying implementation of the queue has not been specified, so allow alternative valid answers. e.g. queueHead = 0 queueTail = 2 Location 2: 129 Location 1: 128 Location 0: 127

	c	i	1 mark per bullet to max 5
			• Function declaration
			• Checking if queue is empty
			• …returning `null`
			• (Otherwise) incrementing `queueHead`
			• …returning `buffer[queueHead-1]`
			
			e.g.
			`function dequeue()`
			`   if queueHead > queueTail then`
			`     return null`
			`   else`
			`     queueHead = queueHead + 1`
			`     return buffer[queueHead-1]`
			`   endif`
			`endfunction`
				5
				AO2.2
				(2)
				AO3.3
				(3)
					Note: Accept alternative valid underlying implementation answers e.g. Shifting all elements in queue forward.

Question			Answer/Indicative content	Marks	Guidance

		ii	1 mark per bullet to max 6
			• Function declaration taking parameter
			• Checking if queue is full
			• …returning -1
			• (Otherwise) incrementing queueTail
			• Adding newJob to buffer(queueTail)
			• Returning 1
			e.g.
			`function enqueue(newJob)`
			`   if queueTail == 99 then`
			`     return -1`
			`   else`
			`     queueTail = queueTail + 1`
			`     buffer[queueTail] = newJob`
			`     return 1`
			`   endif`
			`endfunction`
				6
				AO2.2
				(3)
				AO3.3
				(3)	

Question			Answer/Indicative content	Marks	Guidance

		iii	1 mark per bullet to max 8
			• Inputting user choice
			• If enqueue chosen input job name
			• …call enqueue with input value as parameter
			• …check if return value is -1 and output full
			• …otherwise output message that item is added
			• If dequeue chosen
			• …call dequeue **and** save returned value
			• …output returned value (jobname) if not null
			• …or output queue is empty
			
			e.g.
			`main()`
			`    choice = input("Add or remove?")`
			`    if choice == "ADD" then`
			`      jobname = input("Enter job name")`
			`      returnValue = enqueue(jobname)`
			`      if returnValue == -1 then`
			`        print("Queue full")`
			`      else`
			`        print("Job added")`
			`      endif`
			`    else`
			`      returnValue = dequeue()`
			`      if returnValue == null then`
			`        print("Queue empty")`
			`      else`
			`        output returnValue`
			`      endif`
			`    endif`
			`endmain`
				8
				AO2.2
				(2)
				AO3.3
				(6)
					Allow equivalent checks / logic

	d		1 mark per bullet to 3 • Check if either head or tail are incremented to above 99 • … set to be 0 instead • When checking if array is full check if (queueTail == queueHead – 1) OR (queueTail==99 AND queueHead==0)	3 AO2.1 (1) AO2.2 (2)	Credit equivalent modulo arithmetic solution

Question			Answer/Indicative content	Marks	Guidance
	e		1 mark per bullet to max 3, e.g. • Use a different structure e.g. a linked list • …items can be added at different points in the linked list depending on priority • …by changing the pointers to items needing priority • Have different queues for different priorities • …add the job to the queue relevant to its priority • …print all the jobs in the highest priority queue first	3 AO2.1 (2) AO2.1 (1)	Allow other suitable descriptions that show how the program could be amended.
			Total	32	

Question			Answer/Indicative content	Marks	Guidance
23		i	1 mark per bullet, max 2 for insert, max 2 for remove push - Check if the stack is full (pointer = array.length/array.length+1) - If it is not ¡V insert the item - If it is ¡V return/error that the stack is full pop - Check if the stack is empty (pointer = 0/1) - If it is ¡V return/error that the stack is empty - If it is not ¡V return the item	4 AO1.2 (2) AO2.2 (2)	<u>Examiner's Comments</u> A significant number of candidates did not describe a conditional decision clearly and lost marks when merely describing push/pop operations. Where the first part of the question was answered well some candidates then failed to see the second part of the question and did not describe an impact of the condition. Candidates need to be reminded to read and analyse the wording of the whole question to access all marking points.
		ii	1 mark per line, 1 for change - line 02 - Include an OR with variations (e.g. `userAnswer = "PUSH" OR userAnswer = "Push"` etc.)/Convert input to uppercase/lowercase and just compare to equivalent	2 AO2.2 (2)	<u>Examiner's Comments</u> Most candidates answered well, but few gave answers that demonstrated an ability to combine separate logical statements with the OR operator. A significant number of candidates used a .lower() method being mostly familiar with Python syntax and methods.
			Total	6	
24			- Iteration … - … Example should be either a FOR, WHILE or REPEAT … UNTIL - Selection … - … Example should be either an IF, SWITCH, CASE or SELECT - Sequence … - … Example should be at least two consecutive lines of code	6	Do not accept 'loop' All examples must be code, do not credit explanations. Allow examples from given code <u>Examiner's Comments</u> This was largely well answered, though some of the code examples for iteration were for infinite loops.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
25			• Barcode of item purchased is read at checkout / it is scanned in • Barcode is compared with barcodes stored on stock file • (Field containing) number of tins of beans is decremented • New value is compared with the field containing the minimum number of tins of beans that are allowed • If number of tins in stock is less than minimum stock / value is below limit… • …and no order is outstanding for this item… • …search supplier file for details of supplier and use these details either to place an order automatically or produce report for manager • Set field showing outstanding order • When order arrives, number in stock (field) is incremented	6	Note: This is intended to be a difficult question. Mark points need to be fairly precise – do not read too much into a response **Examiner's Comments** There were some comprehensive responses here, while many good answers were spoiled because the process described did not consider the need to avoid automatically ordering more tins after every tin is sold. Few were unable to earn some credit even if it was only for scanning the tins when sold. This proved to be an excellent discriminator question. This question was an ideal question to be answered as a series of numbered points, answering in this way could have helped some candidates arrange their thoughts in what is a sequential process.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
26			* **Mark Band 3–High Level** **(7–9 marks)** The candidate demonstrates thorough knowledge and understanding of reasons why Nobugs enforces standard rules about writing functions on its programmers; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence / examples will be explicitly relevant to the explanation. The candidate provides a thorough discussion which is well-balanced. Evaluative comments are consistently relevant and well-considered. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2–Mid Level** **(4–6 marks)** The candidate demonstrates reasonable knowledge and understanding of reasons why Nobugs enforces standard rules about writing functions on its programmers; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence / examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1–Low Level** **(1–3 marks)** The candidate demonstrates a basic knowledge of reasons why an organisation enforces standard rules about writing functions on its programmers with limited understanding shown; the material is basic	9	**AO1: Knowledge and Understanding** The following is indicative of possible factors / evidence that candidates may refer to but is not prescriptive or exhaustive: • No function may be longer than a single page of code: this is to reduce complexity and aid readability. • Variable identifiers must conform to a standard convention: this helps others to understand the code and reduces the likelihood of duplication, makes maintenance easier. • Each function must have a single entry point: this reduces complexity and makes the search for any bugs more straightforward. • Variables must not be set up outside the scope of a function: this sets a limit on where to look for bugs and reduces the likelihood of a problem spread across many modules. **AO2.1: Application** The selected knowledge / examples should be directly related to the specific question. The following is indicative of possible factors / evidence that candidates may refer to but is not prescriptive or exhaustive: • Explanation of how the standard rules for programming would impact upon the choices made for using functions and variables and how they are addressed. • Discussion around the use of different functions and variables that are dependent, independent or interdependent. **AO3.3: Evaluation** Candidates will need to consider a variety of viewpoints in relation to following standard rules for functions and variables while developing management software and will make evaluative comments about the issues and solutions they are discussing e.g. • Why using functions longer than one page of code will increase complexity? • Why hardware-specific code must be avoided?

Question			Answer/Indicative content	Marks	Guidance
			and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgments if made are weak and unsubstantiated. *The information is basic and communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		• Why variables must not be setup outside the scope of a function? • How a single entry point reduces complexity and makes the search for any bugs more straightforward? • What will happen if embedded documentation is not adequate?
			Total	9	
27			• Have a Boolean variable for if user has obeyed this rule (1) and set it to true (1). • Check if on the no overtaking section of road (1). • When entering this section make note of the car immediately in front (1). • If the position of this car becomes behind the user set a flag that this rule has been broken (1) / alert the user that rule has been broken (1). • Continue checking until left the overtaking section (1).	3	Up to 3 marks for a valid description.
			Total	3	

Question			Answer/Indicative content	Marks	Guidance
28		i	1 mark for each to max 2 • Processes happen at the same time // processes overlap • One process can start before another one finishes • Each process is given a slice of processor time • Different processes can be executed (in parallel) by different processors/cores	2	**Examiner's Comments** Several candidates confused parallel processing (executing more than one instruction simultaneously) with concurrent processing (where most than one process/task is running at the same time). A lack of technical vocabulary was observed with many candidates giving responses such as 'many things processed at the same time', without specifying what the 'things' were. Occasionally candidates erroneously thought that processor pipelining was an example of concurrent processing. While many candidates successfully identified that multiple processes are run simultaneously, fewer went on give the mechanism by which this could be achieved. Where they did so, these included timeslicing switching between different processes on a single processor, or use of multiple cores/parallelism to simultaneously execute different processes.
		ii	1 mark each to max 2 e.g. • More efficient processor use // Less idle time for processor // Greater throughput • Long running tasks do not delay short running tasks • Tasks requiring preconditions can wait and then resume execution • User is able to interact with the computer while other tasks are running	2	**Examiner's Comments** Candidates found it difficult to give well-qualified responses to this question. Many candidates gave the definition for concurrency running multiple tasks at the same time as a benefit, which was not mark worthy. Quicker processing/improved performance was not enough on its own without specifying that this was within a given time unit. Concurrent processing does not increase the actual speed of the CPU. Efficiency as a benefit on its own was insufficient, whereas 'less CPU idle time' was a well-qualified example of a benefit.
			Total	4	

Question			Answer/Indicative content	Marks	Guidance
29		i	1 mark per bullet to max 2 e.g. • Multiple processes being executed at the same time // appearing to happen simultaneously • Giving processes a slice of the processor time • Having multiple processors each carrying out a different process	2 AO1.1 (2)	
		ii	1 mark per bullet to max 3 e.g: • Game could have large number of requests to the server at a time • … server needs to respond in reasonable time • … having multiple processors handling the different requests would increase response time • Users could override each other's changes • … e.g. needs to handle if someone updates their circus while someone else is visiting • …use record locking to stop edits if someone else has access to data • Different users will have different response times • …therefore the processor can still handle other requests • …so that the performance for other users is not affected	3 AO2.1 (2) AO2.2 (1)	
			Total	5	

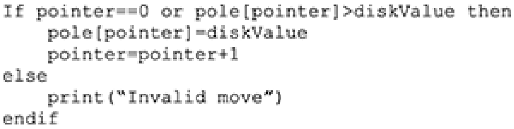
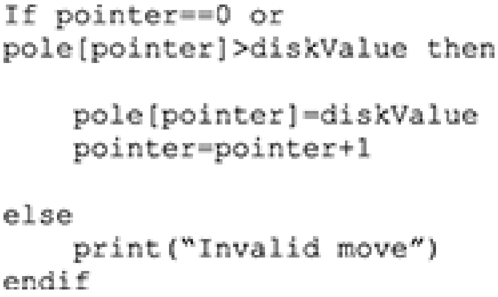
Question			Answer/Indicative content	Marks	Guidance
30			**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a **thorough** knowledge and understanding of concurrent programming; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence / examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured.* *The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates **reasonable** knowledge and understanding of concurrent programming; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence / examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a **basic** knowledge of concurrent programming with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	**AO1: Knowledge and Understanding Indicative content** • Processes are happening at the same time / at overlapping times • Only 1 process can actually happen at a time on a single core processor, concurrent tries to simulate multiple processes • One process may need to start before a second has finished • Individual processes are threads, each thread has a life line **AO2: Application** • Multiple orders can be made and added to the list at the same time • Programming will need to allow multiple threads to manipulate a single list • Will allow those reading and writing to manipulate at the same time • Locking will need implementing – more complex programming **AO3: Evaluation** • Will allow for multiple orders at the same time – as it would happen in real life • Access to the linked list will need to be limited so it cannot be accessed / overwritten by two threads trying to do different operations • Not all of the process will be paralleliseable. X processors does not mean it will run in 1/xth of the time of one processor.

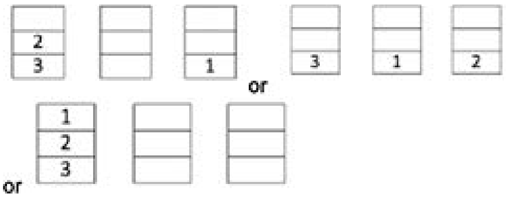
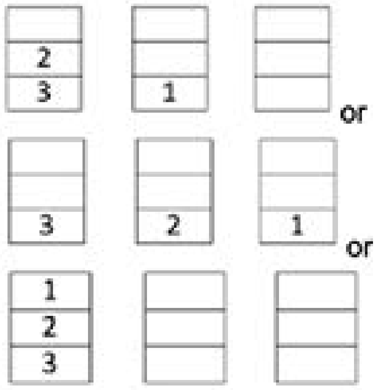
Question			Answer/Indicative content	Marks	Guidance
			unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		**Examiner's Comment:** It was clear that many candidates had not covered the concept of concurrency and how it allows different processes to occur at the same time. Strong candidates appreciated that this could be simulated on a single core with time slicing or implemented within a parallel architecture. Many candidates lost sight of the fact that answers needed to be related to computer science rather than a restaurant chain and could not explain the underlying computer science that would allow a solution to be delivered.
			Total	9	

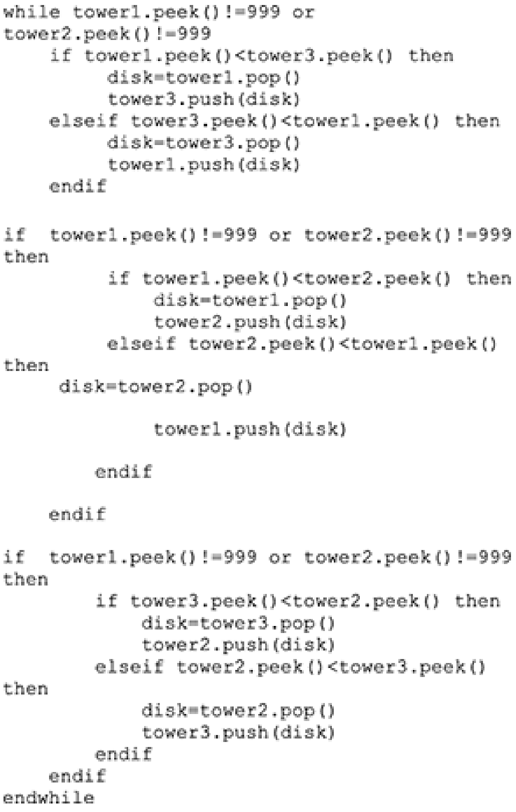
Question			Answer/Indicative content	Marks	Guidance
31			**Mark Band 3 – High level (7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of concurrent processing; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided (searching algorithms). Evidence/examples will be explicitly relevant to the explanation. The candidate provides a thorough discussion which is well balanced. Evaluative comments are consistenly relevant and well considered. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level (4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of concurrent processing; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. (searching algorithms) Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence* **Mark Band 1 – Low Level (1-3 marks)** The candidate demonstrates a basic knowledge of concurrent processing with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided (searching algorithms). The candidate provides a limited discussion which is narrow in focus.	9	**AO1: knowledge and understanding Indicative content** • Carrying out more than one task at a time • Multiple processors • Each processor performs simultaneously • Each processor performs tasks independently and/or • A program has multiple threads • Each thread starts and ends at different times • Each thread overlaps • Each thread runs independently **AO2: Application** • Each processor/thread performs a search in a different direction • Rather than going down one path, go down 2+ • E.g. apply different searches simultaneously - perform breadth- first and depth-first simultaneously • E.g. A* take the two shortest routes at each decision point, update same table • Linear search can have multiple processors searching different areas at the same time. • Binary search doesn't benefit from an increase in speed with additional processors. **AO3: Evaluation** Candidates will need to evaluate the benefits and drawbacks of concurrent processing in searching e.g. • Possibly find solution faster • Takes up more memory • Increase program throughput • May waste time investigating inefficient solutions • More difficult to program especially to cooperate • More memory intensive

Question			Answer/Indicative content	Marks	Guidance
			Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		• Linear search scales very with additional processors • Binary search can perform better on large data sets with one processor than linear search with many processors
			Total	9	

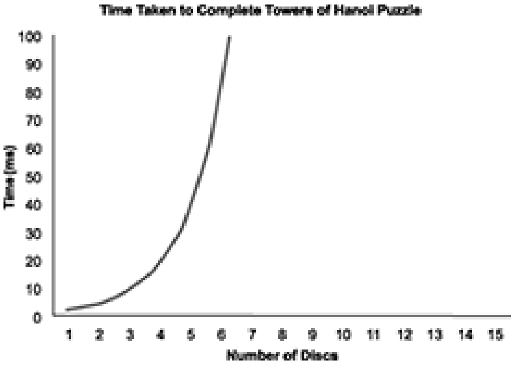
Question			Answer/Indicative content	Marks	Guidance
32	a	i	• A disk can only be put onto the top of the pole (1) and a stack is a last in first out structure (1) whereas a queue is first in first out (1).	2	Up to 2 marks for a valid explanation.

		ii	Individual steps in pseudocode:
			• Checks if pointer is 0 (i.e. pointer is 0)
			• Checks if the disk being added is smaller than the one at the top of the pole
			• Puts diskValue at the top of stack
			• Move the pointer up one
			• If it's not a valid move, no disk is added
			• …and prints Invalid move
			Programming marks to be awarded as follows:
			• Checks if pointer is 0 and checks if the disk being added is smaller than the one at the top of the pole (1 – AO 3.2).
			• Puts diskValue at the top of stack and moves the pointer up one (1 – AO 3.2).
			• If it's not a valid move, no disk is added and prints Invalid move (1 – AO 3.2).
			Possible annotated comments:
			• The IF statement checks the pointer value is 0 so we can assume the disk is available to be moved (1 – AO 2.2).
			• If these conditions are met then pole value is set to the diskValue and incremented by 1 so the disk can move to the next pole (1 – AO 2.2).
			• If these conditions are not met then the else prints Invalid move (1 – AO 2.2).
				6	Up to 3 marks for valid pseudocode (AO3.2).
					Up to 3 marks for annotated comments used (AO2.2).
					Example pseudocode:
					
					`If pointer==0 or pole[pointer]>diskValue then`
					`    pole[pointer]=diskValue`
					`    pointer=pointer+1`
					`else`
					`    print("Invalid move")`
					`endif`
					Example of pseudocode with comments in code for guidance:
					
					`If pointer==0 or`
					`pole[pointer]>diskValue then`
					`    pole[pointer]=diskValue`
					`    pointer=pointer+1`
					`else`
					`    print("Invalid move")`
					`endif`

Question			Answer/Indicative content	Marks	Guidance
	b	i	Left hand boxes, any 2 of:  (1 Mark each, Max 2) Right hand boxes, any 2 of:  (1 Mark each, Max 2)	4	For 4 marks as indicated.
		ii	**Depth first advantage:** • Depth first requires less memory than breadth first search (1). It is quicker if you are looking at deep parts of the tree (1). **Depth first disadvantage:** • Depth first isn't guaranteed to find the quickest solution (1) and possibly may never find the solution (1) if we don't take precautions not to revisit previously visited states (1).	4	Up to 4 marks for a valid description.

Question			Answer/Indicative content	Marks	Guidance
	c		Individual steps in pseudocode: Makes a swap between 1 and 3 …and always makes a valid swap. Then makes a swap between 1 and 2. …And always makes a valid swap …only if the problem is not solved. Repeats if not solved. Will not attempt to make a swap between 2 empty towers. Programming marks to be awarded as follows: • Makes a swap between pole 1 and 3 (1 – AO 3.2). • Checks for a valid move throughout (1 – AO 3.2). • Makes a swap between pole 1 and 2 (1 – AO 3.2). • Checks whether problem is solved throughout (1 – AO 3.2). • Makes a swap between 3 and 2 (1 – AO 3.2). • Repeats if not solved and will not attempt a swap between two empty towers. (1 – AO 3.2). Possible annotated comments: • The loop checks if towers 1 or 2 have anything on them. If they do the problem is not solved as all the disks need to be on pole 3. (1 – AO 2.2). • The first selection (if) makes a swap between 1 and 2 and always makes a valid swap only if the problem is not solved. (1 – AO 2.2). • The second selection (if) makes a swap between 3 and 2 and always makes a valid swap only if the problem is not solved. (1 – AO 2.2). • A swap is only valid if the pole is either empty or the disk on which it is to be put is a larger number (1 – AO 2.2).	10	Up to 6 marks for valid pseudcode (AO3.2) Up to 4 marks for annotated comments used (AO2.2). Example pseudocode: 
	d	i	32 ms (1)	1	For one mark.

```
while tower1.peek()!=999 or
tower2.peek()!=999
    if tower1.peek()<tower3.peek() then
         disk=tower1.pop()
         tower3.push(disk)
    elseif tower3.peek()<tower1.peek() then
         disk=tower3.pop()
         tower1.push(disk)
    endif

if  tower1.peek()!=999 or tower2.peek()!=999
then
         if tower1.peek()<tower2.peek() then
             disk=tower1.pop()
             tower2.push(disk)
         elseif tower2.peek()<tower1.peek()
then
     disk=tower2.pop()

             tower1.push(disk)

        endif

    endif

if  tower1.peek()!=999 or tower2.peek()!=999
then
        if tower3.peek()<tower2.peek() then
            disk=tower3.pop()
            tower2.push(disk)
        elseif tower2.peek()<tower3.peek()
then
            disk=tower2.pop()
            tower3.push(disk)
        endif
    endif
endwhile
```

Question			Answer/Indicative content	Marks	Guidance
		ii	Big O notation shows the limiting behaviour of an algorithm (to classify its complexity) (1). • Other processes may be taking up some of the processor time (1).	1	For one mark.
		iii	Exponential curve drawn (1). Curve passes 100 on the y axis before x reaches 8 (1).	2	For two marks. 
	e		**Mark Band 3–High Level** **(7–9 marks)** The candidate demonstrates thorough knowledge and understanding of concurrency including considerations of the search space tree and iterative algorithm approaches; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence / examples will be explicitly relevant to the explanation. The candidate provides a thorough discussion which is well–balanced. Evaluative comments are consistently relevant and well–considered. *There is a well–developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2–Mid Level** **(4–6 marks)** The candidate demonstrates reasonable knowledge and understanding of concurrency including considerations of the search space tree and iterative algorithm approaches; the material is generally accurate but at times underdeveloped.	9	**If only search space tree or iterative algorithm approaches considered – MAX 5 marks.** **AO1: Knowledge and Understanding Indicative Content:** The following is indicative of possible factors / evidence that candidates may refer to but is not prescriptive or exhaustive: • Concurrency could be used with the search tree. • Different processors could generate the subtrees of different nodes until the correct answer is found. • With the iterative algorithm each stage depends on the previous one meaning it is not possible to use concurrent processing to speed up the process. • Because of the exponential complexity as n increases concurrent processing becomes less practical. **AO2.1: Application** The selected knowledge / examples should be directly related to the specific question. The following is indicative of possible factors / evidence that candidates may refer to but is not prescriptive or exhaustive:

Question			Answer/Indicative content	Marks	Guidance
			The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence / examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1–Low Level** **(1–3 marks)** The candidate demonstrates a basic knowledge of concurrency with limited understanding shown; the material is basic and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgments if made are weak and unsubstantiated. *The information is basic and communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		• Discussions around abstraction and modelling. • Discussions around the preconditions and how these will be applied. • Discussions around identifying the components of the problem and how concurrency might be used. • Discussions around order of steps and sub processes. • Discussion around the flow of the program and the decisions taken and how this effects the use of concurrency. **AO3.3: Evaluation** Candidates will need to consider a variety of factors in relation to the question and will make some evaluative comments about the issues and solutions they are discussing. The following is indicative of possible factors / evidence that candidates may refer to but is not prescriptive or exhaustive: • Discussions of the features and how the different approaches are more or less suitable for a solution. • Discussions of the essential features of the solutions and the differences between them with discussion of the limitations.
			Total	39	

Question			Answer/Indicative content	Marks	Guidance
33			Max 1 for explanation of concurrent programming. Max 3 for each example. Concurrent processing: • One process does not have to finish before the other starts (1) Example e.g. • Each plane can move independently (1) • All move at the same time (1) • All need to react to different events (1) • The weather (1) • Wind, rain, direction of air etc. (1) • Each element needs to be run simultaneously (1) • It will react to its own stimulii (1)	4	Accept any reasonable suggestion for concurrent programming in the simulator For examples: 1 mark for identifying example. 1 mark for saying how they act concurrently. 1 mark for saying why this is necessary.
			Total	4	
34			• Instruction processing (1) – some processors allow parts of instructions to be processed (1) without waiting to complete the whole instruction cycle (1). • Pipes to pass data between programs (1) from programs to peripherals / to programs from peripherals (1), example such as | symbol in Unix, or • Popen() or pipe() in C (1). • Graphics pipelines (1) separate processor renders graphics from data supplied by other processes (1), parts (vertices) of the image are pipelined at the same time as custom software (shader) that renders the display (1).	4	1 mark for each correct identification up to a maximum of two identifications plus up to a further 1 mark for each of two valid descriptions.
			Total	4	

Topic 2

Problem Solving and Programming

1 A programmer is designing a program that will store data.

The programmer is deciding whether to store the data in a stack or a queue.

The pseudocode function, enqueue, inserts an item into a queue.

```
function enqueue(item)
    if tailPointer >= queue.length then
      return false
    else
      queue[tailPointer] = item
      tailPointer = tailPointer + 1
      return true
    endif
endfunction
```

i. Give the name of the parameter in the function enqueue.

[1]

ii. Give the name of **one** global variable that is used in the function enqueue.

[1]

iii. Describe **one** benefit and **one** drawback of using global variables instead of parameter passing in a subroutine.

Benefit

Drawback

[4]

iv. The function enqueue can be called by the main program.

Explain why the function enqueue returns true or false values, and how this can be used by the main program that calls the function.

[3]

v. The pseudocode function, dequeue, removes and returns the first item in the queue. If the queue is empty, the function returns the string "EMPTY".

```
01  function dequeue(data)
02      if headPointer != tailPointer then
03        return "EMPTY"
04      elseif
05        value = queue[headPointer]
06        return value
07        headPointer = headPointer + 1
08      endif
09  endfunction
```

The function dequeue has **several** errors.

Identify the line number of any **three** errors **and** state the correction required.

Error 1 Line Number

Error 1 Correction

Error 2 Line Number

Error 2 Correction

Error 3 Line Number

Error 3 Correction [3]

vi. The programmer has corrected all of the errors in the function dequeue.

The main program repeatedly calls the function dequeue until all of the elements in the queue have been output.

Write the main program using pseudocode or program code.

[3]

2(a) A game is being written that makes use of object-oriented programming. A prototype for one part of the game is being designed that includes a character, a road and a prize to collect.

The road will have 50 spaces that a character can move along. Each space on the road will store a null value or a prize object for the user to collect. Each space is numbered sequentially from the first space (position 0) to the last space (position 49) and will not change during the game. As the player travels down the road, the position the player is on the road will be output.

The road is designed to be a 1-dimensional array with the identifier `road`.

Explain why an array is a suitable data structure to represent the road.

[3]

(b) The characters and prizes are designed as separate classes. 10 of the spaces on the road will contain an instance of the class `Prize`. The other spaces will be empty.

The class design for `Prize` is here.

```
class: Prize
attributes:
private name : string
private type : string
private value : integer
methods:
new()
getName()
getType()
getValue()
```

`new()` is the constructor method. The name, type and value are passed to the constructor as parameters which then assigns these to the attributes.

i. The method `getName()` returns the data in the attribute `name`.
Write the method `getName()` using pseudocode or program code.

[2]

ii. A global 1-dimensional array, `allPrizes`, stores 10 objects of type `Prize`.

The prize in index 3 has the name “Box”, the type is “money” and the value is 25.

Write pseudocode or program code to create a new object for this prize and store it in index 3 of `allPrizes`.

[3]

iii. The game starts with 10 prizes. Each prize is allocated to one space on the road.

An algorithm needs designing that will generate a random space on the road for each prize. Each road space can only store one prize.

Describe the decisions that will need to be made in this algorithm and how these will affect the program flow.

[3]

(c) The class design for `Character` is here.

```
class: Character
---------------------------------
attributes:
private name : string
private money : integer
private experience : integer
private roadPosition : integer
---------------------------------
methods:
new()
getName()
getMoney()
getExperience()
getRoadPosition()
changePosition()
updateValues()
```

The four get methods return the associated attribute.

The number of moves is passed to `changePosition()` as a parameter. The method adds this value to the character's position on the road.

The type and value of an object are passed to `updateValues()` as parameters. If the object is money the value is added to the character's money. If the type is experience the value is added to experience. If the type is neither money or experience no changes are made.

i. `new()` is the constructor method. The name of the character is passed into the constructor as a parameter. The constructor then initialises both the experience and road position of the character to 0 and initialises the amount of money to 5.

Write the constructor method for `Character` using either pseudocode or program code.

You do not need to declare the class, the attributes or any other methods.

[5]

ii. The type and value of a prize are passed as parameters to the method `updateValues`. If the type is money the value is added to the character's money. If the type is experience then the value is added to the experience. If the type is neither money or experience no changes are made.

For example, for the Character `player1`:

`player1.updateValues("money",10)` updates player1's money by 10

`player1.updateValues("experience",5)` updates player1's experience by 5

`player1.updateValues("foo",9)` has no effect on player1.

Write pseudocode or program code for the method `updateValues()`.

[5]

(d) This incomplete pseudocode algorithm:

- creates a new character with the name Jamal
- loops until the character reaches the end of the road
- generates a random number of spaces to move between 1 and 4 (including 1 and 4)
- moves the character and checks if the new space has a prize
- updates the character attributes if there is a prize
- outputs the character’s new attribute values.

Complete the pseudocode algorithm.

```
character1 = new ................................ ("Jamal")
newPosition = 0
while newPosition < ................................
  move = random(1, 4) //this will generate a random number between 1 and 4
  character1.changePosition(move)
  newPosition = character1.getRoadPosition()
  if newPosition < 50 and road[..............................] != null then
  prizeType = road[newPosition].getType()
  valueAmount = road[newPosition].getValue()
  character1.updateValues(........................., valueAmount)
  print("Congratulations you are in position", newPosition, "and found",
  road[newPosition].getName())
  print("Money =", character1.getMoney(), "and experience =",
  character1. ..................... ())
  endif
.......................
print("You reached the end of the road")
```

[6]

(e) The procedure `displayRoad()` outputs the contents of each space in the road. The number of each space is output with either:

- the word “empty” if there is no prize
- the name of the prize if there is a prize.

```
01 procedure displayRoad()
02 for x = 0 to 60
03 print("Space", y)
04 if road[x] == null then
05 print("empty")
06 elseif
07 print(road[x].getValue())
08 endif
09 next x
10 endprocedure
```

The algorithm contains errors.

Give the line number of **four** different errors and write the corrected line for each error.

Error 1

Error line 1 ______________________________

Correction ______________________________

Error 2

Error line 2 ______________________________

Correction ______________________________

Error 3

Error line 3 ______________________________

Correction ______________________________

Error 4

Error line 4 ______________________________

Correction ______________________________ **[4]**

(f) A programmer is going to create a prototype for one small part of the game. Both `road` and `allPrizes` will be needed throughout the whole prototype. The programmer is considering making these global arrays as she thinks it will reduce the development time. Another programmer has suggested that doing this may create some problems when the rest of the game is created at a later stage.

Compare the use of global and local variables in this program.

You should include the following in your answer:

- the use of local and global variables
- alternative methods to using global variables
- the appropriateness of each to this program design.

[9]

3(a) A student has written this pseudocode algorithm:

```
01 a = 12
02 do
03 b = input("Enter a number")
04 until b >= 0 and b <= 100
05 for c = 1 to a
06 print(c * a)
07 next c
```

Rewrite lines 05 to 07 to use a while loop instead of a for loop.

You should write your answer using either program code or pseudocode.

[4]

(b) The student has used a do loop on line `02`.

Describe the difference between a do loop and a while loop.

-- **[2]**

4 A text-based computer game allows a user to dig for treasure on an island. The island is designed as a grid with 10 rows and 20 columns to store the treasure. Each square is given an x and y coordinate. Some of the squares in the grid store the name of a treasure object. Each treasure object has a value, e.g. 100 and a level, e.g. "Bronze."

The main program initialises a new instance of Board. The programmer is considering declaring this as a global variable or as a local variable and then passing this into the subroutines that control the game.

Compare the use of variables and parameters in this game.

You should include the following in your answer:

- what is meant by a local variable and global variable
- how local and global variables can be used in this program
- the use of passing parameters by value and by reference.

----

[9]

5 A programmer has designed a program that includes a reusable program component.

Describe the purpose of the following lines in the function `isInteger()`.

Line 03

Line 04

Line 09

[3]

6 A recursive pseudocode function, `recursiveAlgorithm()`, is shown.

```
function recursiveAlgorithm(value)
  if value <= 0 then
    return 1
  elseif value MOD 2 = 0 then
    return value + recursiveAlgorithm(value – 3)
  else
    return value + recursiveAlgorithm(value – 1)
  endif
endfunction
```

Describe the key features of a recursive algorithm.

You may refer to the function, `recursiveAlgorithm()` in your answer.

[3]

7 Layla writes a pseudocode algorithm to:

- input 20 positive numbers into a 0-indexed 1-dimensional array
- output the average (mean) number as a decimal
- output the smallest number
- output the largest number.

The pseudocode algorithm is shown. It contains various errors.

```
total = 1
smallest = 9999
largest = -1
for x = 0 to 21
    dataArray[x] = input("Enter a number")
    total = total + dataArray[x]
    if dataArray[x] < largest then
        largest = dataArray[x]
    endif
    if dataArray[x] < smallest then
        smallest = dataArray[x]
    endif
next x
print("Average = " + total * 20)
print("Smallest = " + smallest)
print("Largest = " + largest)
```

`dataArray` is defined as a local variable within the main program.

i. State what is meant by a 'local variable'.

[1]

ii. Give **one** benefit and **one** drawback of declaring `dataArray` as a local variable in the main program.

Benefit

Drawback

[2]

8 Nina is writing a computer game using an Integrated Development Environment (IDE). Her friend, James, is writing a computer game using a text-editor which will allow James to create and edit text. James will use a separate compiler.

Discuss the differences between writing and debugging a program using an IDE and a text-editor.

You should include the following in your answer:

- features that are used when writing code
- features that are used when debugging code
- the benefits of using an IDE instead of a text-editor.

[9]

9 A function, `toBinary()`, is needed to calculate the binary value of a denary integer between 0 and 255.

`toBinary()` needs to:

- take an integer value as a parameter
- divide the number by 2 repeatedly, storing a 1 if it has a remainder and a 0 if it doesn't
- combine the remainder values (first to last running right to left) to create the binary number
- return the binary number.

For example, to convert 25 to a binary number the steps are as follows:

25 / 2 = 12	remainder 1
12 / 2 = 6	remainder 0
6 / 2 = 3	remainder 0
3 / 2 = 1	remainder 1
1 / 2 = 0	remainder 1

return value = 11001

The main program:

- asks the user to enter a denary number between 1 and 255
- checks that the input is valid between 1 and 255
- If valid call the function `toBinary()` and pass the input as a parameter
- outputs the return value
- If not valid, repeatedly asks the user to input a number until the number is valid.

Write the algorithm for the main program.

You should write your algorithm using pseudocode or program code.

[4]

10 A card game uses a set of 52 standard playing cards. There are four suits; hearts, diamonds, clubs and spades. Each suit has a card with a number from; 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.

The card game randomly gives 2 players 7 cards each. The unallocated cards become known as the deck.

The players then take it in turns to turn over a card. A valid move is a card of the same suit or the same number as the last card played.

The winner is the first player to play all of their cards.

The cards are held in the 2D array `cards`. The first index stores the card number and the second index stores the suit, both as strings.

Write a pseudocode statement or program code to declare the array `cards`.

[2]

11 A program uses the recursive function `calculate()`. The function is written in pseudocode.

```
1. function calculate(number : byVal)
2.   if number == 1 then
3.     return number
4.   else
5.     return number + calculate (number - 1)
6.   endif
7. endfunction
```

i. Give the line number in the algorithm `calculate()` where a recursive call is made.

[1]

ii. State **two** features of any recursive algorithm.

Feature 1

Feature 2

[2]

12 A computer uses a stack data structure, implemented using an array, to store numbers entered by the user.

The array is zero based and has 100 locations.

The stack is programmed as an object using object-oriented programming. The design for the class, its attributes and methods are shown:

```
class: stack
-----------------------------------------
attributes:
private stackArray : Array of integer
private pointerValue : integer
-----------------------------------------
methods:
new()
function pop()
function push(value)
```

i. The method `pop()` returns the next value in the stack, or `-1` if the stack is empty.

Complete the pseudocode method `pop()`.

```
public function pop()
   if pointerValue == .............................................. then
     return ..............................................
   else
     pointerValue = pointerValue ..........................................
     returnValue = stackArray[..............................................]
     return ..............................................
   endif
endfunction
```

[5]

ii. The method push() accepts an integer as a parameter and adds it to the top of the stack unless the stack is already full.

If the push is successful the method returns true.

If the push is unsuccessful due to the stack being full the method returns false.

Write the method push() using either pseudocode or program code.

[6]

iii. The main program initialises a new object of type stack with the identifier mathsStack.

Write pseudocode or program code to declare the object.

[2]

iv. The main program needs to:

- take numbers as input from the user
- push them onto the stack `mathsStack` until the stack is full
- output an appropriate message if the stack is full.

Complete the pseudocode algorithm to meet these requirements.

```
returnValue = true
while returnValue == ..............................................
    returnValue = mathsStack.
........................................(input("Enter Number"))
    if returnValue == .............................................. then
     ............................................. ("Stack full")
    endif
```

```
  endwhile
```

[4]

v. The main program also needs to:

- remove one item from the stack at a time and add this to a total
- output the total every time an item is removed
- stop removing items when either the stack is empty, or 20 items have been removed.

Write pseudocode or program code to meet these requirements.

[8]

13 A programmer is developing an aeroplane simulator. The user will sit in a cockpit and the simulated environment will be displayed on screens around them.

Describe how caching can be used in the aeroplane simulator.

---

[2]

14(a) The array `words` is defined as a global variable and contains these values:

"house"	"boat"	"car"	"telephone"	"garden"	"spice"	"elephant"

The pseudocode function `useWords()` here uses the global array `words`. The number of words in the array `words` is passed as a parameter.

```
function useWords(numberOfWords : byVal)
  contents = ""
  for count = 0 to numberOfWords - 1
    contents = contents + words[count] + " "
  next count
  return contents
endfunction
```

i. Identify **two** variables in the function `useWords()`.

1 ______________________________

2 ______________________________

[2]

ii. `numberOfWords` is a parameter passed by value.

Describe the difference between passing a parameter by value and by reference.

---

[2]

iii. Rewrite the function `useWords()` to use a while loop instead of a for loop.

The function header and close have been written for you.

Write your answer using pseudocode or program code.

```
function useWords(numberOfWords : byVal)
```

```
endfunction
```

[4]

(b) Give **one** benefit and **one** drawback of declaring an array as a global variable instead of a local variable.

Benefit

Drawback

[2]

15 Given the following procedure:

```
procedure generate(number)
        a = 0
        while number > 0
             if number MOD 2 == 0 then
                  a = a + 2
                  print(a)
                  number = number – 2
             else
                  a = a + 1
                  print(a)
                  number = number – 1
             endif
        endwhile
endprocedure
```

State the values printed by the procedure `generate` when `number` = 8.

[1]

16 Kylie buys used games consoles and then sells them to make a profit. She sells her products in multiples of £5 such as £30, £55 and £95. Kylie only accepts £50, £20, £10 and £5 notes from her customers.

Kylie has written an algorithm which will calculate the amount of change needed by stating how many £20, £10 and £5 notes are needed.

The program should output the minimum number of notes required. For example if £35 change is required then it should output 1 x £20 and 1 x £10 and 1 x £5.

```
total = input("Enter total price of goods")
paid = input("Enter amount paid")
global change = paid – total
calculateChange()

procedure calculateChange()
      twenty = 0
      ten = 0
      five = 0
      while change >= 20 //Calculates number of £20 notes needed
         twenty = twenty + 1
         change = change – 20
      endwhile
```

```
14       while change >= 10 //Calculates number of £10 notes needed
15          ten = ten + 1
16          change = change – 10
17       endwhile
18       while change >= 5 //Calculates number of £5 notes needed
19          five = five + 1
20          change = change – 5
21       endwhile
22       print("The amount of change you need is £" + str(change))
23       print("Total £20 Notes:" + str(twenty))
24       print("Total £10 Notes:" + str(ten))
25       print("Total £5 Notes:" + str(five))
26 endprocedure
```

When line 22 is run, it will always print:

```
The amount of change you need is £0
```

Explain why this error occurs when line 22 is run.

[2]

17(a) Ruhail has been told to make use of reusable components when creating his program code.

Explain **two** benefits of using reusable components when writing program code.

1 ____

____

____

____

2 ____

____

____

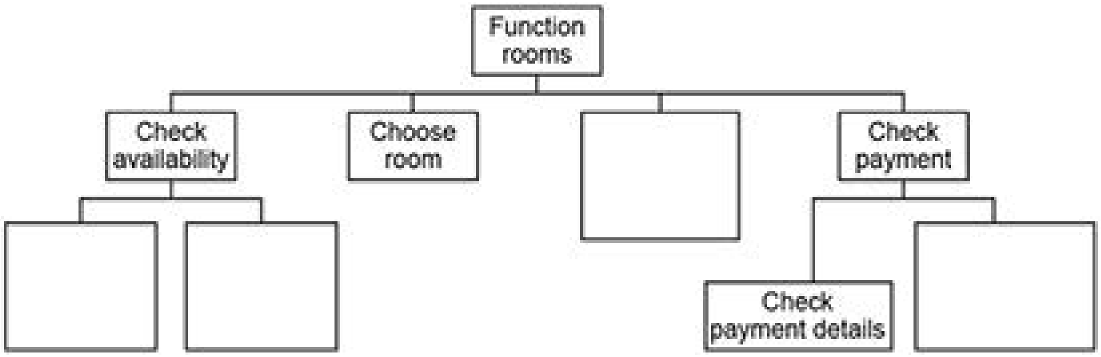
____

[4]

(b) Ruhail owns ten different function rooms which can be hired by different business customers to hold meetings. He would like a program to manage the booking process of each room.

Customers should be able to enter the date they want to hire a function room, and then a list of available rooms will be displayed. Customers can then select which room they want to hire. Customers can then enter their payment details which are then checked and then a confirmation email is sent to the customer.

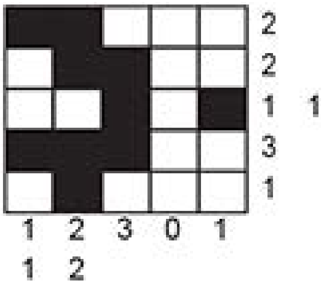
Complete the structure diagram below to show the different component parts of the problem.



[4]

18(a) A Nonogram is a logic puzzle where a player needs to colour in boxes. The puzzle is laid out as a grid and each square needs to be either coloured black or left white.

The numbers at the side of each row and column tells the player how many of the boxes are coloured in consecutively. Where a row has two or more numbers, there must be a white square between the coloured squares.



In this example:

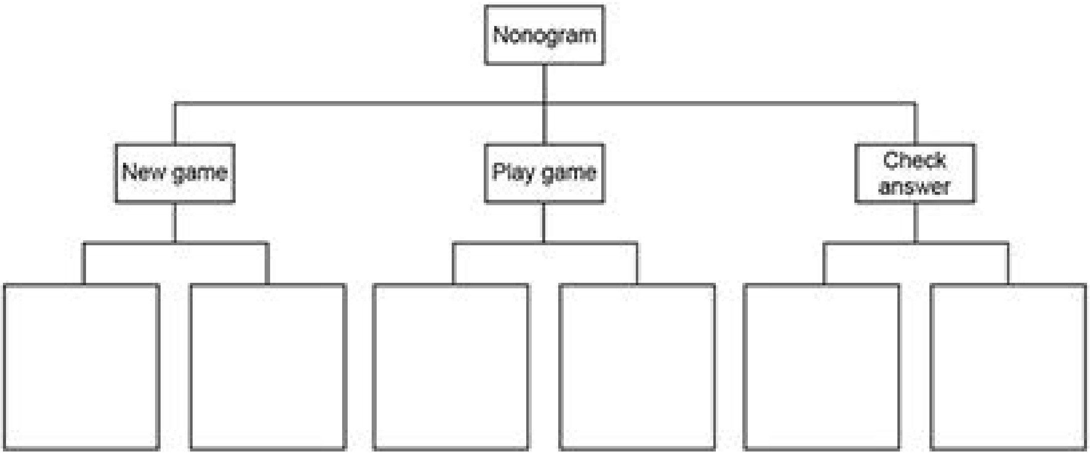
- the first column has 1 1, this means there must be two single coloured boxes in this column. There must be at least 1 white box between them.
- the first row has 2, this means there must be two consecutively coloured boxes in the row.

Juan is creating a program that will store a series of Nonograms for a user to play. The game will randomly select a puzzle and display the blank grid with the numbers for each row and column to the user.

The user plays the game by selecting a box to change its colour. If the box is white it will change to black and if it is black it will change to white. The user can choose to check the answer at any point, and the game will compare the grid to the answers and tell the user if they have got it correct or not.

Juan is creating a structure diagram to design the game.

i. Complete the structure diagram by adding another layer for New game, Play game and Check answer.



[3]

ii. A structure diagram is one method of showing the decomposition of a problem.

Explain why decomposing a problem can help a developer design a solution.

[2]

iii. Identify **one** input, **one** process and **one** output required for the game.

Input

Process

Output

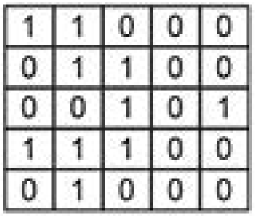
[3]

(b) Juan uses the structure diagram to create a modular program with a number of subroutines.
The program will use two integer 2-dimensional arrays to store the puzzles:

- `puzzle(5,5)` stores the solution
- `answerGrid(5,5)` stores the user's current grid.

A `0` represents a white box and a `1` represents a black box.

i. Juan creates a function, `countRow()`, to count the number of coloured boxes in one row and return the number of consecutive coloured boxes in that row. If there is more than one set of coloured boxes in the row, these are joined together and the string is returned. For example, in the following grid `countRow` for row 0 will return "2" as a string, and `countRow` for row 2 will return "`1 1`" as a string. If there are no 1s in a row, then "`0`" is returned as a string.



1	1	0	0	0
0	1	1	0	0
0	0	1	0	1
1	1	1	0	0
0	1	0	0	0

Complete the pseudocode algorithm `countRow()`.

```
function countRow(puzzle:byref, rowNum:byval)
count = 0
output = " "
for i = 0 to ……………………………………………
  if puzzle[rowNum, i] == …………………………………………… then
```

```
06          count = count + 1
07        elseif count >= 1 then
08          output = output + str(……………………………………) + " "
09          count = 0
10        endif
11      next i
12      if count>= 1 then
13          output=output+str(count)
14      elseif output == "" then
15          output = "……………………………………"
16      endif
17      return ……………………………………
18  endfunction
```

[5]

ii. Explain the purpose of line 03 in the function `countRow`.

____

____

____

____ **[2]**

iii. Describe the purpose of branching and iteration in the function `countRow`.

____

____

____

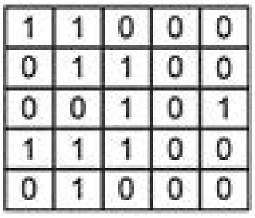
____

____

____ **[3]**

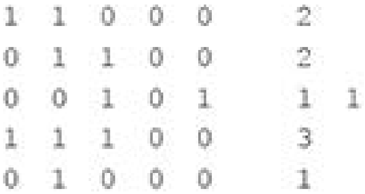
iv. The procedure `displayRowAnswer()` takes `puzzle` as a parameter and outputs the value in each box. Each box in a row is separated by a space. At the end of each row there are two spaces and (by calling the function `countRow` from **part (i)**) the clue values for that row.

For example the puzzle below:



1	1	0	0	0
0	1	1	0	0
0	0	1	0	1
1	1	1	0	0
0	1	0	0	0

Would output:



```
1  1  0  0  0     2
0  1  1  0  0     2
0  0  1  0  1     1  1
1  1  1  0  0     3
0  1  0  0  0     1
```

Write pseudocode or program code for the procedure `displayRowAnswer()`.

[6]

v. The function `checkWon()` takes `answerGrid` and `puzzle` as parameters and compares each element in the grids. If they are identical, it returns `true`, otherwise returns `false`.

```
function checkWon(puzzle)
  for row = 0 to 4
    for column = 0 to 4
      if puzzle[row, column] == answerGrid[row, column] then
        return false
      endif
```

```
07     next column
08    next column
09    return true
10  endfunction
```

There are **three** logic errors in the function `checkWon`

State the line number of each error and give the corrected line.

Error 1 line number

Error 1 correction

Error 2 line number

Error 2 correction

Error 3 line number

Error 3 correction

[3]

(c) * Juan passed the two arrays as parameters, but he did consider making them globally accessible.

Compare the use of global and local variables and data structures in this program. Include the use of parameters and program efficiency in your answer.

[9]

(d) Juan wants to create a program that will generate new Nonograms with different grid sizes. For example a Nonogram with a 10 × 10 grid or a 5 × 20 grid.

Describe how the program could be written to automatically generate a new Nonogram.

[4]

19(a) Hugh has written a recursive function called `thisFunction()` using pseudocode.

```
function thisFunction(theArray, num1, num2, num3)
  result = num1 + ((num2 - num1) DIV 2)
  if num2 < num1 then
    return -1
  else
    if theArray[result] < num3 then
       return thisFunction(theArray, result + 1, num2, num3)
    elseif theArray[result] > num3 then
       return thisFunction(theArray, num1, result - 1, num3)
    else
       return result
    endif
  endif
endfunction
```

The function `DIV` calculates integer division, e.g. 5 DIV 3 = 1

`theArray` has the following data:

Index:	0	1	2	3	4	5	6	7
Data:	5	10	15	20	25	30	35	40

Trace the algorithm, and give the final return value, when it is called with the following statement:

```
thisFunction(theArray, 0, 7, 35)
```

You may choose to use the table below to give your answer.

Function call	num1	num2	num3	result
`thisFunction(theArray,0,7,35)`				

Final return value .... **[5]**

(b) State the name of the standard algorithm `thisFunction()` performs.

[1]

(c) Hugh could have written `thisFunction()` using iteration instead of recursion.

Compare **two** differences between recursion and iteration.

1

2

[4]

(d) The recursive function `thisFunction()` is printed again here for your reference.

```
function thisFunction(theArray, num1, num2, num3)
  result = num1 + ((num2 - num1) DIV 2)
  if num2 < num1 then
    return -1
  else
    if theArray[result] < num3 then
       return thisFunction(theArray, result + 1, num2, num3)
    elseif theArray[result] > num3 then
       return thisFunction(theArray, num1, result - 1, num3)
    else
       return result
    endif
  endif
endfunction
```

Rewrite the function `thisFunction()` so that it uses iteration instead of recursion.

You should write your answer using pseudocode or program code.

[6]

20 The following pseudocode procedure performs an insertion sort on the array parameter.

```
procedure insertionSort(dataArray:byRef)
  for i = 1 to dataArray.Length - 1
    temp = dataArray[i]
    tempPos = i – 1
    exit = false
    while tempPos >= 0 and exit == false
      if dataArray[tempPos] < temp then
        dataArray[tempPos + 1] = dataArray[tempPos]
        tempPos = tempPos - 1
      else
        exit = true
      endif
    endwhile
    dataArray[tempPos + 1] = temp
  next i
endprocedure
```

Explain why `dataArray` is passed by reference and not by value.

[2]

21 Bubble sorts make use of two different loops when sorting items into order.

Describe the **two** loops used and their purpose.

1 ____

____

____

____

2 ____

____

____

____

[4]

22 Barney is writing a program to store data in a linked list. He is writing the initial program for a maximum of 10 data items.

Each node in the linked list has a data value and a pointer (to the next item).

A null pointer is stored with the value `-1`.

* Barney would like his linked list to be part of a base program that is saved in a library. This means that it can be reused and changed by other programs.

Discuss the benefits of using different object-oriented techniques that Barney could use to achieve this.

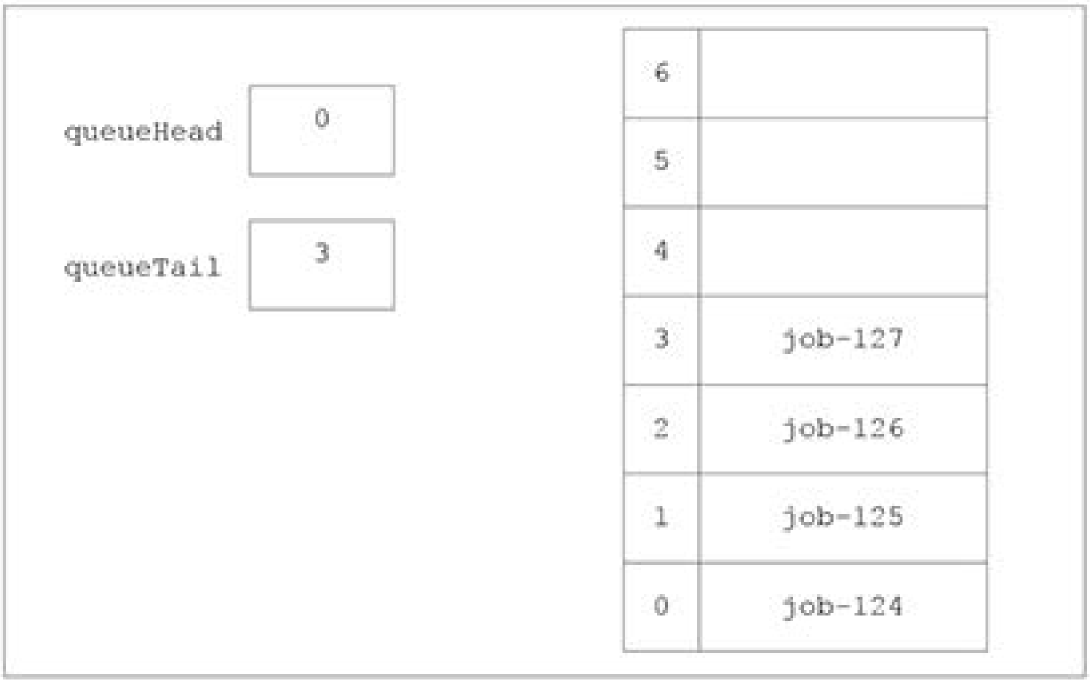
[12]

23(a) A printer buffer is a storage area that holds the data, known as jobs, that are to be printed by a printer.

A simulation of the printer buffer uses a queue data structure to store jobs that are waiting to be printed. The queue is not circular.

The printer buffer is represented as a zero-indexed 1D array with the identifier `buffer`.

Fig. 2 shows the current contents of the queue `buffer` and its pointers.



6	
5	
4	
3	job-127
2	job-126
1	job-125
0	job-124

Fig. 2

State the purpose of the pointers `queueHead` and `queueTail`.

`queueHead` ______________________________

____

`queueTail` ______________________________

____

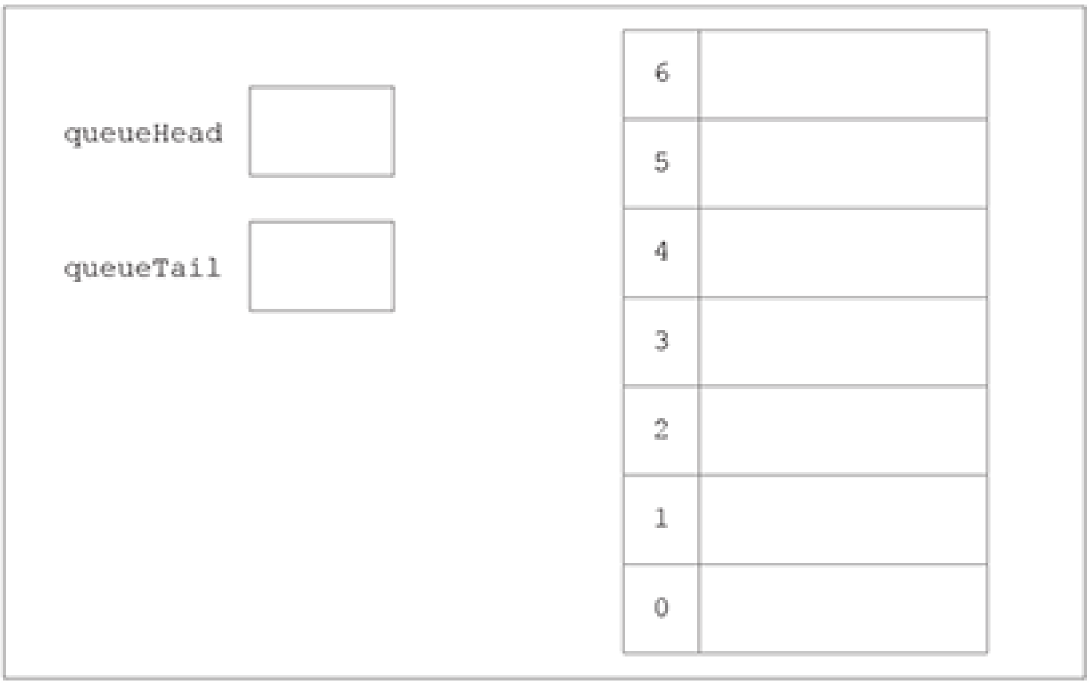
[2]

(b) The function dequeue outputs and removes the next data item in the queue.

The procedure enqueue adds the job passed as a parameter to the queue.

Show the final contents of the queue and pointer values after the following instructions have been run on the queue buffer shown in **Fig. 2**.

```
dequeue()

dequeue()

enqueue(job-128)

dequeue()

enqueue(job-129)
```



6	
5	
4	
3	
2	
1	
0	

[5]

(c) The array, `buffer` and pointer values are declared with global scope.

i. The function `dequeue` returns `null` if the array is empty, and the contents of the next element if not empty. The queue is not circular.

Write an algorithm, using pseudocode or program code, for the function `dequeue()`.

[5]

ii. The function `enqueue` returns -1 if there is no space at the end of the queue to add data, and returns 1 if the parameter was added to `buffer`. The array `buffer` contains a maximum of 100 elements.

Write an algorithm, using pseudocode or program code, for the function `enqueue()`.

[6]

iii. In the main program of the simulation the user is asked whether they want to add an item to the queue or remove an item.

If they choose to add an item they have to input the job name, and the function `enqueue` is called.

If they choose to remove an item, the function `dequeue` is called and the job name is output.

Appropriate messages are output if either action cannot be run because the queue is either empty or full.

Write, using pseudocode or program code, an algorithm for the main program of the simulation.

[8]

(d) The queue is changed to make it a circular queue.

Describe how the functions `enqueue` and `dequeue` will need to be changed to allow `buffer` to work as a circular queue.

[3]

(e) Some print jobs can have different priorities. The higher the priority the sooner the job needs to be printed.

Describe how the program could be changed to deal with different priorities.

---

[3]

---

24 Daisy is a computer technician. She is responsible for making sure all new employees are given a username to access the computer network.

The rules that are followed when creating a new username are as follows:

Step 1: The employee's first name is entered (e.g. Roger)

Step 2: The employee's surname is entered (e.g. Banks)

Step 3: A username is then made up from:

- Their whole surname (e.g. Banks)
- The first letter of their first name (e.g. R)
- A number 1

For example: BanksR1

Step 4: The username is then checked against existing usernames. This is done by calling a function `existingUsers`. This will return `true` if the username is unique and `false` if the username already exists.

Step 5: If the username is unique then "Username is Unique" should be printed. If the username already exists then the number at the end of the username should increase by one (e.g. BanksR2).

Step 6: Steps 4 and 5 should be repeated until the username is unique.

Write a procedure called `createUsername` that meets the rules of Daisy's program.

You should write your procedure using pseudocode or program code.

[9]

25 Sally is a classroom teacher. She would like a program to be able to organise where students will sit in her classroom.

A plan of her classroom is shown **in Fig. 1**.

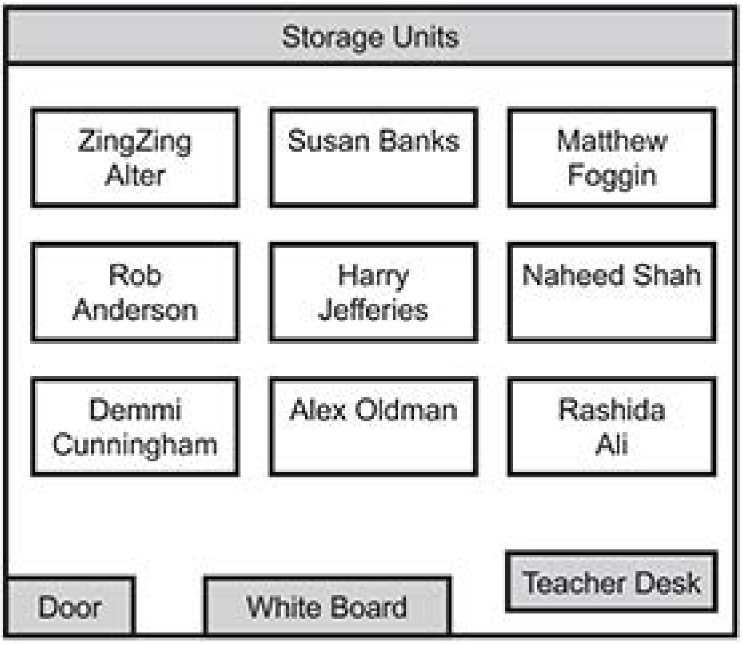


Fig. 1

Sally will make use of an Integrated Development Environment (IDE) to create her program code.

i. Describe **three** features that are commonly found in IDEs that will help Sally write her program code.

1

2

3

[6]

ii. Sally uses a Rapid Application Development (RAD) approach when creating her program.

Describe **two** benefits of using RAD.

1

2

[4]

iii. Sally will make use of an appropriate test strategy to test her programming code.

Compare **one** difference between black box testing and white box testing.

[2]

26 A 1-dimensional array stores the following data:

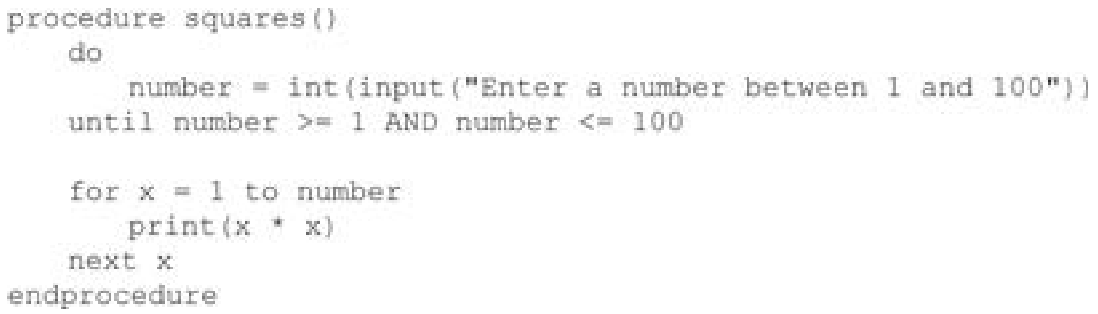
Index	0	1	2	3	4	5
Data	2	18	6	4	12	3

The array needs sorting into descending order.

Describe how a merge sort would sort the given array into descending order.

[6]

27 A procedure takes as input a number between 1 and 100. It calculates and outputs the square of each number starting from 1, to the number input. The square of a number is the result of multiplying a number by itself.



```
procedure squares()
    do
        number = int(input("Enter a number between 1 and 100"))
    until number >= 1 AND number <= 100

    for x = 1 to number
        print(x * x)
    next x
endprocedure
```

The procedure uses one programming construct twice.

State whether the construct that is used twice, is iteration or branching.

____ [1]

28 Dexter is leading a programming team who are creating a computer program that will simulate an accident and emergency room to train hospital staff.

Dexter's team is using an integrated development environment (IDE).

Describe how the programmers could make use of the following IDE tools:

Breakpoints ____

____

____

____

Stepping ____

____

____

____

[4]

29 Explain the advantages of writing an application using a modular approach.

[6]

30 A company employs a systems analyst to create a program for calculating wages. The data that he analyses has the following items:

- Hourly rate.
- Hours worked.
- Tax paid.
- National Insurance paid.

Various variables are used in the execution of the program.

Explain the term 'scope' in relation to variables within a program that calls several different procedures.

--- **[6]**

31 Many functions can be defined using either recursion or iteration.

i. State **one** advantage of using recursion instead of iteration.

ii. State **one** disadvantage of using recursion instead of iteration.

[2]

32(a) A particular programming language uses facts and rules. For one problem in a college, some of the program statements about students and courses are listed.

```
student (ben)                                    {Ben is a student}
student (cindy)
science (computing)                              {Computing is a science}
science (mathematics)
language (english)                               {English is a language}
language (french)

 studies_science (A,B) if student (A) and science (B)
 studies_language (C,D) if student (C) and language (D)
```

The examples you use in your answers to the questions must be based on the program statements listed.

Give the name for this type of programming language.

[1]

(b) Give **one** example of a rule.

-- [1]

(c) Explain the term goal by writing an example.

-- [2]

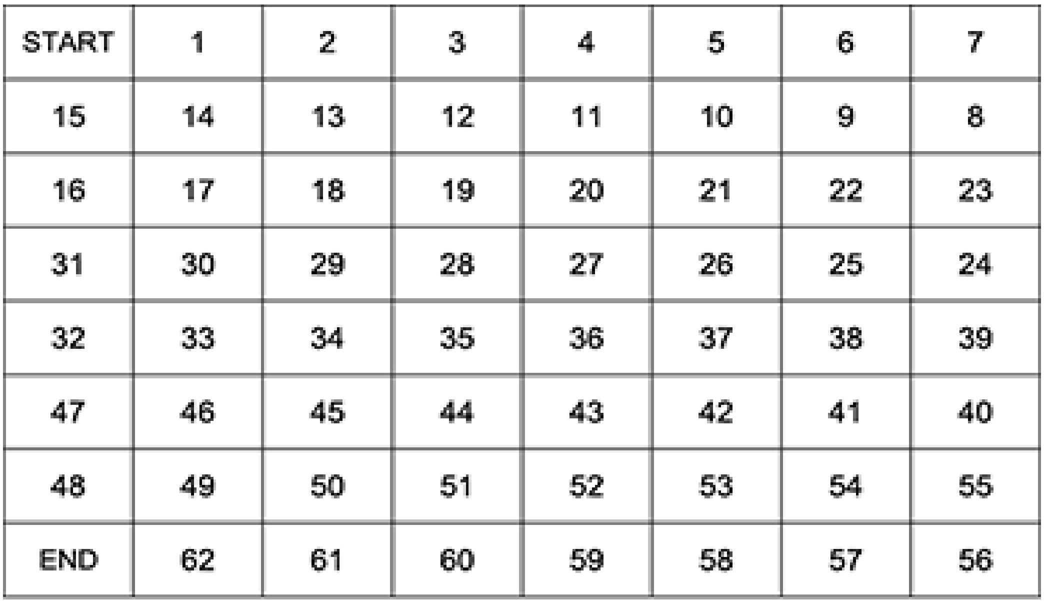
(d) Explain the term instantiation, showing how it is used.

-- [2]

(e) Explain how backtracking is used.

-- [2]

33 The layout for a 2-player board game is shown in Fig 2.1



START	1	2	3	4	5	6	7
15	14	13	12	11	10	9	8
16	17	18	19	20	21	22	23
31	30	29	28	27	26	25	24
32	33	34	35	36	37	38	39
47	46	45	44	43	42	41	40
48	49	50	51	52	53	54	55
END	62	61	60	59	58	57	56

Fig 2.1

The game is played by rolling two 6-sided dice and moving that number of spaces. Both players start on the START space. If a player lands on a space occupied by the other player, they move to the next available space.

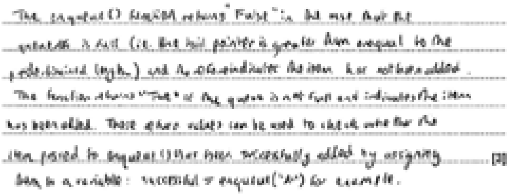
The board is to be stored as a 2-dimensional array.

The programmer is using a number of subroutines in the program. Explain, using an example, the benefits to the programmer of using subroutines in the creation of this game.

[4]

END OF QUESTION PAPER

Question			Answer/Indicative content	Marks	Guidance
1		i	`item`	1	Allow queue can be circular or linear, stack can only be linear <u>Examiner's Comments</u> The vast majority of candidates identified that `item` was the paramete
		ii	`tailPointer`	1	Allow `queue` <u>Examiner's Comments</u> Many candidates identified that `tailPointer` was an example of a global variable in the code.
		iii	1 mark for benefit and 1 mark for expansion: e.g. • Simpler to program ... • ... because values do not need to be passed/renamed/moved between different subroutines • Do not need to worry about returning values // do not need to decide between byval/byref ... • ... all parts of the program can access the (same) value (in the same way) 1 mark for drawback and 1 mark for expansion: • Uses more memory ... • ... because the memory space is declared when the program starts and remains in use throughout • Makes testing / debugging more difficult ... • .... as it's difficult to test an individual block of code • Reduces data accuracy / integrity ... • .... changing a global variable may have an impact on another module	4	2 marks max for benefit 2 marks max for drawback <u>Examiner's Comments</u> The majority of candidates struggled to describe the implications of using global variables, and this highlighted a lack of appreciation of parameter passing and variable scope. At AS Level candidates should become more adept at using parameters and develop an understanding of why this is beneficial – both in theoretical terms and within their personal programming practice. There were few responses that were able to give complete descriptions.

Question			Answer/Indicative content	Marks	Guidance
		iv	1 mark per bullet to max 3 • False is returned if the queue is full ... • ... the main will not attempt to add another item • True is returned if the item is successfully added ... • ... so the main program can try to add another item	3	**Examiner's Comments** Some candidates thought that if `enqueue()` returned `True` it meant that there would be space to add additional values. This was not the case; it meant that the item had been enqueued successfully, but it could have been the last item that then filled the queue to capacity. Sometimes candidates did explain the reasons why `True` and `False` would be returned but were then unable to go on to say how they could be used. Good responses were seen such as 'the main program can give an error message if the operation didn't work if the queue was full'. Exemplar 1 The enqueue() function returns "False" in the case that the queue is full (i.e. the tail pointer is greater than or equal to the predefined length) and to indicate the item has not been added. The function returns "True" if the queue is not full and indicates the item has been added. These return values can be used to check whether the item passed to enqueue() has been successfully added by assigning them to a variable: successful = enqueue("A") for example. [3] This response showed clear understanding of the need to store the result returned by `dequeue()` to be used in the conditional loop test.
		v	1 mark for error **and** correction to max 3 • **Line 01** There is no need to pass in a parameter • **Line 02** has != instead of **==** **Line 04** has elseif and no condition // replace with **else** • **Line 06** should be `headpointer = headpointer` + 1 // Swap lines 06 and 07 • **Line 07** should be `return value`	3	Do not award marks for the line number. The error must be related to the line number stated. **Examiner's Comments** Many candidates identified at least one logical error, but few were able to identify three errors. The most common error identified was the error in line `02.`

Question			Answer/Indicative content	Marks	Guidance

		vi	1 mark per bullet to max 3
			• Use of loop to call dequeue()....
			• ...to output the return value...
			• ...until “EMPTY” is returned
			e.g.
			`elementsLeft = true`
			`while elementsLeft == true`
			`  value = dequeue()`
			`  if not(value = “EMPTY”) then`
			`    print(value)`
			`  else`
			`    elementsLeft = false`
			`  endif`
			`endwhile`
				3	<u>Examiner’s Comments</u>
					Some candidates mistakenly gave a definition for the dequeue() function instead of answering the question set. Candidates need to be mindful to read the question carefully.
					Some candidates used a count-controlled loop whereas a conditional loop was required. Those who made a good attempt at a response often did not assign the result of `dequeue()` to a variable, so could not effectively use the return value from the function as well as print the value returned.

			Total	15	

Question			Answer/Indicative content	Marks	Guidance
2	a		1 mark each to max 3. Max 2 for generic answers with no relation to scenario. e.g. • Has a set/fixed number of values • …and the number of spaces in the road will not change • Stores data of one type • … as the array is only made up of prize objects • Stores data linearly • … match the linear nature of the road • Array contents are mutable • … so prizes can be added/removed from the road • A single indentifier is used to directly index • … any position in the road • Can be iterated by index • … to perform an operation on all road positions	3	<u>**Examiner's Comments**</u> Many responses were too vague, showing little knowledge of the properties of arrays. Relatively few candidates appeared to be able to make explicit links to the scenario to achieve full marks.

	b	i	1 mark each
			• Function/subroutine with identifier `getName` taking no parameters
			• Returning name
			e.g.
			`public function def`
			`getName()       getName(self):`
			`  return name     return`
			`                self.__name`
			`endfunction`
			`public          function`
			`getName(){      getName() {`
			`  return name     return`
			`                this.name`
			`}               }`
				2	BP1 Do not award procedure or method
					BP1 Allow self as an additional parameter if Python is used.
					BP1 If an access modifier is given for the method, it must be public and not private.
					BP2 Do not allow any modified name attribute to be returned.
					<u>**Examiner's Comments**</u>
					While many candidates had little difficulty giving code for a `getter()` there were a number of common errors. Some candidates used a private access modifier when a `getter()` needs to be public. There was often erroneous use of 'procedure' whereas a `getter()` is a function that must return a value. Some candidates tried to set values within the `getter()` function when it should only have returned the class attribute value.

Question			Answer/Indicative content	Marks	Guidance

		ii	1 mark each
			• New instance of `prize` …
			• … with “Box”, “money” and 25 as parameters
			• Assigned to `allPrizes` index 3
			e.g.
			`allPrizes[3] = new prize("Box", "money", 25)`
			`allPrizes[3] = prize.new("Box", "money", 25)`
			`allPrizes[3] = prize("Box", "money", 25)`
				3	MP2 allow any order of parameters
					`"Box"` and `"Money"` must be strings and `25` must be an integer
					Allow `prize.new()` as new is given as the constructor method in the class diagram
					<u>Examiner’s Comments</u>
					Many candidates struggled with the instantiation of an object. Where candidates made an attempt to instantiate some did not use a string for “box” and “money” or did not give 25 as an integer but instead gave the string “25”.

		iii	1 mark for each bullet to maximum 3 e.g. • **Decision** - check whether the space already has a prize allocated … • **Action if true** - another space/number will need to be generated • **Action if false** - the prize will be stored here • **Decision** - check if all 10 prizes have been allocated … • **Action if true** - the algorithm needs to stop generating numbers • **Action if false** - a new number/space needs to be generated and checked	3	Give: • 1 mark for stating a decision • 1 mark for the action required if true • 1 mark for the action required if false <u>Examiner’s Comments</u> There were only two reasonable decisions that could be given from the scenario details. Candidates needed to make it clear that a decision with a Boolean output was present that would dictate two potential outcomes. Some candidates quoted actions such as ‘randomly assign space for prize’ which did not represent a decision. Many responses described the mechanics of setting up the game and the random spaces but did not highlight the program conditions/decisions as required.

Question			Answer/Indicative content	Marks	Guidance

	c	i	1 mark each
			• Constructor header (any suitable name e.g. new, constructor, create, init)
			• …taking **one** parameter only
			• Initialising name to the parameter
			• Initialising money to 5
			• Initialising experience to 0 and roadPosition to 0
			e.g.
			Pseudocode example:
			`public procedure new(pName)`
			`    name = pName`
			`    experience = 0`
			`    roadPosition = 0`
			`    money = 5`
			`endprocedure`
			Python Example:
			`def __init__(self, pName):`
			`    self.__name = pName`
			`    self.__experience = 0`
			`    self.__roadPosition = 0`
			`    self.__money = 5`
			C# Example:
			`public Character(string pName) {`
			`    string name = pName;`
			`    int experience = 0;`
			`    int roadPosition = 0;`
			`    int money = 5;`
			`}`
				5	Allow minor changes to identifiers as long as purpose is clear.
					Allow
					`procedure new(pName)`
					`    this.name = pName`
					…
					(or similar e.g. self.name)
					Allow two parameters if one is *self* and the response is clearly in Python.
					The parameter name should be different to the attribute name.
					Examiner's Comments
					It was clear that those candidates with limited OOP programming knowledge found the writing of a relatively simple constructor method difficult. Those with relevant programming experience often found this to be a very straightforward question. Common errors included passing additional values to set the `experience`, `roadPosition` and `money` attributes rather than setting them to the constant values indicated in the question.

Question			Answer/Indicative content	Marks	Guidance

		ii	1 mark each
			- Procedure/method header …
			- … taking **two** parameters, type (or similar) followed by value (or similar) …
			- … compare type parameter with `"money"`
			- … compare type parameter with `"experience"`
			- … both attributes updated correctly and nothing else modified
			
			e.g.
			`public procedure updateValues(pType, pValue)`
			`    if pType == "money" then`
			`      money = money + pValue`
			`    elseif pType == "experience"`
			`      experience = experience + pValue`
			`    endif`
			`endprocedure`
			
			`def updateValues(self, pType, pValue):`
			`    if pType == "money":`
			`      money += pValue`
			`    elif pType == "experience":`
			`      experience += pValue`
				5	Do not allow Function for BP1
					
					BP2 parameters must be given in the correct order to match the calls to `updateValues()` in the question.
					
					“`money`” and “`experience`” must be string values
					
					Examiner’s Comments
					
					The `updateValues` procedure again proved problematic for candidates with limited OOP experience. No marks were given for the first mark point if a function was declared as there was no return value. Parameter names needed to be fit for purpose, understandable, and had to match the order given in the question scenario to work for the given example calls.

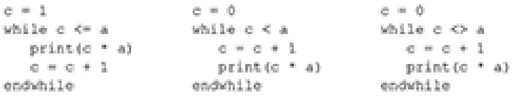
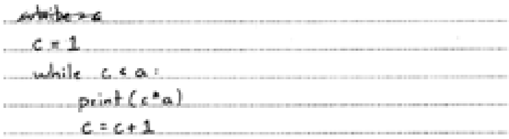
Question			Answer/Indicative content	Marks	Guidance

	d		1 mark for each completed space
			
			`character1 = new` **`Character`**`("Jamal")`
			`newPosition = 0`
			`while newPosition <` **`50`**
			`    move = random(1, 4)`
			`    character1.changePosition(move)`
			`    newPosition = character1.getRoadPosition()`
			`    if newPosition < 50 and road[`**`newPosition`**`] != null then`
			`      prizeType = road[newPosition].getType()`
			`      valueAmount = road[newPosition].getValue()`
			
			`   character1.updateValues(`**`prizeType`**`, valueAmount)`
			`      print("Congratulations you are in position", newPosition, "and found", road[newPosition].getName())`
			`      print("Money", character1.getMoney(), "and experience", character1.`**`getExperience`**`())`
			`    endif`
			`endwhile`
			`print("You reached the end of the road")`
				6	Allow `road.length // len(road)` instead of `50`
					
					Allow <=49 instead of `< 50`
					
					Examiner's Comments
					
					Nearly all candidates achieved some marks, and a majority scored five or six marks.

	e		1 mark each • (Line 02) `for x = 0 to 49` • (Line 03) `print("Space", x)` • (Line 06) `else // elseif road[x] <> null` • (Line 07) `print(road[x].getName())`	4	Line `07` allow `print(road[x].name)` Examiner's Comments Many candidates scored three or four marks but in general candidates found it harder to identify errors in the code than to complete code in the previous question. Some candidates didn't give the line number but rewrote the incorrect line before giving the corrected line, which was acceptable, although not ideal given the scaffolding.

Question			Answer/Indicative content	Marks	Guidance
	f		**Mark Band 3 – High level (7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of global variables and the alternatives; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level (4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of global variables and the alternatives; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level (1-3 marks)** The candidate demonstrates a basic knowledge of global variables and the alternatives with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be*	9	**AO1: Knowledge and Understanding Indicative content** • Global variables are created when the program starts, all subroutines can access/update the contents • Local variables are created in the subroutine they are created in, they are not accessible directly from any other subroutine • Local variables are removed from memory when the subroutine ends. • Local variables can be passed as parameters to a function to be updated, and then returned to override the original local variable • Local variables can be passed by reference to a subroutine to allow the content of the variable to be updated **AO2: Application** • The variables will be stored in memory throughout the whole code execution. However, the amount of data they are storing is relatively low so would not use a lot of memory. • When the game is expanded, the amount of data may increase so it could be memory intensive, especially if graphics are used in the game. • Both arrays are needed throughout the whole game so keeping them as global will make writing the code easier as the programmer will not need to keep passing them as parameters and setting return values. • Only one part of the game is being created initially and therefore the use of global variables would not affect the efficiency greatly. However, when the program expands, it could cause accuracy / testing / debugging and maintenance problems. **AO3: Evaluation** • As this is only a prototype, the use of global variables would be beneficial. • However, when the game expands, the use of global variables could create issues such as running out of memory, coupling, testing & debugging problems and maintenance problems.

Question			Answer/Indicative content	Marks	Guidance
			clear. **0 mark** No attempt to answer the question or response is not worthy of credit.		• The programmer may be best to keep the variables as local and then pass them between the different subroutines as parameters byVal and byRef. <u>Examiner's Comments</u> Most responses were Level 2 for definitions and some expansion to passing parameters. Very few candidates were able to go into depth about alternatives to global variables such as passing by value and passing by reference in detail or extending to issues such as scalability within a larger more extended game. Few candidates picked up on the fact that this was a more limited prototype that was likely to be expanded on which would require more consideration to be given to variable scope.
			Total	40	

Question			Answer/Indicative content	Marks	Guidance
3	a		1 mark each • Initialisation of counter variable before loop • Condition (e.g. while c <= a) • c incremented in loop • Completed loop will produce correct results 12, 24 … 144 e.g.  c = 1 while c <= a print(c * a) c = c + 1 endwhile c = 0 while c < a c = c + 1 print(c * a) endwhile c = 0 while c <> a c = c + 1 print(c * a) endwhile	4	BP2 – Allow any suitable logic for the while loop condition that iterates between 1 and a. Allow != for <> Allow += or equivalent for c = c + 1 Allow hard coded values for upper bound such as a = 12 or a = 13 depending on the relational operator used. No marks awarded if a conditional loop has not been used. Max 3 if solution does not completely work. **Examiner's Comments** This question was generally well attempted with marks relatively evenly distributed throughout the range of marks available. There were some off-by-one errors, with the loop counter or the position of the output line being incorrectly positioned before/after the counter increment. Incorrect loops such as count-controlled for loops were rejected. If responses did not produce a fully correct output, marks given were limited to 3 marks. Exemplar 1  ~~while c>=a~~ c = 1 while c < a : print(c*a) c = c+1 This response showed a typical off-by-one error where the candidate had not thought through the logic of the entire response. The condition operator should have been <= rather than < to iterate through all values of c from 1 to 12.

Question			Answer/Indicative content	Marks	Guidance
	b		1 mark for do loop and 1 mark for while loop up to a maximum of 2 marks. • While loop will check the condition at the **start** of the loop // pre-condition loop • Do loop will check the condition at the **end** of the loop // post-condition • The code in a while loop may never run (if the condition is already met) • The code in a do loop will always run at least one.	2	Answer must cover both do loop and while loop for 2 marks to be awarded BP1 and BP2 must be specific as to the location that the condition is placed Examiner's Comments Many responses were vague and did not accurately identify or distinguish that a while loop is a pre-condition loop while a do loop is a post-condition loop. Responses had to be clear as to the relative position of the condition in the loop. Accepted responses included a while loop is a pre-condition loop while a do loop is a post-condition loop. Fewer candidates identified that the body of a while loop may not be executed while the body of a do loop will always be executed at least once.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
4			**Mark Band 3 – High level** **(7–9 marks)** The candidate demonstrates a thorough knowledge and understanding of parameters and local/global variables; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4–6 marks)** The candidate demonstrates reasonable knowledge and understanding of parameters and local/global variables; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1–3 marks)** The candidate demonstrates a basic knowledge of parameters and local/global variables with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated*	9	**AO1: Knowledge and Understanding Indicative content** • Local variable can only be accessed within sub-program/main program it is declared within • Global variable can be accessed by all sub-programs • Parameters are items passed to a subproblem • Passing by reference sends a pointer to the original value, so this will be changed when control is returned • Passing by value sends the a copy of the value, so the original will not be changed when control is returned **AO2: Application** • If board is local it can only be accessed in the main program • This will need to be passed to any sub-programs that need to use it • If the board needs to be changed it will need passing by reference, so that the board is updated • If it only needs to be accessed and not changed it can be passed by value **AO3: Evaluation** • If global then this would be present in memory throughout hence using more memory • …however the board will be required throughout the program so may be as efficient as passing it through parameters • …if global then the programming may be more straight forward, and less likely to have errors with passing the board incorrectly to subprograms, i.e. it may not be updated when it needs to be • Using local means that the board can be manipulated by subprograms without affecting the actual board if needed, for example to simulate potential changes. Examiner's Comments A good range of Level 2 responses were observed with competent descriptive definitions of local versus global and

Question				Answer/Indicative content	Marks	Guidance

				in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.
				0 mark
				No attempt to answer the question or response is not worthy of credit.
						byVal/byRef parameter passing.
						There were far fewer high level AO3 evaluations of relative memory use within the context of the scenario when passing byVal/byRef for a relatively large grid object. Good, contextualised responses took the elements of the scenario in section B of the paper and talked about the grid object and passing parameters like x and y coordinates. Responses such as this were relatively rare.
						Assessment for learning
						Candidates would benefit from producing solutions centred around the scenario presented in Section B of the paper. Having extensive OOP programming experience in a relevant high level language will help candidates to successfully tackle questions where algorithms have to be presented. Sample Python code based on Section B is presented below for consideration.
						Assessment for learning
						# 2023 Section B
						class Treasure:
						    def __init__(self, pValue: int, pLevel: str):
						        self.__value = pValue
						        self.__level = pLevel
						    def getValue(self):
						        return self.__value
						    def setValue(self,pValue: int):
						        return self.__pValue
						    def getLevel(self):
						        return self.__level
						    def setLevel(self,pLevel: str):
						        return self.__pLevel
						class Board():
						    def __init__(self):
						        self.__grid = [[Treasure(-1,"") for _

Question			Answer/Indicative content	Marks	Guidance

					in range(20)] for _ in range(10)]
					
					    def getGridItem(self,x: int, y: int):
					        return self.__grid[x][y]
					
					    def setGridItem(self,x: int,y: int, pTreasure: Treasure):
					        self.__grid[x][y] = pTreasure
					def guessGrid(b: Board):
					    row = int(input("Row: "))
					    col = int(input("Col: "))
					    t = b.getGridItem(row, col)
					    if t.getLevel() == "":
					        print("No treasure")
					    else:
					        print(f"Treasure found at {row} {col}!")
					        print(f"Level {t.getLevel()} Value {t.getValue()}")
					
					# Test code - Treasure @ 2,2 all other locations no treasure
					island = Board()
					prize = Treasure(5,"Prize!")
					island.setGridItem(2,2,prize)
					guessGrid(island)

			Total	9	

Question			Answer/Indicative content	Marks	Guidance
5			1 mark each 03 - Loop through each of the **characters/digits** in the `number` string (parameter) 04 - Find the ASCII value of the current **character/digit** 09 - Return true if the value is an integer and false otherwise	3	<u>Examiner's Comments</u> This question required the *purpose* of the lines of code to be described, but many candidates just described the functionally of the lines of code such as 'line 03 is a counter controlled loop from 0 to number.length -1'. The expected purpose of this line of code was to set up a loop to iterate through each character in the input string parameter. While many candidates could describe the purpose of at least one of the lines of code given, few could clearly describe the purpose of all three lines.
			Total	3	
6			1 mark each to max 3 - The function calls itself…. - …..such as line `05` / `07` - Each recursive call will create a new copy of the values in the function…. - ….and add all of the values of the copy the call is being made from to a stack - There is a base case // condition that stops the recursive calls… - …condition in line `02` - There may be more than one base case	3	Allow answers in context as long as they are clear what the features are. <u>Examiner's Comments</u> Weaker responses were limited to one or two points for 'calls itself' with example such as 'line 05'. Fewer candidates went on to identify the requirement for a base case. A variety of terminology was used for the base case such as terminating case and stopping case. If candidates were clearly referring to the case where the recursive calls stopped, and the recursion started to unwind, the mark was given.
			Total	3	

Question			Answer/Indicative content	Marks	Guidance
7		i	it can only be accessed within the subroutine//block in which it is declared	1	Examiner's Comments Few candidates were able to clearly define the term 'local variable'. The concept of scope of variable s appeared to be poorly understood, with few able to define that a local variable's scope was that of the function/procedure in which it was declared.
		ii	1 mark for benefit e.g. • Increases data integrity • More efficient memory usage • Stops other subroutines accidently altering variable 1 mark for drawback e.g. • Cannot be accessed directly by other subroutines • It has to be passed into a subroutine as a parameter	2	Examiner's Comments Some candidates confused the terms local variable and global variable and gave de finitions the wrong way round. A significant number of responses demonstrated conceptual misconceptions. The contents of dataArray could be used in other parts of the program if it was passed as a parameter to a function/procedure, but could not be referenced directly. Responses such as giving 'can't be used anywhere else in the program' as a disadvantage were, therefore, incorrect.
			Total	**3**	

Question			Answer/Indicative content	Marks	Guidance
8			**Mark Band 3–High Level** **(7–9 marks)** The candidate demonstrates thorough knowledge and understanding of IDEs; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate provides a thorough discussion which is well-balanced. Evaluative comments are consistently relevant and well-considered. There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated. **Mark Band 2-Mid Level** **(4–6 marks)** The candidate demonstrates reasonable knowledge and understanding of IDEs; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are for the most part appropriate, although one or two opportunities for development are missed. There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence. **Mark Band 1-Low Level** **(1–3 marks)** The candidate demonstrates a basic knowledge of IDEs, with limited understanding shown; the material is basic and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgments if made are weak and unsubstantiated. The information is basic and	9	**AO1: Knowledge and Understanding** e.g. IDE: • pretty print / syntax highlighting • auto-complete • auto-correction • breakpoints • stepping Editor: • no helpful writing/debugging features • no excess features/interface **AO2.1: Application** e.g. IDE • identify syntax errors as writing • …saves trying to find them • easier debugging as can step through a program • auto-indenting avoids errors from incorrect indentation • May have built in unit testing to automate testing and avoid new errors being introduced. Editor • does not offer suggestions on code corrections • Has a lower footprint on memory/CPU which may be suited to quick alterations or working on lowed spec'd systems. • May be better when learning to program as it forces the user to type everything in full / doesn't give suggestions, helping things stick in memory. **AO3.3: Evaluation** e.g. • IDE is helpful in reducing original errors • IDE is helpful in finding and correcting errors • Editor is a simpler system to use e.g. less memory needed to run it, does not try and auto-correct incorrectly or introduce errors that the programmer has not made.

Question			Answer/Indicative content	Marks	Guidance
			communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear. **0 mark** No attempt to answer the question or response is not worthy of credit.		Examiner's Comments A pleasing number of candidates could identify and describe a range of features of an IDE, with most being able to demonstrate sufficient knowledge to achieve mark band Level 2. Few candidates displayed a level of evaluative reasoning of sufficient depth to achieve mark band Level 3. More successful responses clearly explained how relevant debugging tools in an IDE led to much greater productivity than a text editor because of the way that they could assist with finding logical errors in code through the use of stepping and tracing. An acknowledgement of the amount of system resources and the complexity of some IDE environments was less often evaluated. Candidates need to focus on making clear logical arguments with evidence of evaluation (AO3) to be able to achieve mark band 3.
			Total	9	

Question			Answer/Indicative content	Marks	Guidance

9			1 mark per bullet to max 4
			• taking value as input
			• looping until valid between 1 and 255
			• calling function with correct parameter
			• outputting return value
			
			`denary = -1`
			`while denary < 1 or denary > 255`
			`   denary = input("Enter denary value between 1 and 255")`
			`endwhile`
			`print(toBinary(denary))`
				4	Allow other checks for a valid number. For example
					
					`denary.isInteger == False`
					
					Examiner's Comments
					
					Candidates found Question 4 (b) easier to approach than Question 4 (a). Most could write pseudocode to accept a user input. When validating the input to be a value between 1 and 255, there was incorrect use of relational operators with off -by-one errors on occasion. There was also incorrect use of logical operators where or was used instead of *and*, and vice-versa. When candidates called `toBinary(inputVal)`, the result was not always stored for later use.
					
					Assessment for learning
					
					When preparing candidates for the examination, they will benefit from a wide range of programming experience.
					
					Questions such as 4 (a) and 4 (b) present an ideal opportunity for developing coded solutions to test and discuss before looking at how the algorithms could be presented as pseudocode.

			Total	4	

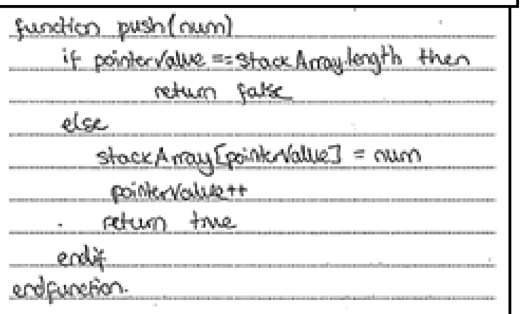
Question			Answer/Indicative content	Marks	Guidance
10			1 mark per bullet • identifier `cards`… • …with 2 dimensions	2	**Examiner's Comments** Many candidates gained a mark by initialising the identifier `cards`, but fewer gained the second mark for correctly setting it to be a two-dimensional structure. Many obscure forms of syntax were observed, but marks was given if it was clear that the structure was two-dimensional, however, for many responses, it was clear that a one-dimensional list had been initialised.
			Total	2	
11		i	Line number 5	1	**Examiner's Comments** Nearly all candidates gave the correct answer line 5.
		ii	1 mark per feature • A function that calls itself // a function that is defined in terms of itself • …has a base case (that terminates the recursion)	2	**Examiner's Comments** Most candidates knew that a recursive algorithm is self-referential and calls itself, but some candidates were too vague specifying that it 'calls a function'. Candidates often found it harder to gain the second mark and some gave answers not related to the question such as explanations of how recursion uses stack frames during execution. Technical vocabulary is important, and some candidates did not make it clear that recursion has a base case. Those that stated a stopping/terminating condition needed to qualify their response to say that these conditions stopped/halted the recursion. Where candidates just wrote 'stopping condition' it was too vague as it was unqualified since they could have been talking about any conditional loop.
			Total	3	

Question			Answer/Indicative content	Marks	Guidance

12		i	1 mark per correctly completed statement
			
			e.g.
			pubic function pop()
			  if pointerValue == **0** then
			    return **-1**
			  else
			    pointerValue = pointerValue **-1**
			    returnValue = stackArray[**pointerValue**]
			    return **returnValue**
			  endif
			endfunction
				5	<u>Examiner's Comments</u>
					
					Most candidates gained some marks with many gaining full marks. The most common errors were writing identifiers for pointerValue and returnValue with spaces in, returning values as strings, or returning True/False instead of -1 as required in the question.

		ii	1 mark per bullet to max 6 • function header • ..taking parameter (ignore byval/byref) • checking if stack is full (pointerValue at 100)… • …and returning false • (otherwise) adding value to top of stack • …incrementing top of stack pointer • return true e.g. function push(value) if pointerValue < 100 then stackArray[pointerValue] = value pointerValue = pointerValue + 1 return true else return false endif endfunction	6	Ignore additional parameters in function definition Do not accept the return of string values FT following a reasonable attempt to check if the stack is full <u>Examiner's Comments</u> Candidates produced a higher standard of pseudocode this session and many scored most if not full marks for a standard stack push routine. Common errors included omitting a parameter and/or getting a user input as the value to place into the stack, returning strings or printing the return values, and off-by-one errors when testing to see if the stackPointer was at the top of the stack to determine if the stack was full. It was noticeable that a number of students who were only familiar with Python gave list append type solutions rather than using the array and pointers as per the implementation given. Another common error was incrementing the value of the stackPointer before the parameter was assigned to stackArray at the stackPointer index, which was frequently seen when candidates did not know that the stackPointer actually pointed to the index of the next free space in the stack. **Exemplar 3**

Question			Answer/Indicative content	Marks	Guidance

					
					function push(num)
					if pointerValue ==stackArray.length then
					return false
					else
					stackArray[pointerValue] = num
					pointerValue++
					return true
					endif
					endfunction.
					
					Candidates are encouraged to present pseudocode solutions with clear indentation to aid readability. No specific language/syntax is expected, but the logic of the solution must be clear.

		iii	1 mark per bullet • instantiation of new object of type stack • assigned to variable mathsStack `mathsStack = new stack()`	2	Accept `mathsStack = stack()` allow missing brackets this time only e.g. mathsStack = stack Examiner's Comments Many candidates struggled to answer this question because they lacked practical experience of OOP that showed a lack of familiarity in terms of creating instances of a class. Some candidates tried to declare `mathsStack` as a procedure or class, and many got it the wrong way round and actually declared a new identifier called `stack` as an instance of the class `mathStack`.

		iv	1 mark for each completed statement
			
			`returnValue = true`
			`while returnValue == `**`true`**
			`returnValue = mathsStack.`**`push`**`(input("Enter Number"))`
			`if returnValue == `**`false`**` then`
			`print``("Stack full")`
			`endif`
				4	Accept equivalent for print e.g. output
					
					Examiner's Comments
					
					The majority of candidates scored some marks but relatively few gained full marks. Return was seen instead of print/output when the question did, on this occasion ask for the message to be output (as the code was specified as being in the main program rather than a subroutine).

Question			Answer/Indicative content	Marks	Guidance

		v	mark per bullet to max 8
			• initialise a total to 0 outside of loop
			• looping
			• removing an item from the stack using the method pop
			• check if stack is empty
			• (if not) add value returned to total
			• …outputting total
			• counting how many values are returned
			• stopping loop when either 20 items removed or no items left
			
			`total = 0`
			`quantity = 0`
			`returnValue = 0`
			
			`while quantity<20 and retunValue!=-1`
			`   returnValue = mathsStack.pop()`
			`   if(returnValue != -1) then`
			`     quantity = quantity + 1`
			`     total = total + returnValue`
			`     print(total)`
			`   endif`
			`endwhile`
				8	Examiner's Comments
					
					The standard of pseudocode / programming code was better than in previous sessions and most candidates made a reasonable attempt to pop 20 values from the stack. Many candidates found it difficult to reference the class methods correctly or made assumptions about the existence of other methods that were not provided within the scenario (e.g. `.full(), .empty(), .length(), .remove()`) that should not be presumed to exist.
					
					Those who had little understanding of encapsulation often tried to access class attributes such as `stackPointer` directly to manipulate `mathsStack`, rather than using methods to interact with the instance of the stack.
					
					When candidates did use the `.pop()` method to retrieve a value from `mathsStack` they frequently did not store the result for later use to check whether then stack was empty.

			Total	25	

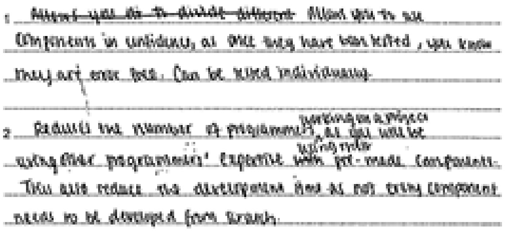
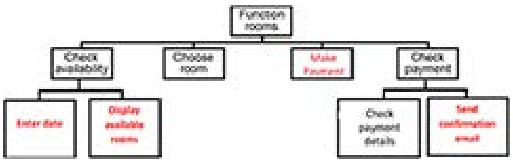
Question			Answer/Indicative content	Marks	Guidance
13			1 mark per bullet e.g. • Store data that has been used in cache/RAM in case needed again • e.g. store design of the weather/a cloud/external environment	2	Allow 2 valid examples for 2 marks **Examiner's Comments** Responses to caching were better than those seen in recent papers and many candidates managed to either describe the concept of caching or give an example of something that would realistically be cached. Far fewer managed to do both in detail.
			Total	2	

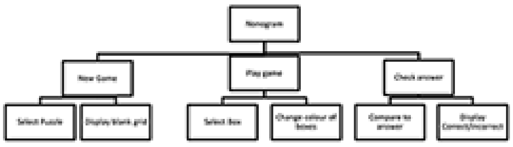
Question			Answer/Indicative content	Marks	Guidance
14	a	i	1 mark for each variable • `contents` • `count` • `numberOfWords` • `words / words[]`	2	Accept exact spelling only Do not award numberOfWords if there are obvious spaces in 'number of Words'. It must be a valid identifier. <u>Examiner's Comments</u> The majority of candidates answered correctly, with the most popular answers being `count` and `contents`. A few candidates incorrectly gave data values from the array rather than identifying variables in the function.
		ii	1 mark per bullet • By reference the function receives the memory location of the data • By value the function receives a copy of the variable • By reference will make changes to the original variable • By value will make changes to the copy of the variable • By reference will overwrite data in the original variable • By value will not overwrite the data in the original variable • By reference will keep the changes after the function ends • By value will not keep the changes after the function ends	2	Must cover byVal and byRef for 2 marks to be awarded. Must be clear that byVal <u>is a copy</u> of the original value. <u>Examiner's Comments</u> It was pleasing to see an improvement in responses to this topic this session. There were still some answers that were too vague that did not specify that by *value* uses a <u>copy</u> of the parameter and that by *reference* passes the <u>memory address</u>. Other examples of vagueness that were not given marks included answers such as 'by value can't change the value while by reference can' that did not qualify the scope within which changes can be made. A variable passed by value can clearly be changed in a function, but it is the local copy that is changed and then disregarded when the function finishes.

Question			Answer/Indicative content	Marks	Guidance

		iii	1 mark per bullet
			
			• initialising a counter
			• looping between 0 and numberOfWords -1
			• incrementing counter inside loop
			• remainder of algorithm correct (initialisation, concatenation and return)
			
			e.g.
			`contents = ""`
			`count = 0`
			`while count < numberOfWords`
			`   contents = contents + words[count] + " "`
			`   count = count + 1`
			`endwhile`
			`return contents`
				4	Accept:
					while count <= numberOfWords - 1
					
					Accept other combinations for example counting from 1 and then subtracting 1 for the array element (but do not credit off by one errors)
					
					Accept:
					len(words) for numberOfWords
					
					Examiner's Comments
					
					Many candidates struggled with this question. Some candidates rewrote the `for` loop putting the word *while* in place of *for* showing little understanding of the difference between a counter-controlled and a conditional loop.
					
					Common errors included not initialising the `count` variable before using it within the body of the while loop, off-by-one errors, and forgetting to increment the `count` variable within the loop. Poor indentation was often a problem, and a number of candidates erroneously placed the return statement inside the body of the loop.
					
					Off-by-One errors
					There were many off-by-one errors observed, e.g. `while count <= numberOfWords` rather than `while count < numberOfWords`.
					
					Candidates need to give code that is logically accurate, and, in this instance, it required the loop to run the correct number of times so that all the words in the array were processed.

Question			Answer/Indicative content	Marks	Guidance
	b		1 mark for benefit, 1 mark for drawback e.g. Benefits: • Variable doesn't need passing as a parameter (byref) • You don't need to return a value • Can be accessed from any function / anywhere in the program Drawback: • Increases memory usage (as it is used until full program execution is over) • Alterations within the function may have unwanted side effects elsewhere in the program.	2	<u>Examiner's Comments</u> Many candidates found it easier to describe a benefit than to give a drawback. The most commonly identified benefit was that the array would have global scope (and would therefore not need to be passed as a parameter), but often the descriptions given were too vague, e.g. 'can be accessed anywhere'. The correct technical vocabulary is required. Drawbacks were poorly described. Potential side effects and resultant complexity debugging were frequently alluded to as 'accidental change' but not fully developed into complete qualified points. There was also a frequent misconception that you cannot have multiple variables with the same name, which is not true. When a local variable is declared with the same name as a global variable that already exists, it takes precedence within the local scope.
			Total	10	
15			• 2,4,6,8	1 (AO3.3) (1)	<u>Examiner's Comments</u> In many cases candidates gave the response 2 and did not iterate through the loop until the initial loop condition was false.
			Total	1	
16			1 mark per bullet up to a maximum of 2 marks, e.g.: • the variable `change` is global (set on line 03) • the value is printed after it has been changed to 0 by the procedure	2 (AO3.3) (2)	<u>Examiner's Comments</u> Many candidates identified that the procedure would reduce the `change` to 0 but few identified that it was because `change` was a global variable which meant that the initial value was being decreased to zero or that line `22` would need to be moved to before line `10` to resolve the error.
			Total	2	

Question			Answer/Indicative content	Marks	Guidance
17	a		1 mark per bullet up to a maximum of 2 marks for each advantage (4 marks in total), e.g.: • Saves times (1) as code does not need to be rewritten (1) • Code may already be tested (1) which will therefore save development//debugging time (1) • More efficient code (1) which will use less memory//be easier to maintain (1) • May require less technical knowledge (1) as code can used rather than written (1)	4 (AO1.1) (2) (AO1.2) (2)	Award a maximum of two marks for each advantage. Allow other suitable examples. **Examiner's Comments** Many candidates did not give a clear explanation for a point made, and there were a number of 'cheaper' or 'quicker' type responses that had no qualification for what was being alluded to as 'cheaper' or 'quicker'. **Exemplar 1**  1. ~~Allows you to divide different~~ Allows you to use components in confidence, as once they have been tested, you know they are error free. Can be used individually. 2. Reduces the number of programmers working on a project, as you will be using other programmers' expertise with using their pre-made components. This also reduces the development time as not every component needs to be developed from scratch. A response that shows a clear point being made with a detailed explanatory expansion for both benefits.
	b		1 mark per box up to a maximum of 4 marks:  Function rooms Check availability Choose room Make Payment Check payment Enter date Display available rooms Check payment details Send confirmation email	4 (AO2.2) (4)	**Examiner's Comments** Candidates needed to apply computational thinking skills to identify the components of the problem presented in the stem of the question and to determine the order of the steps required. Many candidates responded in context and generally answered well, but a significant number either repeated the points they made or repeated steps already given in the diagram.
			Total	8	

Question			Answer/Indicative content	Marks	Guidance
18	a	i	1 mark per box up to a maximum of 3 marks. • **Select puzzle** and **display blank grid** (below new game) • **Select box** and **change colour of boxes** (below play game) • **Compare to answer** and **display correct/incorrect** (below check answer) e.g.  Nonogram New Game Play game Check answer Select Puzzle Display blank grid Select box Change colour of boxes Compare to answer Display Correct/incorrect	12 AO1.1 (2) AO1.2 (2) AO2.1 (23 AO3.3 (5)	
		ii	1 mark per bullet up to a maximum of 2 marks, e.g: e.g. • Splits the problem into smaller chunks • Smaller problems are more manageable • Smaller problems are easier to solve • To see where code can be reused in the solution • To split tasks between different programmers	2 AO1.1 (1) AO1.2 (1)	
		iii	1 mark for input, 1 for process 1 for output e.g. Input: • Clicking a box Process: • Generating new puzzle • Checking if block is black • Changing block to white Output: • Grid with coloured squares	3 AO2.2 (3)	

Question			Answer/Indicative content	Marks	Guidance

	b	i	1 mark for each correctly completed statement up to a maximum of 5 marks:
			function countRow(puzzle:byref, rowNum:byval)
			  count = 0
			  output = " "
			  for i = 0 To **4**
			    if puzzle[rowNum, i] == **1** then
			      count = count + 1
			    elseif count >= 1 then
			      output = output + str(**count**) + " "
			      count = 0
			    endif
			  next i
			  if count>= 1 then
			      output=ouput+str(count)
			  elseif output == "" then
			      output = **"0"**
			  endif
			  return **output**
			endfunction
				5 AO2.2
				(2)
				AO3.2
				(3)
					Accept
					for i = 0 to row.length-1
					for i = 0 to row.length
					for i=0 to 5

		ii	1 mark per bullet up to a maximum of 2 marks, e.g: • Initialise the variable output… • …with a space • …for use later on in the code… • …So it can be used for concatenation later in the code … • …to avoid an error being generated	2 AO1.2 (1) AO2.2 (1)	
		iii	1 mark per bullet up to a maximum of 3 marks, e.g: • check the value stored in each index • check whether it is at the end of a row • check whether each row has been given an output or not	3 AO2.2 (3)	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td></td><td></td><td>iv</td><td>1 mark per bullet up to a maximum of 6 marks:
• Procedure heading for displayRowAnswer
• …taking puzzle as parameters
• Nested loops through all array elements
• …outputting all rows
• … at the end of each row calling countRow ….
• …..with parameters puzzle and the current loop counter

e.g.
<pre>procedure
displayRowAnswer(puzzle)
   for i = 0 To 4
     for j = 0 To 4
       print(puzzle[i, j] + " ")
     next j
     print (" " + countRow
     (puzzle, i))
   next i
endprocedure</pre></td><td>6 AO2.2 (3)
AO3.2 (3)</td><td>Accept

for i = 0 to <b>row.length-1</b>

for i = 0 to <b>row.length</b>

for i=0 to <b>5</b></td></tr>
<tr><td></td><td></td><td>v</td><td>1 mark for clearly identifying each error and giving the correction.
• Line 01 needs <code>answerGrid</code> as parameter
• Line 04 == should be !=
• Line 08 should be <code>next row</code></td><td>3 AO2.1 (3)</td><td>Do not award marks for line numbers alone without stating the error.

Consider 1 mark for not changing line 04 but changing 05 to true and 09 to False</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
	c		**Mark Band 3 – High level (7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of local and global variables; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate is able to weigh up the use of both local and global variables which results in a supported and realistic judgment as to whether it is possible to use them in this context. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level (4-6 marks)** The candidate demonstrates reasonable knoledge and understanding of local and global variables; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion showing some recognition of influencing factors that would determine whether it is possible to use local and global variables in this context. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence* **Mark Band 1 – Low Level (1-3 marks)** The candidate demonstrates a basic knowledge of local and global variables with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than an unsupported assertion. *The information is basic and comunicated in an unstructured way. The information is*	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	**AO1: Knowledge and Understanding Indicative content** Local variables: • Scope within the module defined within • Cannot access externally unless passed as parameter, or returned from function • When module is exited, memory of variable is freed Global variables: • Scope within the entire program • Can access from anywhere • Retained in memory permanently ByRef Points to location of variable ByVal Sends the value **AO2: Application** • If global the arrays can be accessed from all modules by direct reference • If local to the main, the arrays will need to be passed as parameters byreference • Can send ByVal – but not always possible with arrays in some languages • Modules are self contained and then can be reused in other programs he wants to create without needing to take the global variables with them **AO3: Evaluation** e.g. • +ve Local = memory efficient • +ve Global = easier programming, simpler to follow, easier to debug • -ve Global = memory inefficient, not good programming technique • -ve Local = more difficult to trace/debug/follow where the values are passed • Relatively small program – don't know about overall plan for it, it might not be memory intensive, unlikely anyone else is going to access/amend e.g. use as a library – therefore global would not waste significant resources

Question			Answer/Indicative content	Marks	Guidance
			supported by limited evidence and the relationship to the evidence may not be clear. **0 marks** No attempt to answer the question or response is not worthy of credit.		
	d		1 mark per bullet to max 4 e.g. • Make use of random numbers • Generate an x/horizontal size for the grid • Generate a y/vertical size for the grid • Loop through each row/column • …generate a number between 0 and the number of rows/columns (depending on MP4 answer) • Loop through each box • …generate a 1 or 0 to store in it	4 AO2.1 (2) AO2.2 (2)	
			Total	40	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>19</td><td>a</td><td></td><td>1 mark per bullet

- Calculation of result to 3
- Call with thisFunction(theArray, num1=4, num2=7, num3=35)
- Result = 5
- call with thisFunction(theArray,num1=6,num2=7,num3=35)
- (Result = 6) return of value 6

<table>
<tr><th>Function call</th><th>num1</th><th>num2</th><th>num3</th><th>result</th></tr>
<tr><td>thisFunction(theArray,0,7,35)</td><td>0</td><td>7</td><td>35</td><td>3</td></tr>
<tr><td>thisFunction(theArray,4,7,35)</td><td>4</td><td>7</td><td>35</td><td>5</td></tr>
<tr><td>thisFunction(thisArray,6,7,35)</td><td>6</td><td>7</td><td>35</td><td>6</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td><td>5
AO2.1
(3)
AO2.2
(2)</td><td></td></tr>
<tr><td></td><td>b</td><td></td><td>Binary search</td><td>1
AO2.1
(1)</td><td></td></tr>
<tr><td></td><td>c</td><td></td><td>1 mark per bullet to max 4, e.g.

- Recursion uses more memory…
- …iteration uses less memory
- Recursion declares new variables //variables are put onto the stack each time…
- …iteration reuses the same variables
- Recursive can run out of memory/stack space…
- …while iteration cannot run out of memory
- Recursion can express a problem more elegantly // in fewer lines of code…
- …while iteration can take more lines of code // be harder to understand
- Recursion will be self-referential // will call itself…
- … whereas iteration does not
</td><td>4
AO1.1
(2)
AO1.2
(2)</td><td></td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance

	d		1 mark per bullet to max 6
			• Retains function call
			• Uses a loop
			• …that will loop until all elements inspected or value found
			• Updates num1 appropriately
			• Updates num2 appropriately
			• Returns -1 in the correct place if the value has not been found
			• Returns the result in the correct place if the value has been found
			e.g.
			`function thisFunction(theArray, num1, num2, num3)`
			`  while (true)`
			`    result = num1 + ((num2 - num1) DIV 2)`
			`    if num2 < num1 then`
			`      return -1`
			`    else`
			`      if theArray[result] < num3 then`
			`        num1 = result + 1`
			`      elseif theArray[result] > num3 then`
			`        num2 = result - 1`
			`      else`
			`        return result`
			`      endif`
			`    endif`
			`  endwhile`
			`endfunction`
				6
				AO2.2
				(3)
				AO3.1
				(3)	

			Total	**16**	

Question			Answer/Indicative content	Marks	Guidance
20			1 mark per bullet • By reference will change the actual contents of the array in the main program// when control returns to the main program the array will be sorted • By value would create a copy and not change the original // when control returns to the main program the array will **not** be sorted • By value the array is local to the function. • By reference will use less memory	2 AO1.2 (1) AO2.2 (1)	
			Total	2	
21			1 mark per bullet up to a maximum of 4 marks: • The inner loop will compare all of the adjacent items…. • ….in a single pass • The outer loop will repeat the process of checking adjacent items…. • …until all passes are complete / the items are sorted/no swaps have been made in a pass	4 A01.2 (4)	Allow other valid interpretations e.g. conditional while loop … used to compare against swap flag after each pass; counter controlled for loop … used to check adjacent items on each pass
			Total	4	

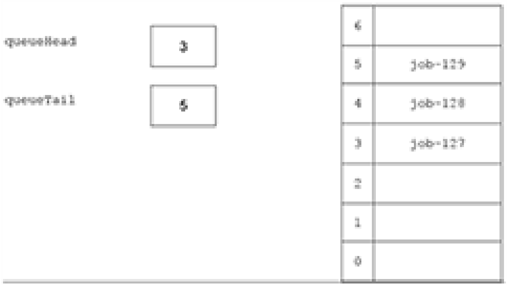
Question			Answer/Indicative content	Marks	Guidance
22			**Mark Band 3 – High level (9-12 marks)** The candidate demonstrates a thorough knowledge and understanding of the object orientied techniques; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation.The candidate is able to weigh up the use of all of the object orientied techniques which results in a supported and realistic judgment as to whether it is possible to use them in this context. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level (5-8 marks)** The candidate demonstrates reasonable knowledge and understanding of the object orientied techniques; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion showing some recognition of influencing factors that would determine whether it is possible to use each object orientied technique in this context. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence* **Mark Band 1 – Low Level (1-4 marks)** The candidate demonstrates a basic knowledge of the object orientied techniques with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than an unsupported assertion.	12 AO1.1 (3) AO1.2 (3) AO2.1 (3) AO3.3 (3)	**AO1: Knowledge and Understanding Indicative content** • Classes, this a template. It will define what attributes and methods an object should have. • Objects, when you create an instance of a class. Each object that is instantiated from the same class will share the same attributes and methods. • Inheritance, when a sub class takes on the attributes and methods from a superclass/parent class. It can also have its own extra attributes/methods. • Overriding, when a method name is the same in a parent and sub class, then the method in the parent/super class will be overridden • Encapsulation, this protects attributes of an object by making them private so that they can’t be accessed or altered accidentally by other objects. **AO2: Application** • A class can be used to declare the attributes and methods for the linked list. These will initialise the nodes and join them. • Objects can then be used be used to instantiate the class each time a new linked list is needed. Each can be given a different identifier by the other programs. • Further subclasses may be used by other programs. These can therefore take on the attributes and methods from the base class. These can also be changed or overridden depending on the purpose of the other programs. • Encapsulation can be used by using set and get methods to ensure that the nodes in the linked list are changed in a way that is intended. **AO3: Evaluation** • Use of OPP techniques will allow for code reusability. His linked list can be saved as library and then reused many times leading to less code. • OOP also allows programs to be easier to modify and maintain.

Question			Answer/Indicative content	Marks	Guidance
			The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear. 0 marks No attempt to answer the question or response is not worthy of credit.		
			Total	12	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>23</td><td>a</td><td></td><td>1 mark per pointer

- queueHead: Point to the first element in the queue // next element to remove
- queueTail: Point to the last element in the queue</td><td>2
AO1.2
(2)</td><td></td></tr>
<tr><td></td><td>b</td><td></td><td>1 mark per bullet up to max 5

- first 3 jobs removed
- 128 and 129 added in positions 4 and 5 respectively
- no additional jobs
- queueHead being 3 (FT errors)
- queueTail being 5 (FT errors)


</td><td>5
AO2.1
(2)
AO2.2
(3)</td><td>The underlying implementation of the queue has not been specified, so allow alternative valid answers.
e.g.
queueHead = 0
queueTail = 2
Location 2: 129
Location 1: 128
Location 0: 127</td></tr>
<tr><td></td><td>c</td><td>i</td><td>1 mark per bullet to max 5

- Function declaration
- Checking if queue is empty
- …returning `null`
- (Otherwise) incrementing `queueHead`
- …returning `buffer[queueHead-1]`

e.g.

```
function dequeue()
   if queueHead > queueTail then
     return null
   else
     queueHead = queueHead + 1
     return buffer[queueHead-1]
   endif
endfunction
```
</td><td>5
AO2.2
(2)
AO3.3
(3)</td><td>Note: Accept alternative valid underlying implementation answers e.g. Shifting all elements in queue forward.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance

		ii	1 mark per bullet to max 6
			• Function declaration taking parameter
			• Checking if queue is full
			• …returning -1
			• (Otherwise) incrementing queueTail
			• Adding newJob to buffer(queueTail)
			• Returning 1
			e.g.
			`function enqueue(newJob)`
			`   if queueTail == 99 then`
			`     return -1`
			`   else`
			`     queueTail = queueTail + 1`
			`     buffer[queueTail] = newJob`
			`     return 1`
			`   endif`
			`endfunction`
				6
				AO2.2
				(3)
				AO3.3
				(3)	

Question			Answer/Indicative content	Marks	Guidance

		iii	1 mark per bullet to max 8
			• Inputting user choice
			• If enqueue chosen input job name
			• …call enqueue with input value as parameter
			• …check if return value is -1 and output full
			• …otherwise output message that item is added
			• If dequeue chosen
			• …call dequeue **and** save returned value
			• …output returned value (jobname) if not null
			• …or output queue is empty
			e.g.
			`main()`
			`    choice = input("Add or remove?")`
			`    if choice == "ADD" then`
			`        jobname = input("Enter job name")`
			`        returnValue = enqueue(jobname)`
			`        if returnValue == -1 then`
			`            print("Queue full")`
			`        else`
			`            print("Job added")`
			`        endif`
			`    else`
			`        returnValue = dequeue()`
			`        if returnValue == null then`
			`            print("Queue empty")`
			`        else`
			`            output returnValue`
			`        endif`
			`    endif`
			`endmain`
				8
				AO2.2
				(2)
				AO3.3
				(6)
					Allow equivalent checks / logic

	d		1 mark per bullet to 3 • Check if either head or tail are incremented to above 99 • … set to be 0 instead • When checking if array is full check if (queueTail == queueHead – 1) OR (queueTail==99 AND queueHead==0)	3 AO2.1 (1) AO2.2 (2)	Credit equivalent modulo arithmetic solution

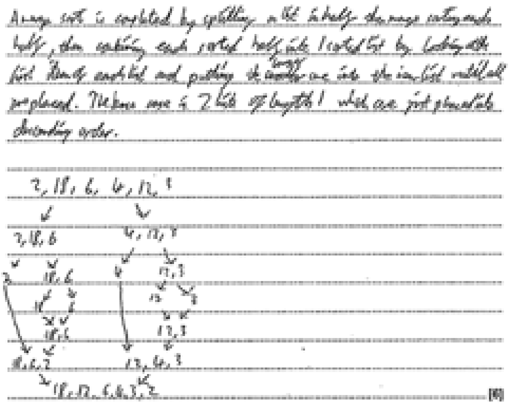
Question			Answer/Indicative content	Marks	Guidance
	e		1 mark per bullet to max 3, e.g. • Use a different structure e.g. a linked list • …items can be added at different points in the linked list depending on priority • …by changing the pointers to items needing priority • Have different queues for different priorities • …add the job to the queue relevant to its priority • …print all the jobs in the highest priority queue first	3 AO2.1 (2) AO2.1 (1)	Allow other suitable descriptions that show how the program could be amended.
			Total	32	

Question			Answer/Indicative content	Marks	Guidance

24			1 mark per bullet up to a maximum of 9 marks:
			• Defining the `createUsername` procedure correctly
			• Suitable logic for inputting the first name
			• Suitable logic for inputting the surname
			• Suitable logic for using the first letter of the first name
			• Suitable logic for joining the different sections of the username together
			• Suitable logic to pass the username into the function `existingUsers` (eg as a parameter or global variable)
			• Suitable logic for continually increasing the number by 1 …..
			• …until the username is unique
			• Sensible use of variable names and indentation throughout
				9
				A03.1 (3)
				A03.2 (6)
					Example solution:
					`procedure createUsername()`
					`    firstname = input("Enter First Name")`
					`    surname = input("Enter Surname")`
					`    number = 0`
					`    while unique == false`
					`      number = number + 1`
					`      username = surname + firstname.substring(0,1) + str(number)`
					`      unique = existingUsers(username)`
					`    endwhile`
					`    print("Username is unique")`
					`endprocedure`
					There are many different ways that this procedure could have been achieved. Therefore other alternative methods should be given credit (candidates may use `substring` or `mid` to access first character of `firstname`).

			Total	9	

Question			Answer/Indicative content	Marks	Guidance
25		i	1 mark for identifying a feature and 1 mark for stating how it can be used up to a maximum of 2 marks for each IDE feature (6 marks maximum in total.) For example: • Autocomplete (1) which will predict variable / built-in functions (1) • Auto indent (1) to automatically indent code when selection / iterative statements are used (1) • Colour coding (1) to be able to distinguish between the different parts of each statement/line (1)	6 A01.1 (3) A01.2 (3)	Allow other suitable responses that are appropriate to **writing** programming code such as automatic syntax analysis, automatic cross-referencing, line numbers and code comments, automated refactoring, automated code generation (e.g. creating templates for common patterns).
		ii	1 mark per bullet up to a maximum of 2 marks for each benefit(4 marks maximum in total), e.g: • Reduced development time (1) due to time boxing / each subtask being given a strict time limit (1) • Increased user involvement (1) so issues can be identified and fixed early / more likely to meet client requirements (1) • The requirements do not all need to be stated at the start (1) so therefore it is more flexible (1)	4 A01.1 (2) A01.2 (2)	Allow other suitable responses.
		iii	1 mark per bullet up to a maximum of 2 marks, e.g: • Black box is when the internal structure/ design is not known (to the tester) • Black box requires limited/no programming knowledge • White box is when the internal structure/ design is known (to the tester) • White box requires programming knowledge	2 A01.2 (2)	Response must cover both black box and white box testing for 2 marks.
			Total	**12**	

Question			Answer/Indicative content	Marks	Guidance
26			1 mark per bullet to max 6. Max 4 if generic description given with no application Max 4 if a diagrammatic solution is given with no description • Splits the list in half repeatedly… • … until it is in independent arrays / elements e.g. 2, 18, 6, 4, 12, 3 • Compare the first two items (index 0 and 1) e.g. 2, 18 • … and combine to create a new array in descending order i.e. 18, 2 • Repeat with indexes 2 and 3 (6, 4), then 4 and 5 (12, 3) • Compare the first element in the first two new arrays • …Choose the largest element, writing this to the new array first • …repeat until no elements left • Combine the two remaining lists into one list e.g. e.g. [2, 18, 6, 4, 12, 3] [2, 18, 6, 4, 12, 3] [2, 18, 6] [4, 12, 3] [2, 18, 6] [4, 12, 3] [2, 18] [6] [4, 12] [3] [2, 18] [6] [4, 12] [3] [2] [18] [6] [4] [12] [3] [2] [18] [6] [4] [12] [3] [18, 2] [6, 4] [12, 3] [18 2] [6] [12 4] [3] [18, 6, 4, 2] [12, 3] [18 6 2] [12 4 3] [18, 12, 6, 4, 3, 2] [18, 12, 6, 4, 3, 2]	6 AO1.2 (3 AO2.2 (3)	Allow max 5 if correct description but in ascending order. Examiner's Comments **Exemplar 3**  A merge sort is completed by splitting a list in halves, the merge sorting each half, then combining each sorted half into 1 sorted list by looking at the first item of each list and putting the larger one into the new list until all are placed. There are 6 lists of length 1 which are put placed into descending order. 2, 18, 6, 4, 12, 3 2, 18, 6 4, 12, 3 2 18, 6 4 12, 3 18 6 12 3 18, 6 12, 3 18, 6, 2 12, 4, 3 18, 12, 6, 4, 3, 2 [6] A number of candidates simply presented learnt definitions without application to the given dataset, while other candidates presented diagrammatic solutions to the workings of the merge sort for the given data, but did not explain what was happening. Good responses such as the exemplar had explanatory text alongside a clear diagram, showing the steps that took place when the algorithm was run.
			Total	**6**	
27			Iteration [1]	**1** **AO2.1** **(1)**	**Examiner's Comment:** Well answered by most candidates.
			Total	**1**	

Question			Answer/Indicative content	Marks	Guidance
28			1 mark per bullet, max 2 for each tools Breakpoints • Use to test the program works up to/at specific points • Check variable contents at specific points • Can set a point where the program stops running Stepping • Can set the program to run line by line • Slow down/watch execution • Find the point where an error occurs	4	
			Total	4	
29			• Work is easier to divide between a team • each team member just needs to know what values go into their subroutine and the expected functionality • Saves time as work takes place in parallel • each team member can work on their area of expertise. • Breaks problems into smaller areas. • Easier to test/ debug/ read • each subroutine can be tested before integration. • Code can be reused in the project/ future projects	6	Maximum 6 marks **Examiner's Comments** Many candidates achieved some credit for this answer, but few could identify and expand upon a number of different points regarding the advantages of using a modular approach. This highlighted a lack of exam technique whereby candidates did not think about the number of separate points that they were expected to give to achieve the full six marks.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
30			• Global variables are (usually) defined at the start of a program • Global variables can be seen / used everywhere in the program • Local variables can only be seen / used in a procedure / function / sub routine in which they are declared • Local variables cease to exist once the procedure / function / sub routine they are in is finished • Local variables with the same name as global variables… • …will overwrite / take precedence over the values in the global variable • Local variables within two different procedures will not interfere with one another	6	For 4th bullet accept construct Examiner's Comments A large majority of very good answers to this question with candidates writing confidently.
			Total	**6**	
31		i	*eg* • Code is generally shorter • (can be) closer to natural language description	1	Allow humans think recursively Examiner's Comments Many poor answers were given here. Many students didn't appear to have a depth of knowledge regarding recursion and its issues beyond “it calls itself”.
		ii	*eg* • Uses more memory / resources • Difficult to trace / debug	1	Difficult to understand is TV Allow difficult to follow Examiner's Comments Many poor answers were given here. Many students didn't appear to have a depth of knowledge regarding recursion and its issues beyond “it calls itself”.
			Total	**2**	

Question			Answer/Indicative content	Marks	Guidance
32	a		declarative	1	Throughout question, accept any appropriate example using the statements given in question **Examiner's Comments** Most could name the type of language as there was only one possible answer.
	b		e.g. studies_science (A,B) if student (A) and science (B)	1	**Examiner's Comments** Most candidates achieved this mark.
	c		A problem that needs to be solved e.g. student (X) ?	2	Accept e.g. "searching for a list of students" with either of answers here **Examiner's Comments** Those candidates that did not gain the full two marks here did so because their answers were either too vague or lacked an example.
	d		Setting an initial value to replace a variable e.g. find X=ben, set X=ben to test science(Y)	2	**Examiner's Comments** Another question that was set for the higher ability candidates and results on this were as expected with only a few candidates getting the full two marks on this.
	e		After finding a solution / failing to find a solution… …go back to an earlier step to test an alternative	2	Accept example that demonstrates this **Examiner's Comments** Mostly well answered by those who knew what the program was meant to achieve, a few wild guesses from candidates otherwise.
			Total	8	

Question			Answer/Indicative content	Marks	Guidance
33			Max 2 marks for explanation of benefits. Max 2 marks for example related to this scenario • Code can be re-used (1) • …Saves time (1) • Can use subroutine(s) in other programs (1) • …saves time (1) • Can test independently… (1) • … may make finding errors easier (1) • Any suitable example, e.g. the code for rolling dice can be written once (1), then called whenever needed in the game (1)	4	The question states there is only 1 programmer, so splitting the code and giving it to different programmers is not relevant to this scenario
			Total	4	

Topic 3

Algorithms

1(a) A program makes use of searching and sorting algorithms.

The following incomplete pseudocode algorithm uses a binary search to find the integer `numberToFind` in the array `array`. It returns the index of the array or -1 if the integer is not found.

Complete the pseudocode algorithm.

```
function binarySearch(array,.....................................)
  lowerbound = 0
  upperbound = array.length - 1

  while true
    if(upperbound < lowerbound) then
      return .............................................
    else
      mid = (upperbound + lowerbound) ..................................
      if(array[mid] <numberToFind) then
        lowerbound = mid .............................................
      elseif(array[mid] > numberToFind) then
        upperbound = mid .............................................
      else
        return .............................................
      endif
    endif
  endwhile
endfunction
```

[6]

(b) **An array** stores the following data:

20	8	33	16

i. Describe how the given data will be sorted into **descending** numerical order using an insertion sort.

You should refer to the data in this array throughout your answer.

[5]

ii. The size of the array has now been increased to **seven** elements.

The insertion sort algorithm needs to be tested to ensure it sorts a range of test data into **descending** numerical order.

For example, the test data in the array here will test to see if the insertion sort will sort data in the opposite order.

1	2	3	4	5	6	7

Give **two other** different sets of data in the array that can be used to test the insertion sort and state the purpose of each set of test data.

Each test needs to have a different purpose.

Set One

Test Data 1

Purpose of test data 1

Set Two

Test Data 2

Purpose of test data 2

[2]

2 A programmer is designing a program that will store data.

The programmer is deciding whether to store the data in a stack or a queue.

Identify **one** similarity and **one** difference between a stack and a queue.

Similarity

Difference

[2]

3 Kofi and Zac both write a different pseudocode algorithm to read the data from a text file into an array.

Kofi's Algorithm

```
fileName = "data.txt"
fileToRead = openRead(fileName)
for x = 0 to 1000
  anyData = fileToRead.endOfFile()
  if NOT anyData then
    x = 1001
  else
    dataValue = fileToRead.readLine()
    array[x] = dataValue
  endif
next x
fileToRead.close()
```

Zac's Algorithm

```
function readData(fileName)
  array data[100]
  x = 0
  fileToRead = openRead(fileName)
  while NOT fileToRead.endOfFile()
    data[x] = fileToRead.readLine()
    x = x + 1
  endwhile
  return data
endfunction
```

The solution needs to be used in different programs. Each program will use a different text file where the number lines in the text file is unknown.

Compare the suitability of each algorithm for the given problem.

You should include the following in your answer:

- the suitability of the programming techniques including the use of loops
- how effectively the solution meets the requirements.

[9]

i. A sorting algorithm has a best **time** complexity of O(n).
Describe what is meant by the best **time** complexity O(n) for a sorting algorithm.

[2]

ii. Another sorting algorithm has a worst **space** complexity of O(log(n)).

Describe what is meant by the worst **space** complexity O(log(n)) for a sorting algorithm.

[2]

iii. Identify the **time** complexity that means the time will not change even when the number of items increases.

[1]

iv. Identify the **space** complexity that means the amount of memory (space) used will double each time a new item is included.

[1]

5(a) A game is being written that makes use of object-oriented programming. A prototype for one part of the game is being designed that includes a character, a road and a prize to collect.

The road will have 50 spaces that a character can move along. Each space on the road will store a null value or a prize object for the user to collect. Each space is numbered sequentially from the first space (position 0) to the last space (position 49) and will not change during the game. As the player travels down the road, the position the player is on the road will be output.

The road is designed to be a 1-dimensional array with the identifier `road`.

Explain why an array is a suitable data structure to represent the road.

[3]

(b) The characters and prizes are designed as separate classes. 10 of the spaces on the road will contain an instance of the class `Prize`. The other spaces will be empty.

The class design for `Prize` is here.

`class: Prize`

`attributes:`
`private name : string`
`private type : string`
`private value : integer`

`methods:`
`new()`
`getName()`
`getType()`
`getValue()`

`new()` is the constructor method. The name, type and value are passed to the constructor as parameters which then assigns these to the attributes.

i. The method `getName()` returns the data in the attribute `name`.
   Write the method `getName()` using pseudocode or program code.

[2]

ii. A global 1-dimensional array, `allPrizes`, stores 10 objects of type `Prize`.

The prize in index 3 has the name “Box”, the type is “money” and the value is 25.

Write pseudocode or program code to create a new object for this prize and store it in index 3 of `allPrizes`.

[3]

iii. The game starts with 10 prizes. Each prize is allocated to one space on the road.

An algorithm needs designing that will generate a random space on the road for each prize. Each road space can only store one prize.

Describe the decisions that will need to be made in this algorithm and how these will affect the program flow.

[3]

(c) The class design for `Character` is here.

```
class: Character
```

```
attributes:
private name : string
private money : integer
private experience : integer
private roadPosition : integer
---------------------------------
methods:
new()
getName()
getMoney()
getExperience()
getRoadPosition()
changePosition()
updateValues()
```

The four get methods return the associated attribute.

The number of moves is passed to `changePosition()` as a parameter. The method adds this value to the character's position on the road.

The type and value of an object are passed to `updateValues()` as parameters. If the object is money the value is added to the character's money. If the type is experience the value is added to experience. If the type is neither money or experience no changes are made.

i. `new()` is the constructor method. The name of the character is passed into the constructor as a parameter. The constructor then initialises both the experience and road position of the character to 0 and initialises the amount of money to 5.

Write the constructor method for `Character` using either pseudocode or program code.

You do not need to declare the class, the attributes or any other methods.

[5]

ii. The type and value of a prize are passed as parameters to the method `updateValues`. If the type is money the value is added to the character's money. If the type is experience then the value is added to the experience. If the type is neither money or experience no changes are made.

For example, for the Character `player1`:

`player1.updateValues("money",10)` updates player1's money by 10

`player1.updateValues("experience",5)` updates player1's experience by 5

`player1.updateValues("foo",9)` has no effect on player1.

Write pseudocode or program code for the method `updateValues()`.

[5]

(d) This incomplete pseudocode algorithm:

- creates a new character with the name Jamal
- loops until the character reaches the end of the road
- generates a random number of spaces to move between 1 and 4 (including 1 and 4)
- moves the character and checks if the new space has a prize
- updates the character attributes if there is a prize
- outputs the character's new attribute values.

Complete the pseudocode algorithm.

```
character1 = new ................................. ("Jamal")
newPosition = 0
while newPosition < ................................
  move = random(1, 4) //this will generate a random number between 1 and 4
  character1.changePosition(move)
  newPosition = character1.getRoadPosition()
  if newPosition < 50 and road[..............................] != null then
  prizeType = road[newPosition].getType()
  valueAmount = road[newPosition].getValue()
  character1.updateValues(........................., valueAmount)
  print("Congratulations you are in position", newPosition, "and found",
  road[newPosition].getName())
  print("Money =", character1.getMoney(), "and experience =",
  character1. ..................... ())
  endif
.......................
print("You reached the end of the road")
```

[6]

(e) The procedure `displayRoad()` outputs the contents of each space in the road. The number of each space is output with either:

- the word “empty” if there is no prize
- the name of the prize if there is a prize.

```
01 procedure displayRoad()
02 for x = 0 to 60
03 print("Space", y)
04 if road[x] == null then
05 print("empty")
06 elseif
07 print(road[x].getValue())
08 endif
09 next x
10 endprocedure
```

The algorithm contains errors.

Give the line number of **four** different errors and write the corrected line for each error.

Error 1

Error line 1 ______________________________

Correction ______________________________

Error 2

Error line 2 ______________________________

Correction ______________________________

Error 3

Error line 3 ______________________________

Correction ______________________________

Error 4

Error line 4 ______________________________

Correction ______________________________ **[4]**

(f) A programmer is going to create a prototype for one small part of the game. Both `road` and `allPrizes` will be needed throughout the whole prototype. The programmer is considering making these global arrays as she thinks it will reduce the development time. Another programmer has suggested that doing this may create some problems when the rest of the game is created at a later stage.

Compare the use of global and local variables in this program.

You should include the following in your answer:

- the use of local and global variables
- alternative methods to using global variables
- the appropriateness of each to this program design.

[9]

6 A computer game has a building containing 7 rooms. There are secret passages between each room. **Fig. 3** shows the rooms and the passages between the rooms represented as a graph data structure.

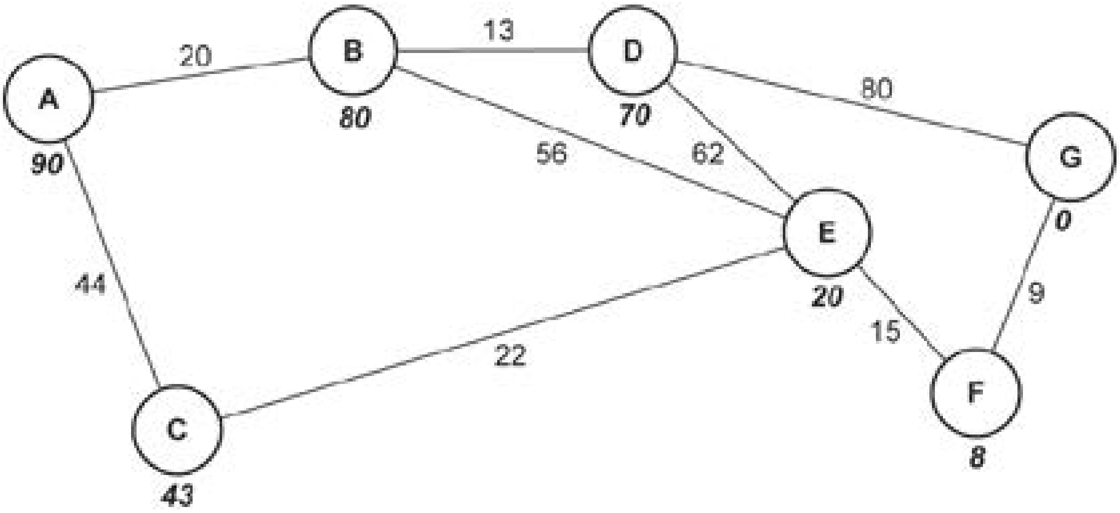


Fig. 3

The number in bold below each node in **Fig. 3** is the heuristic value.

Perform an A* algorithm on the graph shown in **Fig. 3** to find the shortest path from A to G.

Show your working, the nodes visited and the distance.

You may use the table below to give your answer.

Node	Distance travelled	Heuristic	Distance travelled + Heuristic	Previous node

Final path: ....

Distance: ....

[7]

7(a) The current contents of a queue data structure are shown in **Fig. 4**.

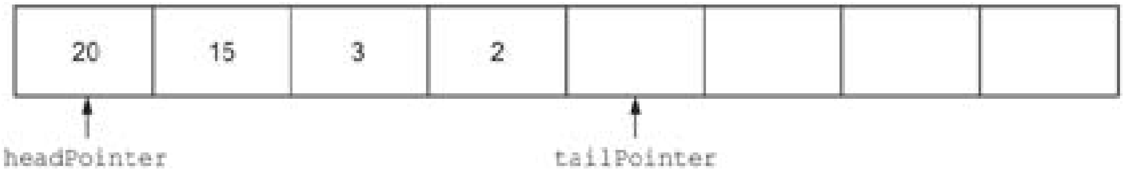


Fig. 4

`enqueue` will add data to the queue. `dequeue` will remove data from the queue.

Show the contents of the queue and the position of both pointers after the following actions have been executed on the queue shown in **Fig. 4** in the order given:

- `enqueue(20)`
- `dequeue()`
- `dequeue()`

[2]

(b) The queue is used to store ID numbers of jobs that a program needs to process. Some jobs will be given a priority which means they need to be processed first.

Explain why this queue is **not** a suitable data structure for this program.

[2]

8(a) The following strings are stored in an array.

"rainbow"	"moon"	"sun"	"stars"	"clouds"	"tornado"

Explain how a linear search would search the array for the index that stores "clouds".

[3]

(b) State why a binary search cannot be used in this example.

[1]

9 A program designer needs to decide on an algorithm to use from a choice of three. The table shows the worst-case Big O complexities for each algorithm.

Algorithm	Time Complexity	Space Complexity
1	Linear	Exponential
2	Exponential	Constant
3	Logarithmic	Logarithmic

The program will be used to analyse data that can range from 2 items to 2 billion items.

Compare the use of all **three** algorithms and suggest which the programmer should use.

You should include the following in your answer:

- the meaning of constant, logarithmic, linear and exponential complexity
- how well each algorithm scales as the amount of data increases
- which algorithm is the most suitable for the given task.

[9]

10 A graph is a type of data structure.

An example graph is shown in **Fig. 1**.

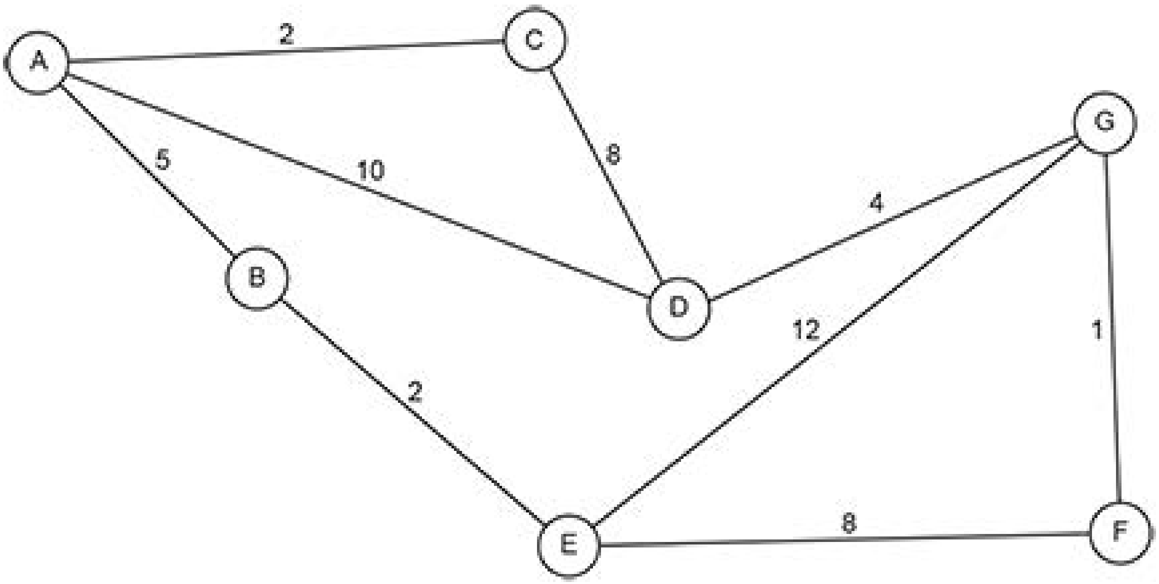


Fig. 1

Show how Dijkstra's algorithm can be used on the graph shown in **Fig. 1** to find the shortest path from start node A to end node G.

You must state the nodes on the final path and the distance of this path. Show your working.

You may use the table below to give your answer.

Node	Distance travelled	Previous node

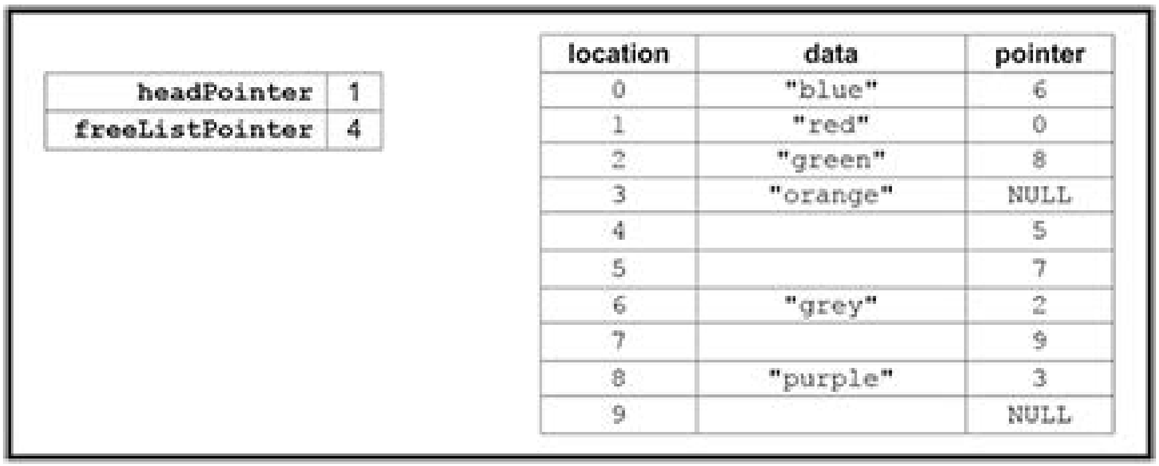
Final path: ....

Distance: ....

[6]

11 A program stores data in a linked list.

The current contents of the linked list are shown in **Fig. 3**, along with the linked list pointers.



headPointer	1
freeListPointer	4

location	data	pointer
0	"blue"	6
1	"red"	0
2	"green"	8
3	"orange"	NULL
4		5
5		7
6	"grey"	2
7		9
8	"purple"	3
9		NULL

Fig. 3

State the meaning of the pointers with the value `NULL` in the linked list shown in **Fig. 3**.

[1]

12 A function, `toBinary()`, is needed to calculate the binary value of a denary integer between 0 and 255.

`toBinary()` needs to:

- take an integer value as a parameter
- divide the number by 2 repeatedly, storing a 1 if it has a remainder and a 0 if it doesn't
- combine the remainder values (first to last running right to left) to create the binary number
- return the binary number.

For example, to convert 25 to a binary number the steps are as follows:

25 / 2 = 12 remainder 1
12 / 2 = 6 remainder 0
6 / 2 = 3 remainder 0
3 / 2 = 1 remainder 1
1 / 2 = 0 remainder 1

return value = 11001

Write the function `toBinary()`.

You should write your function using pseudocode or program code.

[6]

13 Layla writes a pseudocode algorithm to:

- input 20 positive numbers into a 0-indexed 1-dimensional array
- output the average (mean) number as a decimal
- output the smallest number
- output the largest number.

The pseudocode algorithm is shown. It contains various errors.

```
01 total = 1
02 smallest = 9999
03 largest = -1
04 for x = 0 to 21
05   dataArray[x] = input("Enter a number")
06   total = total + dataArray[x]
07   if dataArray[x] < largest then
08     largest = dataArray[x]
09   endif
10   if dataArray[x] < smallest then
11     smallest = dataArray[x]
12   endif
13 next x
14 print("Average = " + total * 20)
15 print("Smallest = " + smallest)
16 print("Largest = " + largest)
```

The algorithm that Layla has written has many errors.

Identify the line number of **four** different errors and write the corrected line of code.

Error 1 line number ....

Error 1 correction

Error 2 line number ....

Error 2 correction

Error 3 line number ....

Error 3 correction

Error 4 line number ....

Error 4 correction **[4]**

14 A computer program is being written to store data about students.

Fig. 2 shows a binary search tree that stores data about students.

Each student is represented by their ID number. The current contents of the binary search tree are:

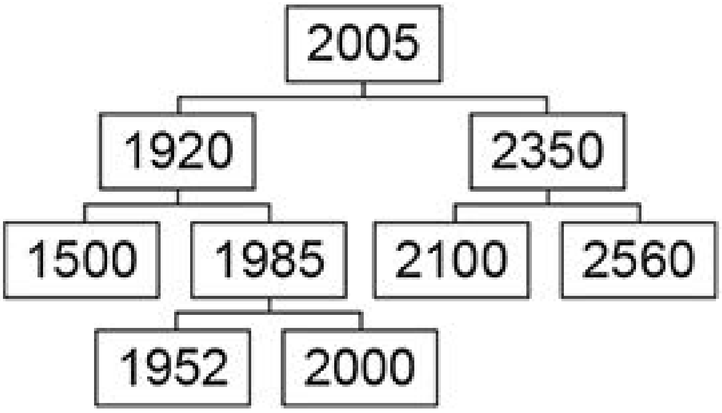


Fig. 2

* A programmer would like to traverse the binary search tree shown in **Fig. 2**.

Compare the use of a breadth-first traversal and a depth-first (post-order) traversal on the binary search tree.

You should include the following in your answer:

- how each traversal works
- the order of the return values for each traversal.

[9]

15(a) **Fig. 5** shows a graph data structure representing a small section of a parcel delivery network. Each node represents an address where deliveries need to be made. The edges show the possible routes and distances between these deliveries.

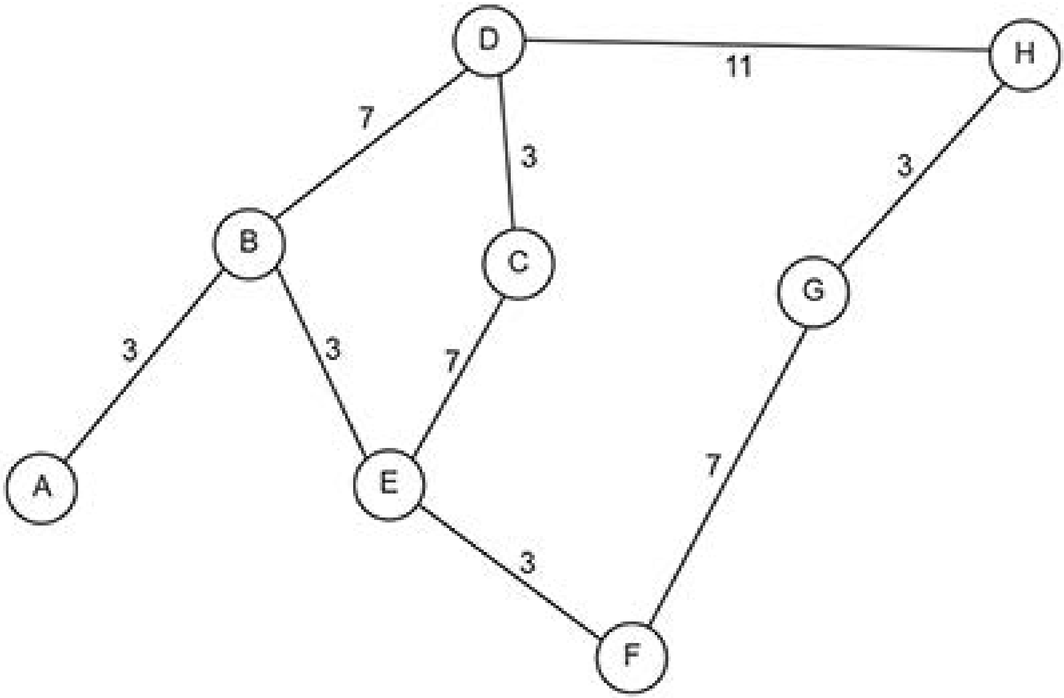


Fig. 5

Give **one** reason why the graph is a visualisation of the problem.

[1]

(b)

i. Show how Dijkstra's algorithm can be used on the graph shown in **Fig. 5** to find the shortest path from the start node A and the end node H.

You should state the nodes on the final path and the overall distance. Show your working.

You may choose to use the table below to give your answer.

Node	Distance travelled	Previous node

Final path: ....

Distance: .... **[6]**

ii. Give a similarity and difference between the performance of Dijkstra's algorithm and the performance of A* algorithm.

Similarity ____

____

Difference ____

____

[2]

16 A veterinary surgery uses a two dimensional array to store bookings for customers to bring in their animal to see the vet. There are ten possible booking slots during each day.

An example of the two dimensional array is shown in **Fig. 1**.

- The first column stores the booking slot number, ranging between 1 and 10.
- The second column stores the time of the appointment.
- The third column stores the customerID of the customer who has booked that slot.

1	9:00	5877RC
2	9:30	9655AS
3	10:00	
4	10:30	8754TT
5	11:00	
6	11:30	8745SD
7	13:00	9635GH
8	13:30	
9	14:00	9874PL

10	14:30	9658SV

Fig. 1

If a customerID has been entered for a booking slot then the booking slot has been taken. If no customerID has been entered then the booking slot is available for booking.

A function `findFirst`, is used to find the first available appointment.

Write the function `findFirst` that will find the first available appointment and return the booking slot number. If no appointments are available then the function should return `"-1"`.

You should write your function using pseudocode or program code.

[7]

17 Identify **one** situation where a linear search is more appropriate than a binary search.

[1]

18(a) A one dimensional array holds data that needs to be sorted.

Describe how a quicksort would sort data into ascending order.

---

[5]

(b) Explain why a quicksort is known as a divide and conquer algorithm.

---

[2]

19 Some of the characters in a game will move and interact independently. Taylor is going to use graphs to plan the movements that each character can take within the game.

DancerGold is one character. The graph shown in **Fig. 1** shows the possible movements that DancerGold can make.

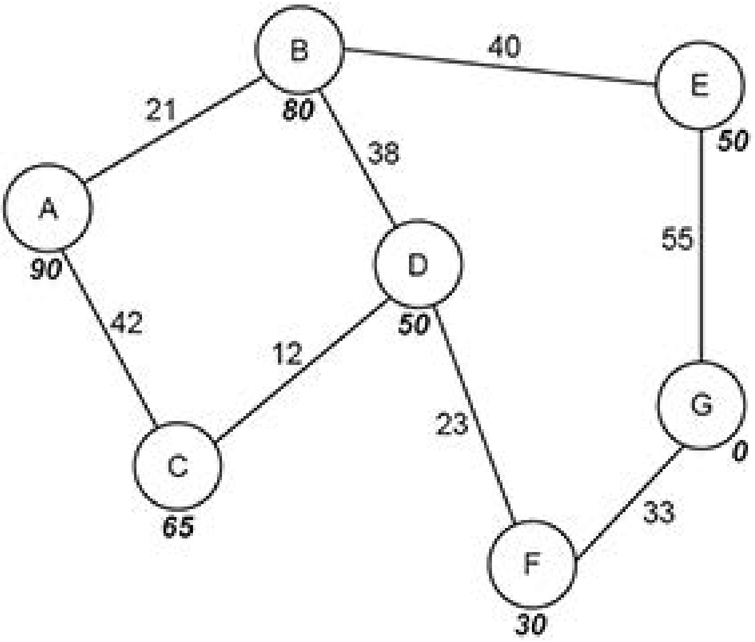


Fig. 1

DancerGold's starting state is represented by node A. DancerGold can take any of the paths to reach the end state represented by node G.

The number on each path represents the number of seconds each movement takes.

The number in bold below each node is the heuristic value from A.

i. Define the term heuristic in relation to the A* algorithm.

[2]

ii. Perform an A* algorithm on the graph shown in **Fig. 1** to find the shortest path from the starting node to the end node. Show your working, the nodes visited and the distance. You may choose to use the table below to give your answer.

Node	Distance travelled	Heuristic	Distance travelled + Heuristic	Previous node

Final path:

Distance:

[8]

20 Kira is creating a computer game where the user can play against the computer.

In each turn, each character can make one move from a selection of possible moves.

Kira uses a tree data structure shown in **Fig. 1** to identify the range of possible moves the computer can make from starting position A. Each connection is a move, with each node representing the result of the move.

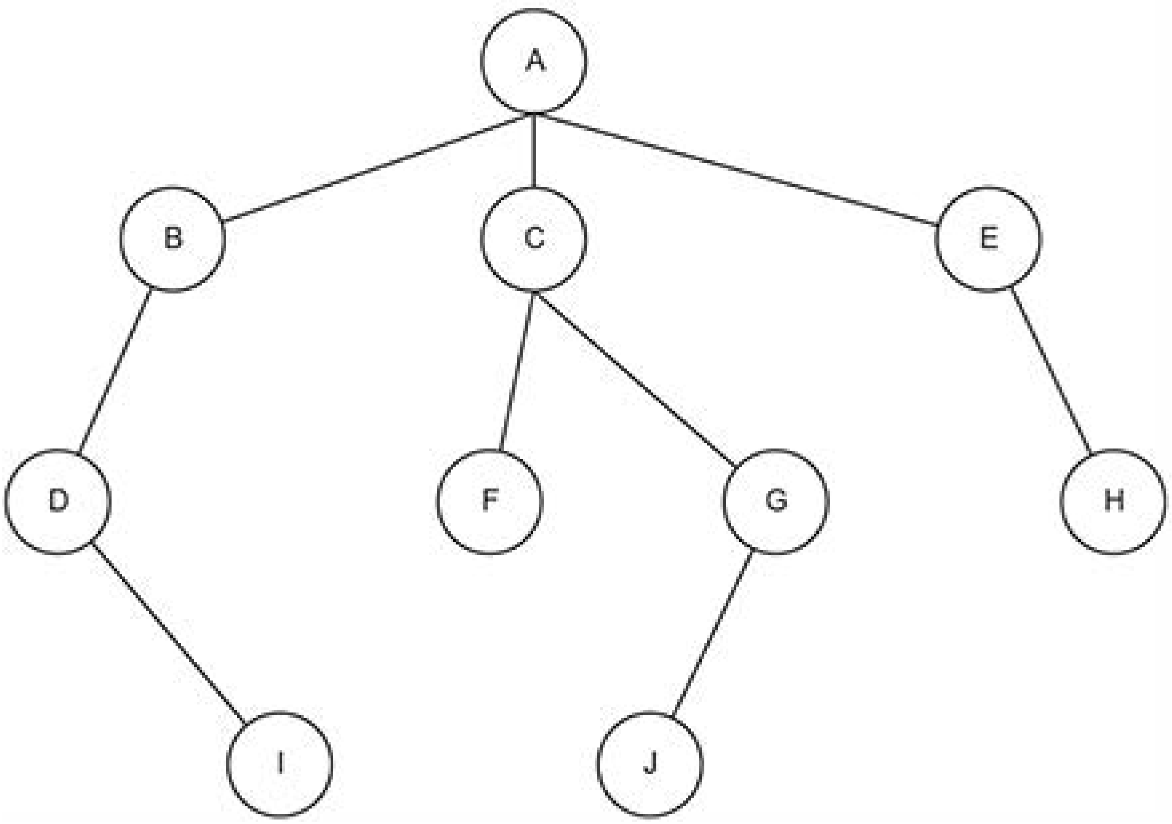


Fig. 1

Kira wants the program to traverse the tree to evaluate the range of possible moves. She is considering using a breadth-first traversal or a depth-first (post-order) traversal.

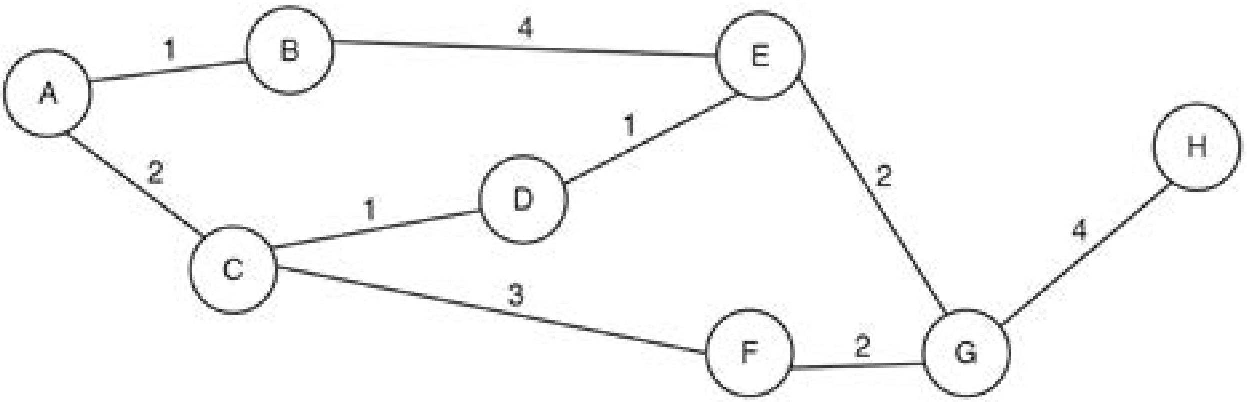
Show how a breadth-first traversal would traverse the tree shown in **Fig. 1**.

[4]

21 Mabel is a software engineer. She is writing a computer game for a client. In the game the main character has to avoid their enemies. This becomes more difficult as the levels of the game increase.

The game's 'challenging' level has intelligent enemies that hunt down the character in an attempt to stop the user from winning. The program plans the enemies' moves in advance to identify the most efficient way to stop the user from winning the game.

The possible moves are shown in a graph. Each node represents a different state in the game. The lines represent the number of moves it will take to get to that state.



Show how Dijkstra's algorithm would find the shortest path from A to H.

[6]

22 A program needs to store the names of plants that are in a garden, so they can be easily found and accessed in alphabetical order.

The data is stored in a tree structure. Part of the tree is shown.

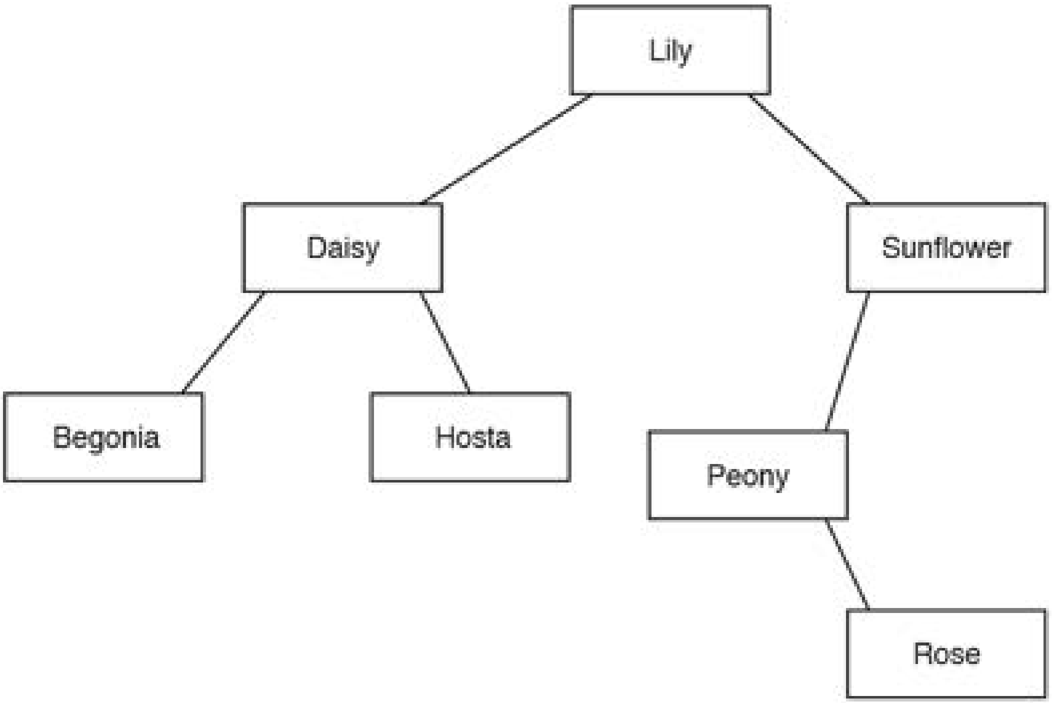


Fig. 2.1

The elements in the tree in Fig. 2.1 are read into a linked list producing an alphabetised list.

i. Complete the following table to show the linked list for the data.

Data item	Data	NextPointer
0	Begonia	
1	Daisy	
2	Hosta	
3	Lily	
4	Peony	
5	Rose	
6	Sunflower	

[2]

ii. A new plant, Lavender, needs adding to the linked list. The linked list needs to retain its alphabetical order.

Complete the table to show the linked list after Lavender is added.

Data item	Data	NextPointer
0	Begonia	
1	Daisy	
2	Hosta	
3	Lily	
4	Peony	
5	Rose	
6	Sunflower	

[3]

iii. Hosta needs removing from the linked list.

Explain how a data item is removed from a linked list. Use the removal of Hosta in your answer.

---

[4]

iv. The linked list is stored as a 2D array with the identifier `plantList`. The index of the first element of the linked list is stored in the identifier `firstElement`.

All contents of the linked list need to be output in alphabetical order.

Write an algorithm to follow the pointers to output the contents of the linked list in alphabetical order.

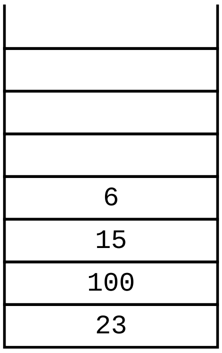
Add comments to explain your code.

[5]

23 A computer program stores data input on a stack named `dataItems`. The stack has two subprograms to add and remove data items from the stack. The stack is implemented as a 1D array, `dataArray.`

Sub-program	Description
`push()`	The parameter is added to the top of the stack
`pop()`	The element at the top of the stack is removed

The current contents of `dataItems` are shown:



The stack is implemented as a 1D array, `dataArray.`

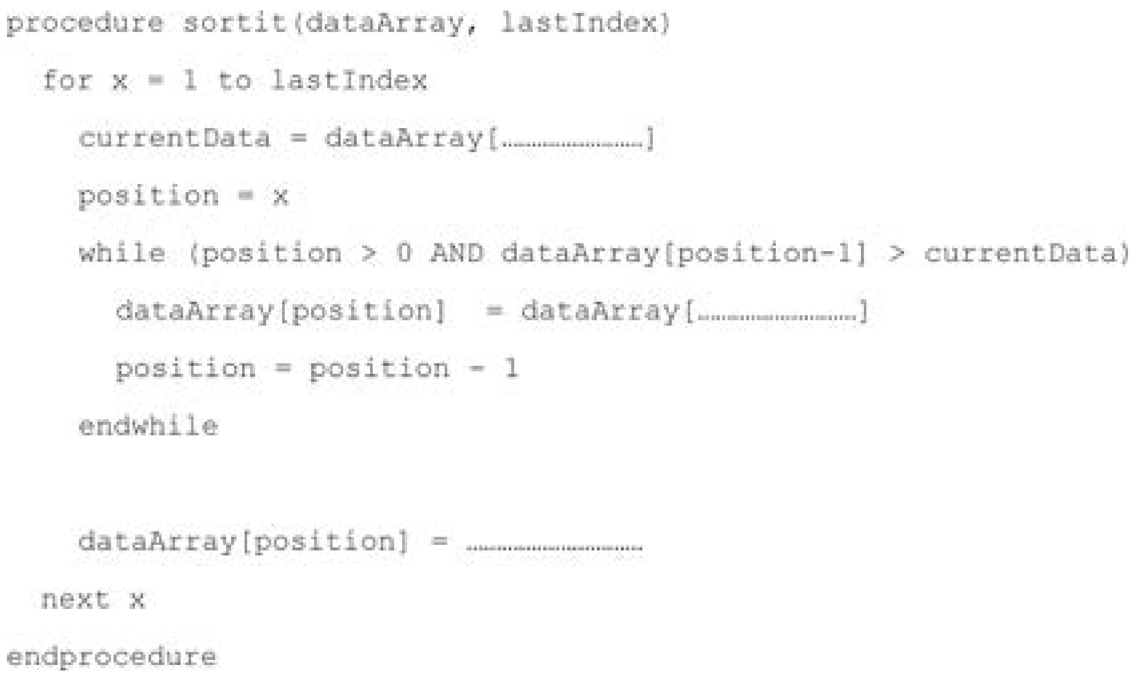
Describe how a 1D array can be set up and used to push and pop items as a stack.

[3]

24(a) A programmer needs to sort an array of numeric data using an insertion sort.

i. The following, incomplete, algorithm performs an insertion sort.

Complete the algorithm.



```
procedure sortit(dataArray, lastIndex)
  for x = 1 to lastIndex
    currentData = dataArray[……………………]
    position = x
    while (position > 0 AND dataArray[position-1] > currentData)
      dataArray[position]  = dataArray[……………………]
      position = position - 1
    endwhile

    dataArray[position] = ……………………
  next x
endprocedure
```

[3]

ii. Show how an insertion sort would sort the following data:

6	1	15	12	5	6	9

[6]

(b)

i. Using Big-O notation state the best case complexity of insertion sort.

[1]

ii. Explain what your answer to part **(b)(i)** means.

[3]

(c)

The number of data items in the array is continually increasing.

Insertion sort has a worst case time complexity of O(n^2) and space complexity of O(1).

An alternative sorting algorithm that could be used is bubble sort which also has a worst case time complexity of O(n^2) and space complexity of O(1).

Briefly outline how the bubble sort algorithm works. Discuss the relationship between the complexities and the two sorting algorithms and justify which of the two algorithms is best suited to sorting the array. **[9]**

25 A game developer is storing the number of pets a player has in a 1-dimensional array. At each timer interval, the array is searched, using a linear search, to check if any pets' hunger or bored values are greater than 90%.
If they are, an alert is displayed to the user.

i. State the complexity of searching the pets in Big-O notation.

____ [1]

ii. A given computer takes 4 milliseconds (ms) to search an array of 20 pets. Calculate an estimate of how long the computer will take to search an array of 100 pets.

Show your working.

____

____

____ [2]

26 A programmer is developing an ordering system for a fast food restaurant. When a member of staff inputs an order, it is added to a linked list for completion by the chefs.

Explain why a linked list is being used for the ordering system.

____

____

____

____ [2]

27 Dexter is leading a programming team who are creating a computer program that will simulate an accident and emergency room to train hospital staff.

Two of Dexter's programmers have developed different solutions to one part of the problem. Table 3.1 shows the Big O time complexity for each solution, where *n* = the number of data items.

	Solution A	**Solution B**
Time	O(n)	O(n)
Space	O(k^n) (where k > 1)	O(log n)

Table 3.1

i. The Big O time complexity for time of each solution is O(n).

Explain what is meant by time complexity, with reference to the solutions' Big O time complexity.

____

____

____

____

____

____ [3]

ii. Name the space complexity for each solution:

Solution A ____

Solution B ____

[2]

iii. Explain, with reference to the Big O complexities of each solution, which solution you would suggest Dexter chooses.

____

____

____

____

____

____

____

____ [4]

28(a) Fig. 2.1 shows the flight paths between a country's airports. The value in bold beneath each node is the heuristic value from E.

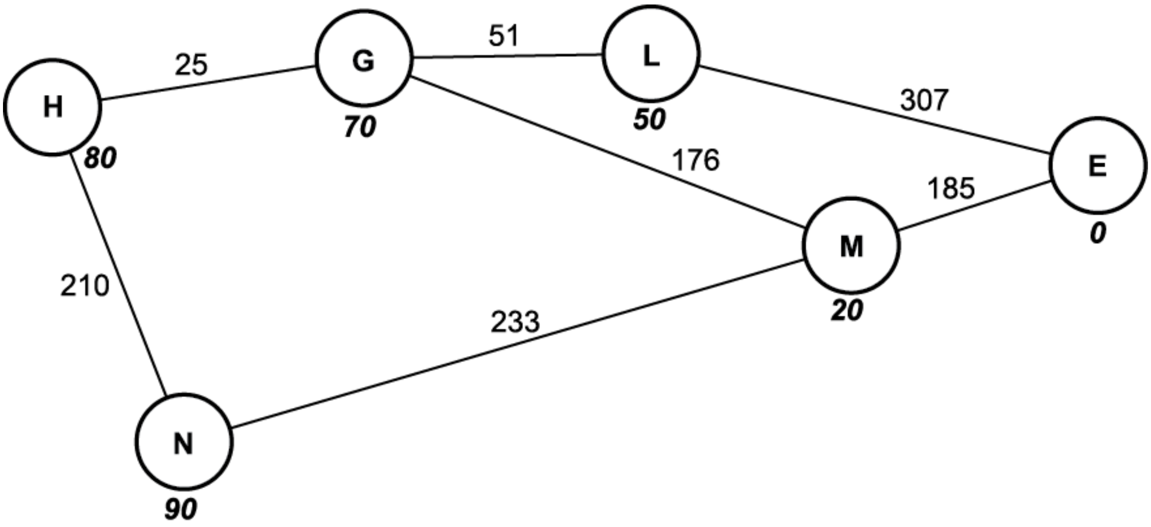


Fig. 2.1

State the full name of the data structure shown in Fig. 2.1.

-- **[2]**

(b) The structure in Fig. 2.1 is searched using the A* algorithm making use of the heuristic values.

i. State what the heuristic values could represent in Fig. 2.1.

-- **[1]**

ii. State the purpose of heuristic values in the A* algorithm.

-- **[1]**

iii. Perform an A* algorithm on the data structure in Fig. 2.1 to find the shortest distance between H and E. Show each step of the process, and the calculations performed for each node visited.

----

[8]

iv. Give **one** decision that is made in the A* algorithm, and describe the effect of this decision on the next step(s) of the algorithm.

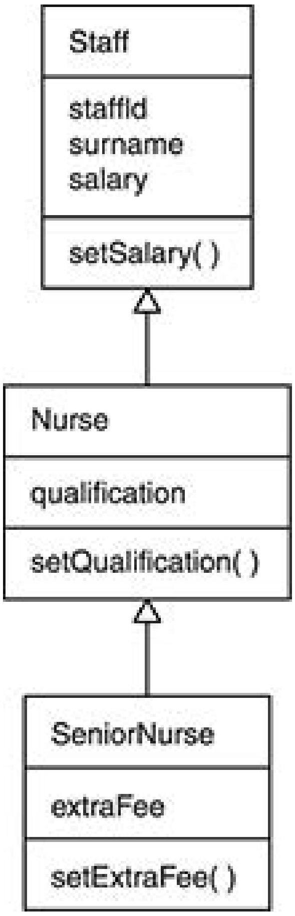
Decision

Effect

[3]

29 A large health centre employs a number of nurses, each with a qualification. Some nurses are senior nurses, and these are paid an extra fee in addition to their salary.

The diagram below shows part of the system used.



i. State the type of diagram shown.

-- **[1]**

ii. State the term that describes setSalary( ), setQualification( ) or setExtraFee( ).

-- **[1]**

iii. Explain the meaning of the arrows in the diagram, using an example.

----

[2]

iv. Mary Jones is a new employee at the health centre.

State why the following program statement is **not** valid.

```
Jones.setSalary(12000)
```

[1]

v. You may assume that thisSeniorNurse has been correctly defined as an object of SeniorNurse, and that x is a number.

Explain why the program statement

```
thisSeniorNurse.setSalary(x)
```

is valid even though setSalary( ) is not shown in the SeniorNurse part of the diagram.

[3]

30(a) A salesman travels around the country, stopping at specific places, and then returning to the starting place.

Fig 6.1 shows an example map of places that the salesman visits.

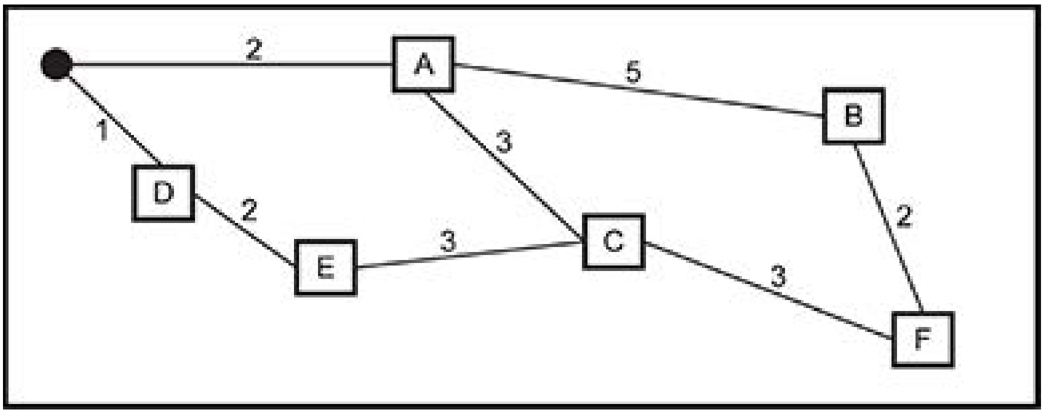


Fig 6.1

The filled in circle represents the start and end point. The letters represent the places to visit. The lines are the routes available and the numbers are the length of time each route takes to travel.

Explain how abstraction has been applied in the production of Fig 6.1

[2]

(b) The travelling salesman aims to find the shortest route between these places to visit.

A programmer is writing an algorithm to solve the travelling salesman problem.

The programmer is using a tree to find the most efficient route. Fig 6.2 shows part of the tree with three levels completed.

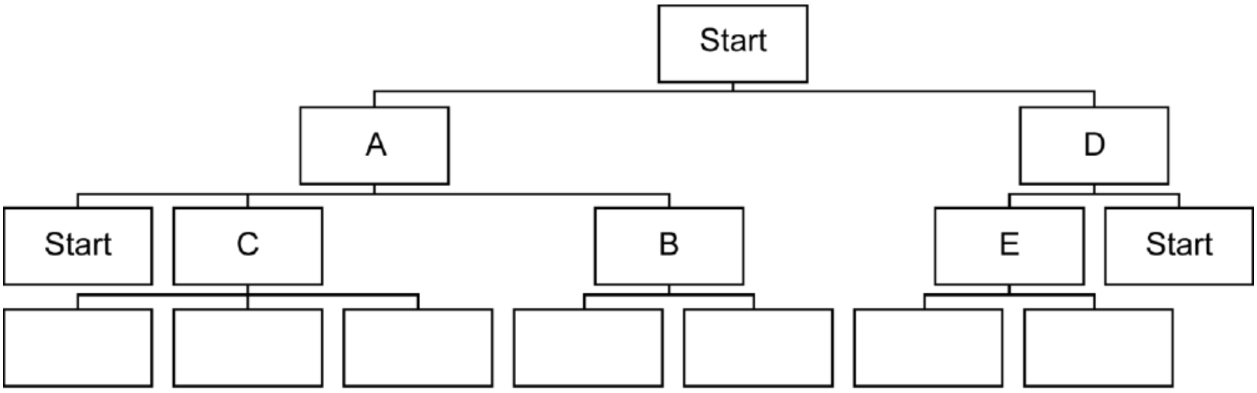


Fig 6.2

i. The 'Start' nodes on level three are not expanded again as this is a repeat, 'Start' has already been expanded.

Write the place names in the boxes in Fig 6.2, to complete the fourth level of the tree structure for the map shown in Fig 6.1.

[3]

ii. Explain why the tree in Fig 6.2 is **not** a binary tree.

____

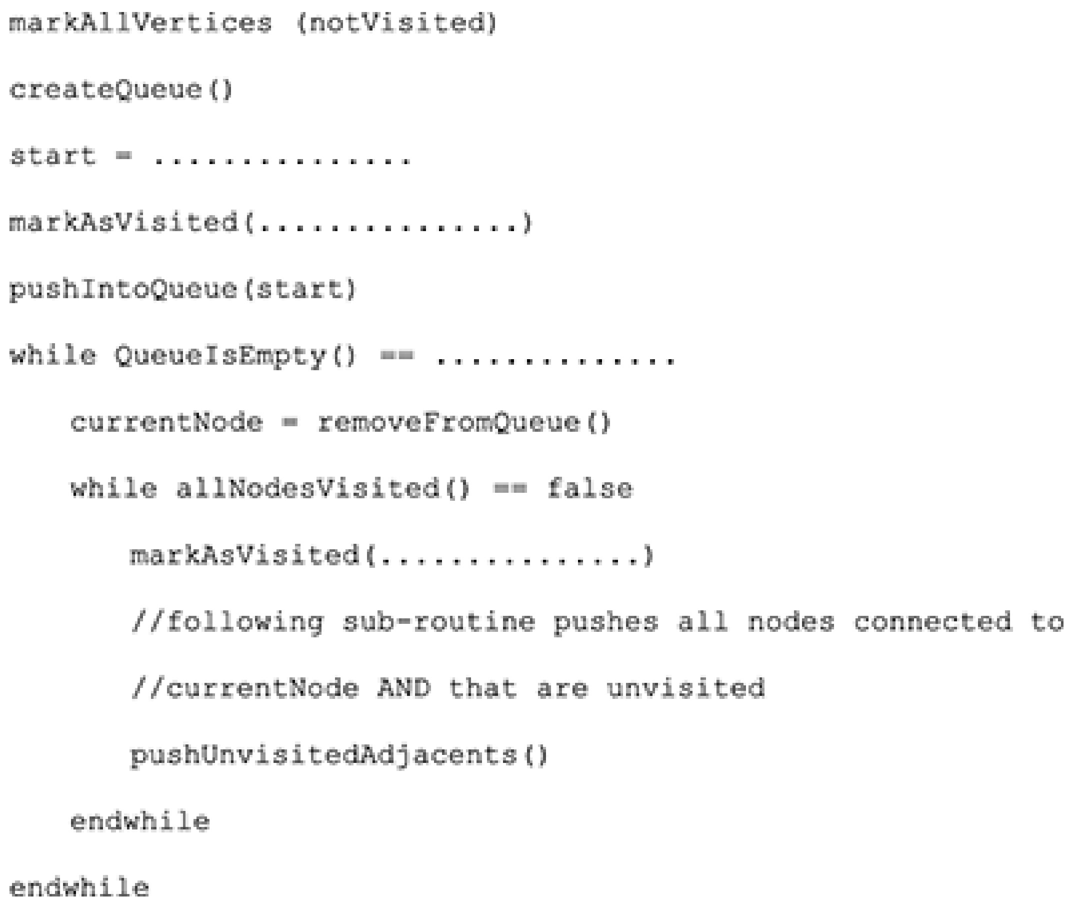
____ **[1]**

(c) The programmer has decided to use a graph instead of a tree structure.

i. Describe what is meant by a graph structure.

[2]

ii. The pseudocode below shows part of an algorithm which uses a queue to traverse the graph breadth-first. Complete the missing elements of the algorithm.



```
markAllVertices (notVisited)

createQueue()

start = ...............

markAsVisited(...............)

pushIntoQueue(start)

while QueueIsEmpty() == ..............

    currentNode = removeFromQueue()

    while allNodesVisited() == false

        markAsVisited(...............)

        //following sub-routine pushes all nodes connected to

        //currentNode AND that are unvisited

        pushUnvisitedAdjacents()

    endwhile

endwhile
```

[4]

(d) Fig 6.3 is a graph representation of the places that the travelling salesman visits. Using this graph, show how Dijkstra's algorithm would find the shortest path from place A to place F.

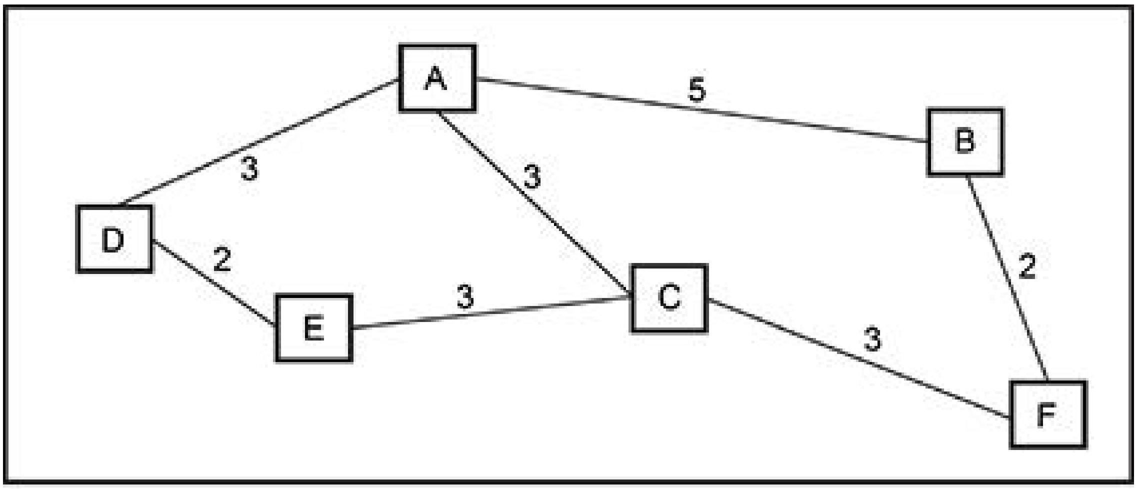


Fig 6.3

[6]

31(a) Linear search and binary search are two different algorithms which can be used for searching arrays.

When comparing linear and binary search it is possible to look at the best, worst and average number of items in the array that need to be checked to find the item being searched for. Assume every item in the array is equally likely to be searched for.

Complete the table below

	Worst Case number of searches	Average Case	Best Case
Binary Search		$\text{Log}_2(n)$-1	
Linear Search		n/2	

[4]

(b) As the size of an array increases the average number of checks grows logarithmically. State what is meant by logarithmic growth.

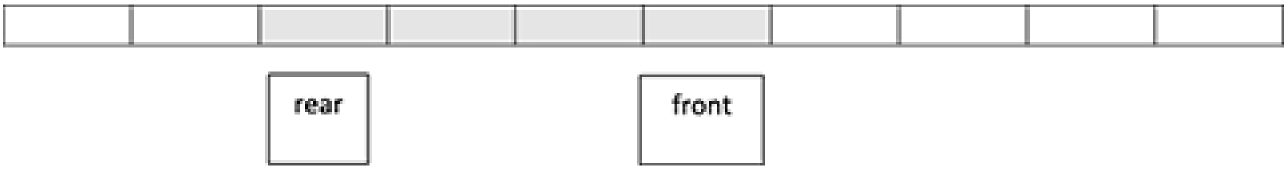
[1]

(c) Assuming an array is sorted give a situation when a linear search would perform better than a binary search.

[1]

32(a) The array queue shown below is set up to hold a small queue. Assume that there is sufficient storage to hold all necessary additions to the queue.

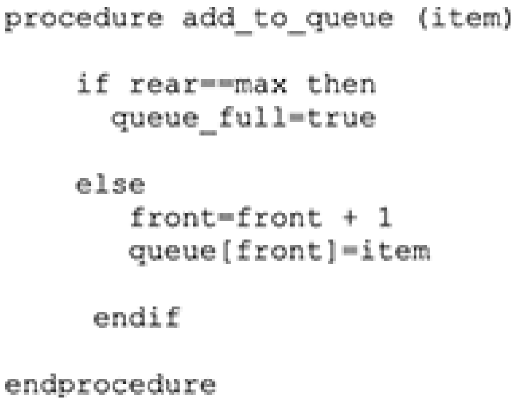
Queue



The table below shows variables that are used to maintain the queue:

Variable	Type	Purpose
front	integer	pointer to the front element of the queue
rear	integer	pointer to the rear element of the queue
queue_full	Boolean	indicates whether the queue is full
max	integer	the maximum size of the queue

Shown below is an algorithm that is intended to add an item to the queue.



```
procedure add_to_queue (item)

    if rear==max then
      queue_full=true

    else
       front=front + 1
       queue[front]=item

     endif

endprocedure
```

Identify the parameter that is passed to this procedure.

-- [1]

(b)

i. This algorithm contains a logic mistake. Explain what the mistake is.

---[2]

ii. Rewrite the algorithm to correct the mistake.

---[2]

END OF QUESTION PAPER

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>1</td><td>a</td><td></td><td>

1 mark for each completed statement

```
function binarySearch(array, numberToFind)
  lowerbound = 0
  upperbound = array.length - 1

  while true
    if(upperbound <lowerbound) then
      return -1
    else
      mid = (upperbound + lowerbound)DIV 2
      if(array[mid] <numberToFind) then
        lowerbound = mid + 1
      elseif(array[mid] > numberToFind) then
        upperbound = mid - 1
      else
        return mid
      endif
    endif
  endwhile
endfunction
```

</td><td>6</td><td>

Accept
`mid` = **int**((`upperbound` + `lowerbound`)/ 2)

<u>Examiner's Comments</u>

Binary search (in both iterative and recursive form) is a standard algorithm that candidates are expected to be able to code at AS Level. Candidates made a number of common errors including not using integer division (`DIV` and // were accepted) for the midpoint calculation.

Some candidates returned the value being searched for at the array index rather than returning mid as the index as specified in the question.

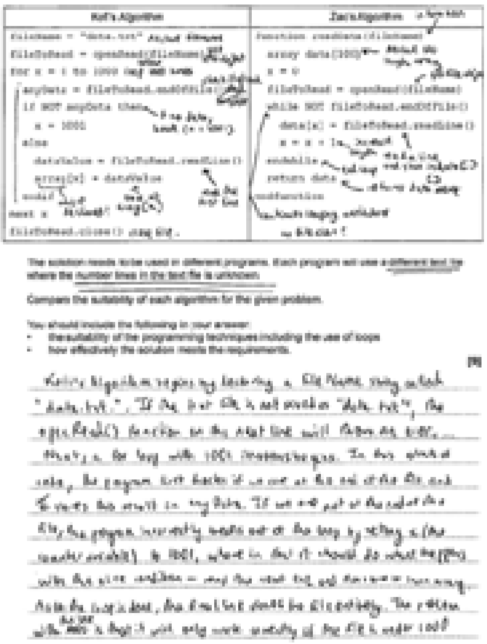
Another common error that was made was to update `mid` by the wrong value, swapping the +1 and -1 round.

</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
	b	i	1 mark per bullet to max 5 • 20 becomes the sorted list // [20] is the sorted list • ... take 8 and compare against ... • ... it is less than 20 so keep it in place // [20 8] is now the sorted list • Take 33 and compare it to each value in the sorted list. It is greater than 20, so move 20 and 8 and insert 33 // 33 is compared and moved down and [33 20 8] is now the sorted list • Repeat for all other elements.	5	Max 4 if ascending and not descending Zero marks if not provided a description and just shown data values Examiner's Comments Insertion sort is one of the standard sorting algorithms that candidates need to be familiar with using. A pleasing number of candidates made a reasonable start to the response, but fewer could produce a comprehensive response. The question required candidates to describe and not just show how insertion sort worked, so responses that only gave diagrams showing numbers moving with no annotation did not gain credit. Some candidates mistakenly described bubble sort or merge sort. Quite often, where candidate scored four marks, they omitted the first pass of the sort where the first item (`20`) is set as the sorted list. Some potentially quite good responses were too vague as to how the next item at the start of the next pass filtered back through the sorted list to its correct position.
		ii	1 mark for each set of test data **and** purpose e.g. • **Test Data:** 7 6 5 4 3 2 1 **Purpose:** Testing that data is already in order • **Test Data:** 1 7 2 6 3 5 4 **Purpose:** Testing that mixed data is sorted • **Test Data:** Six One Seven Three Four Fix Two • **Purpose:** Testing that words are rejected/ Testing words are sorted (alphabetically) • **Test Data:** -1, 0, 2, -1, 3, -3 • **Purpose:** Testing the algorithm works with negative numbers.	2	Each purpose must be different. Accept viable alternatives The purpose must be relevant to the test data given. A mark cannot be award unless both the test data and purpose are given. Examiner's Comments Many candidates gained some credit for giving a valid reason for a set of test data that they presented. Clear responses included testing decimals, negatives, different data types, and testing on data that was not already in order.
			Total	13	

Question			Answer/Indicative content	Marks	Guidance
2			1 mark for similarity: • Both are data structures / store data • Both allow data to be added/removed • Both have identifier(s) / pointer values 1 mark for difference: • Stack is LIFO and queue is FIFO • Queue supports enqueue/dequeue operations whilst stack supports push/pop operations • Queue require two pointers whilst stacks only require one pointer	2	Examiner's Comments Most candidates demonstrated knowledge of the purpose of stacks and queues. For the similarity, a frequent response was that both stacks and queues are data structures used to store data. For the difference, most candidates identified that a stack was a LIFO structure while a queue was a FIFO structure.
			Total	2	

Question			Answer/Indicative content	Marks	Guidance
3			**Mark Band 3-High Level** **(7-9 marks)** The candidate demonstrates thorough knowledge and understanding of suitability of algorithms; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate provides a thorough discussion which is well-balanced. Evaluative comments are consistently relevant and well-considered. There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated. **Mark Band 2-Mid Level** **(4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of suitability of algorithms; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are for the most part appropriate, although one or two opportunities for development are missed. There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence. **Mark Band 1-Low Level** **(1-3 marks)** The candidate demonstrates a basic knowledge of suitability of algorithms, with limited understanding shown; the material is basic and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgments if made are weak and unsubstantiated. The information is basic and	9	**AO1: Knowledge and Understanding** e.g. • Kofi’s algorithm reads data from the text file and writes these value to an array. It cannot be called by different programs, doesn’t work with different text files and does not consider the number of items in the file. • Zac’s program also reads data from the text file and writes these values to an array. It can be called by different programs, works with different file names and does consider the number of items in the file. **AO2.1: Application** e.g. Kofi's algorithm: • Has additional/unnecessary variables • ...e.g. fileName, dataValue • Loops 1000 times which is inappropriate when it needs to be used with unknown quantity • Will not read any lines after the first 1000 Zac's algorithm: • Uses a function so that is can be used in different programs // is independent • Takes file name as parameter so that it can be called with different files • The array size is 100. Therefore, if the data being read is greater than 100 the program will crash. • Does not close the file **AO3.3: Evaluation** e.g. • Zac's is more memory efficient as it uses fewer variables • Kofi's is more suitable in that it closes the file so it does not remain open in memory • Zac's is more suitable to be used in different programs due to function and not hard coding the file name • Zac’s is easier to alter for a different file size as fewer lines of code would need to be altered **Examiner’s Comments**

Question			Answer/Indicative content	Marks	Guidance
			communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear. **0 mark** No attempt to answer the question or response is not worthy of credit.		There were a large number of either blank or very brief responses where candidates displayed no ability to read and interpret code. The question source material was designed to help candidates to comment on a range of programming practices such as file opening and closing, conditional versus count-controlled loops and the benefits of the use of functions with parameters. Level 1 responses often just focused on the use of two different loop types. There were few Level 3 responses that gave insightful evaluative comments. Where candidates were able to do so they often focused on memory usage or the ability to cope with different data sets that could be read. Exemplar 2 

Kofi's Algorithm

```
filename = "data.txt"
fileToRead = openRead(filename)
for x = 1 to 1000
  anyData = fileToRead.endOfFile()
  if NOT anyData then
    x = 1001
  else
    dataValue = fileToRead.readLine()
    array[x] = dataValue
  endif
next x
fileToRead.close()
```

Zac's Algorithm

```
function readData(filename)
  array data[1000]
  x = 0
  fileToRead = openRead(filename)
  while NOT fileToRead.endOfFile()
    data[x] = fileToRead.readLine()
    x = x + 1
  endwhile
  return data
endfunction
```

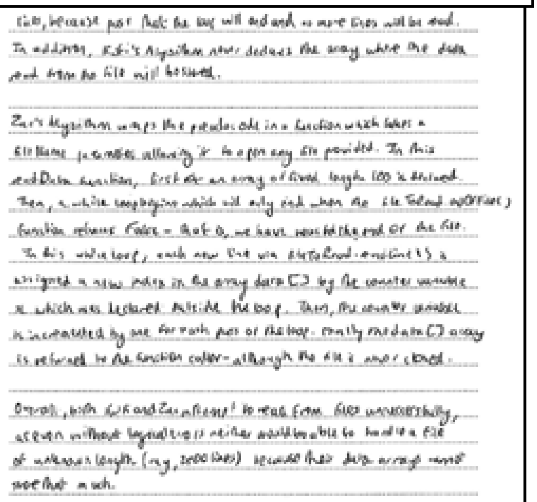
The solution needs to be used in different programs. Each program will use a different text file where the number lines in the text file is unknown.

Compare the suitability of each algorithm for the given problem.

You should include the following in your answer:
- the suitability of the programming techniques including the use of loops
- how effectively the solution meets the requirements.

[9]

Kofi's algorithm begins by declaring a file name string called "data.txt". If the text file is not called "data.txt", the openRead() function on the next line will throw an error. Then, a for loop with 1000 iterations begins. In this block of code, the program first checks if we are at the end of the file and stores this result in anyData. If we are not at the end of the file, the program incorrectly breaks out of the loop by setting x (the counter variable) to 1001, when in fact it should do what happens with the else condition – read the next line and store it in the array. After the loop is done, the final line closes the file correctly. The problem with the for loop is that it will only work correctly if the file is under 1000

Question			Answer/Indicative content	Marks	Guidance
					 file, because past that the loop will end and no more lines will be read. In addition, Kofi's Algorithm never declares the array where the data read from the file will be stored. Zac's Algorithm wraps the pseudocode in a function which takes a filename parameter allowing it to open any file provided. In this readData function, first an array of fixed length 100 is declared. Then, a while loop begins which will only end when the file.endOfFile() function returns False – that is, we have reached the end of the file. In this while loop, each new line via file.readLine() is assigned a new index in the array data[] by the counter variable x which was declared outside the loop. Then, the counter variable is incremented by one for each pass of the loop. Finally the data[] array is returned to the function caller – although the file is never closed. Overall, both Kofi and Zac attempt to read from files unsuccessfully, as even without logical errors neither would be able to handle a file of unknown length (e.g. 1000 lines) because their data arrays cannot store that much. This response showed that the candidate has thought about the issues related to the question and had carefully annotated the code when analysing it. The response was well-structured taking each of the algorithms in turn and displayed a clear level of evaluative insight.
			Total	9	

Question			Answer/Indicative content	Marks	Guidance
4		i	1 mark each • Linear • The time will (increase) in **direct proportion** to the number of items	2	Examiner's Comments Many candidates did not identify linear but gave a description for 'directly proportional' or vice-versa, so only gained one mark. Candidates needed to specify that the time required increased in direct proportion to the number of items. Proportion on its own was too vague and could have meant inverse proportion for example. Incorrect responses included 'quickest time to find / least operations' defining best case rather than answering the question regarding O(n).
		ii	1 mark each • Logarithmic • The additional memory space required grows at a decreasing rate as the number of items increases	2	Examiner's Comments Many candidates did not identify logarithmic but tried to give a description or vice-versa. Many candidates just said the memory space increased in proportion to the log of n without explaining what this meant. For the second mark candidates had to explain that as the number of items n increased the amount of additional memory required became progressively smaller.
		iii	Constant // O(1)	1	Examiner's Comments The vast majority of candidates gave the correct response of constant or O(1).

Question			Answer/Indicative content	Marks	Guidance
		iv	Exponential // O(2^n) // O(K^n)	1	**Examiner's Comments** Half the candidates correctly identified exponential complexity, but common erroneous responses included O(n^2) or O(2n) instead of O(2^n). **Misconception** A number of candidates erroneously thought that n^2 or 2^n demonstrated exponential growth instead of 2^n. Candidates need to have the mathematical grounding to understand the difference between different Big O growth factors.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
5	a		1 mark each to max 3. Max 2 for generic answers with no relation to scenario. e.g. • Has a set/fixed number of values • …and the number of spaces in the road will not change • Stores data of one type • … as the array is only made up of prize objects • Stores data linearly • … match the linear nature of the road • Array contents are mutable • … so prizes can be added/removed from the road • A single indentifier is used to directly index • … any position in the road • Can be iterated by index • … to perform an operation on all road positions	3	**Examiner's Comments** Many responses were too vague, showing little knowledge of the properties of arrays. Relatively few candidates appeared to be able to make explicit links to the scenario to achieve full marks.

	b	i	1 mark each
			• Function/subroutine with identifier `getName` taking **no** parameters
			• Returning `name`
			e.g.
			`public function getName()`
			`  return name`
			`endfunction`
			`def getName(self):`
			`  return self.__name`
			`public getName(){`
			`  return name`
			`}`
			`function getName() {`
			`  return this.name`
			`}`
				2	BP1 Do not award procedure or method
					BP1 Allow self as an additional parameter if Python is used.
					BP1 If an access modifier is given for the method, it must be public and not private.
					BP2 Do not allow any modified name attribute to be returned.
					Examiner's Comments
					While many candidates had little difficulty giving code for a `getter()` there were a number of common errors. Some candidates used a private access modifier when a `getter()` needs to be public. There was often erroneous use of 'procedure' whereas a `getter()` is a function that must return a value.
					Some candidates tried to set values within the `getter()` function when it should only have returned the class attribute value.

Question			Answer/Indicative content	Marks	Guidance

		ii	1 mark each
			- New instance of `prize` …
			- … with “Box”, “money” and 25 as parameters
			- Assigned to `allPrizes` index 3
			
			e.g.
			`allPrizes[3] = new prize("Box", "money", 25)`
			`allPrizes[3] = prize.new("Box", "money", 25)`
			`allPrizes[3] = prize("Box", "money", 25)`
				3	MP2 allow any order of parameters
					
					`"Box"` and `"Money"` must be strings and `25` must be an integer
					
					Allow `prize.new()` as new is given as the constructor method in the class diagram
					
					Examiner’s Comments
					
					Many candidates struggled with the instantiation of an object. Where candidates made an attempt to instantiate some did not use a string for “box” and “money” or did not give 25 as an integer but instead gave the string “25”.

		iii	1 mark for each bullet to maximum 3 e.g. - **Decision** - check whether the space already has a prize allocated … - **Action if true** - another space/number will need to be generated - **Action if false** - the prize will be stored here - **Decision** - check if all 10 prizes have been allocated … - **Action if true** - the algorithm needs to stop generating numbers - **Action if false** - a new number/space needs to be generated and checked	3	Give: - 1 mark for stating a decision - 1 mark for the action required if true - 1 mark for the action required if false Examiner’s Comments There were only two reasonable decisions that could be given from the scenario details. Candidates needed to make it clear that a decision with a Boolean output was present that would dictate two potential outcomes. Some candidates quoted actions such as ‘randomly assign space for prize’ which did not represent a decision. Many responses described the mechanics of setting up the game and the random spaces but did not highlight the program conditions/decisions as required.

Question			Answer/Indicative content	Marks	Guidance

	c	i	1 mark each
			• Constructor header (any suitable name e.g. new, constructor, create, init)
			• …taking **one** parameter only
			• Initialising name to the parameter
			• Initialising money to 5
			• Initialising experience to 0 and roadPosition to 0
			e.g.
			Pseudocode example:
			`public procedure new(pName)`
			`    name = pName`
			`    experience = 0`
			`    roadPosition = 0`
			`    money = 5`
			`endprocedure`
			Python Example:
			`def __init__(self, pName):`
			`    self.__name = pName`
			`    self.__experience = 0`
			`    self.__roadPosition = 0`
			`    self.__money = 5`
			C# Example:
			`public Character(string pName) {`
			`    string name = pName;`
			`    int experience = 0;`
			`    int roadPosition = 0;`
			`    int money = 5;`
			`}`
				5	Allow minor changes to identifiers as long as purpose is clear.
					Allow
					`procedure new(pName)`
					`    this.name = pName`
					…
					(or similar e.g. self.name)
					Allow two parameters if one is *self* and the response is clearly in Python.
					The parameter name should be different to the attribute name.
					Examiner's Comments
					It was clear that those candidates with limited OOP programming knowledge found the writing of a relatively simple constructor method difficult. Those with relevant programming experience often found this to be a very straightforward question. Common errors included passing additional values to set the `experience`, `roadPosition` and `money` attributes rather than setting them to the constant values indicated in the question.

Question			Answer/Indicative content	Marks	Guidance

		ii	1 mark each
			- Procedure/method header …
			- … taking **two** parameters, type (or similar) followed by value (or similar) …
			- … compare type parameter with `"money"`
			- … compare type parameter with `"experience"`
			- … both attributes updated correctly and nothing else modified
			e.g.
			`public procedure updateValues(pType, pValue)`
			`    if pType == "money" then`
			`      money = money + pValue`
			`    elseif pType == "experience"`
			`      experience = experience + pValue`
			`    endif`
			`endprocedure`
			
			`def updateValues(self, pType, pValue):`
			`    if pType == "money":`
			`      money += pValue`
			`    elif pType == "experience":`
			`      experience += pValue`
				5	Do not allow Function for BP1
					
					BP2 parameters must be given in the correct order to match the calls to `updateValues()` in the question.
					
					“`money`” and “`experience`” must be string values
					
					<u>Examiner’s Comments</u>
					
					The `updateValues` procedure again proved problematic for candidates with limited OOP experience. No marks were given for the first mark point if a function was declared as there was no return value. Parameter names needed to be fit for purpose, understandable, and had to match the order given in the question scenario to work for the given example calls.

Question			Answer/Indicative content	Marks	Guidance

	d		1 mark for each completed space
			
			`character1 = new `**`Character`**`("Jamal")`
			`newPosition = 0`
			`while newPosition < `**`50`**
			`    move = random(1, 4)`
			`    character1.changePosition(move)`
			`    newPosition = character1.getRoadPosition()`
			`    if newPosition < 50 and road[`**`newPosition`**`] != null then`
			`      prizeType = road[newPosition].getType()`
			`      valueAmount = road[newPosition].getValue()`
			
			`      character1.updateValues(`**`prizeType`**`, valueAmount)`
			`      print("Congratulations you are in position", newPosition, "and found", road[newPosition].getName())`
			`      print("Money", character1.getMoney(), "and experience", character1.`**`getExperience`**`())`
			`    endif`
			`endwhile`
			`print("You reached the end of the road")`
				6	Allow `road.length // len(road)` instead of `50`
					
					Allow `<=49` instead of `< 50`
					
					Examiner's Comments
					
					Nearly all candidates achieved some marks, and a majority scored five or six marks.

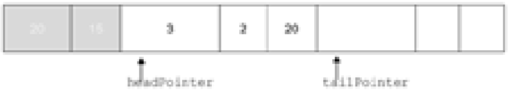
	e		1 mark each • (Line 02) `for x = 0 to 49` • (Line 03) `print("Space", x)` • (Line 06) `else // elseif road[x] <> null` • (Line 07) `print(road[x].getName())`	4	Line `07` allow `print(road[x].name)` Examiner's Comments Many candidates scored three or four marks but in general candidates found it harder to identify errors in the code than to complete code in the previous question. Some candidates didn't give the line number but rewrote the incorrect line before giving the corrected line, which was acceptable, although not ideal given the scaffolding.

Question			Answer/Indicative content	Marks	Guidance
	f		**Mark Band 3 – High level (7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of global variables and the alternatives; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level (4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of global variables and the alternatives; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level (1-3 marks)** The candidate demonstrates a basic knowledge of global variables and the alternatives with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be*	9	**AO1: Knowledge and Understanding Indicative content** • Global variables are created when the program starts, all subroutines can access/update the contents • Local variables are created in the subroutine they are created in, they are not accessible directly from any other subroutine • Local variables are removed from memory when the subroutine ends. • Local variables can be passed as parameters to a function to be updated, and then returned to override the original local variable • Local variables can be passed by reference to a subroutine to allow the content of the variable to be updated **AO2: Application** • The variables will be stored in memory throughout the whole code execution. However, the amount of data they are storing is relatively low so would not use a lot of memory. • When the game is expanded, the amount of data may increase so it could be memory intensive, especially if graphics are used in the game. • Both arrays are needed throughout the whole game so keeping them as global will make writing the code easier as the programmer will not need to keep passing them as parameters and setting return values. • Only one part of the game is being created initially and therefore the use of global variables would not affect the efficiency greatly. However, when the program expands, it could cause accuracy / testing / debugging and maintenance problems. **AO3: Evaluation** • As this is only a prototype, the use of global variables would be beneficial. • However, when the game expands, the use of global variables could create issues such as running out of memory, coupling, testing & debugging problems and maintenance problems.

Question			Answer/Indicative content	Marks	Guidance
			clear. **0 mark** No attempt to answer the question or response is not worthy of credit.		• The programmer may be best to keep the variables as local and then pass them between the different subroutines as parameters byVal and byRef. Examiner's Comments Most responses were Level 2 for definitions and some expansion to passing parameters. Very few candidates were able to go into depth about alternatives to global variables such as passing by value and passing by reference in detail or extending to issues such as scalability within a larger more extended game. Few candidates picked up on the fact that this was a more limited prototype that was likely to be expanded on which would require more consideration to be given to variable scope.
			Total	40	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>6</td><td></td><td></td><td>
<table>
<tr><th>Node</th><th>Distance travelled</th><th>Heuristic</th><th>Distance travelled + Heuristic</th><th>Previous node</th><th>Marking Guidance</th></tr>
<tr><td>A</td><td>0</td><td>90</td><td>90</td><td>N/A</td><td>1 Mark</td></tr>
<tr><td>B</td><td>20</td><td>80</td><td>100</td><td>A</td><td rowspan="2">1 Mark</td></tr>
<tr><td>C</td><td>44</td><td>43</td><td>87</td><td>A</td></tr>
<tr><td>D</td><td>128</td><td>70</td><td>198</td><td>E</td><td>1 Mark</td></tr>
<tr><td>E</td><td>66</td><td>20</td><td>86</td><td>C</td><td>1 Mark</td></tr>
<tr><td>F</td><td>81</td><td>8</td><td>89</td><td>E</td><td>1 Mark</td></tr>
<tr><td>G</td><td>90</td><td>0</td><td>90</td><td>F</td><td>1 Mark</td></tr>
</table>

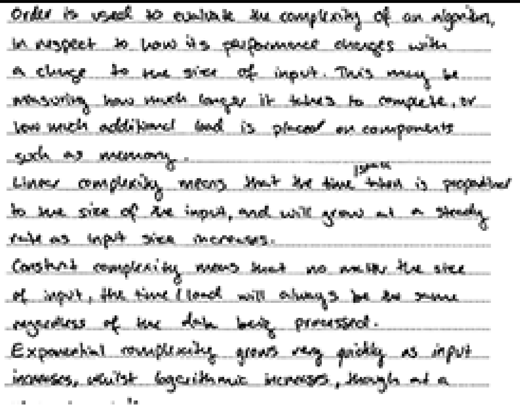
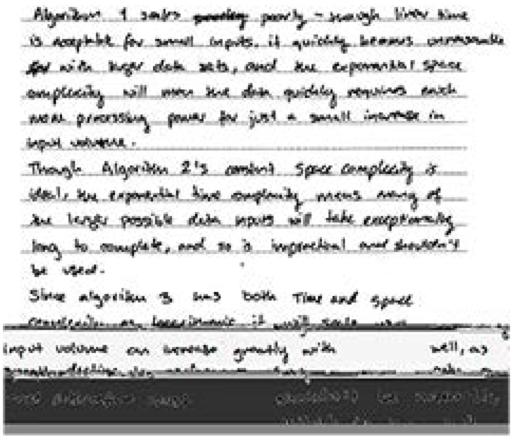
Path: A→ C →E → F → G Distance: 90 (1 Mark)</td><td>7</td><td>For Row A allow N/A, None, Null, - or a blank cell / equivalent.
<u>Examiner’s Comments</u>
Many candidates demonstrated a good understanding of the A* algorithm and most candidates achieved at least some of the marks available, with the most commonly given being the final path (often through inspection) and the first row of the table.
A few candidates treated the algorithm as if it were Dijkstra’s and explored all possible paths/routes rather than stopping as soon as the goal node was located.</td></tr>
<tr><td></td><td></td><td></td><td><b>Total</b></td><td><b>7</b></td><td></td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
7	a		1 mark for queue elements 1 mark for both pointers  	2	Allow 20 and 15 in place but crossed out OR allow 20 and 15 in place only if `headPointer` and `tailPointer` are correct **Examiner's Comments** Most candidates either tended to score full marks or no marks. Many candidates missed labelling the pointers in their diagrams and just gave updated queue values. Some candidates kept the values 20 and 15 on the diagram and correctly moved the `headPointer` to point to index 3 and the `tailPointer` to the next free space available. A considerable number of candidates erroneously shifted all items forward, showing a lack of understanding as to how a queue is efficiently implemented.
	b		1 mark each to max 2 e.g. • A queue is a FIFO structure // elements processed in the order entered • A queue will not allow new data inserted at the front // only allows new data to be enqueued at the rear • The queue contents cannot be resequenced/sorted without rewriting	2	**Examiner's Comments** The first part of many responses was well attempted, with most candidates identifying the First In First Out (FIFO) property of a queue. Far fewer candidates were able to successfully expand on this for the second mark. A linked explanation was required such as a higher priority item cannot be inserted at the front/in the middle of the queue because only the first element can be accessed/dequeued. Some candidates just reiterated what FIFO meant instead of giving the linked explanation to a priority queue which was insufficient to gain the second mark.
			Total	4	

Question			Answer/Indicative content	Marks	Guidance
8	a		1 mark each • Compare the first element (rainbow) to search item / clouds • If it is equal to the search item return index / found • If it is not equal move to the next element • Repeat until either search item / clouds is equal // or the end of the list has been reached	3	Allow answers by example from the given dataset Examiner’s Comments Most candidates scored the majority of the marks available and demonstrated a clear understanding of a linear search. Many candidates answered by example with values from the given list.
	b		1 mark for: the data is not in order/sorted	1	Examiner’s Comments Most candidates correctly identified the requirement for data to be sorted/ordered for a binary search to work. ‘Organised’ was too vague and was not accepted.
			Total	4	

Question			Answer/Indicative content	Marks	Guidance
9			**Mark Band 3 – High level** **(7–8 marks)** The candidate demonstrates a thorough knowledge and understanding of Big O; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.* **Mark Band 2 – Mid level** **(4–6 marks)** The candidate demonstrates reasonable knowledge and understanding of Big O; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 – Low Level** **(1–3 marks)** The candidate demonstrates a basic knowledge of Big O with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be*	9	**AO1: Knowledge and Understanding Indicative content** • Big O measures the number of steps and memory usage change according to the data as the amount of data being processed increases • Linear - grows in proportion to amount of data • Exponential – the rate of increase is at the rate kn as n increases • Constant - it does not change • Logarithmic – means the rate of increase gets smaller as the amount of data increases time / time increases at a rate of logkn as n increases. **AO2: Application** • Algorithm 1 – The time taken increases as the data set grows. The space taken also significantly increases. This algorithm is not memory efficient. • Algorithm 2 – The time increases significantly and therefore this algorithm is not time efficient. The space will never change which means the amount of memory will not change as the data set grows. • Algorithm 3 – The time will grow less fast as the data set grows relative to the other algorithms. The space required will also increase, but not insurmountably. This is therefore an efficient algorithm with large data sets compared to algorithm 1 and 2 overall. **AO3: Evaluation** • Number of elements is unknown. Exponential is least appropriate because this could increase significantly and be unmanageable. • Constant is the most ideal as the time will not increase. • Algorithm 3 is more suitable because it has a logarithmic time complexity, so it increases less quickly than the other algorithms. It will be reasonable with a small amount (2 items) of data, but then when very large amounts (2 billion items) are needed it will not be significantly more.

Question			Answer/Indicative content	Marks	Guidance
			clear. **0 mark** No attempt to answer the question or response is not worthy of credit.		<u>Examiner's Comments</u> Several candidates confused logarithmic and exponential growth rates, and a significant number of candidates thought that exponential complexity was the polynomial function O(n^2) rather than O(2^n). Many candidates also defined terms such as exponential complexity as 'where it grows exponentially', such circular definitions were not given marks. Level 1 responses identified some characteristics of Big O for constant, logarithmic, linear and exponential complexity with some occasional errors or inconsistencies. Level 2 responses gave reasonable descriptions of both time and space complexity alongside accurate definitions for the complexity terms with an identification of algorithm 3 as being the best overall choice. Level 3 responses had far more evaluative AO3 analysis but were few and far between. In this scenario candidates identified that it depended on the value of n, as if n was very small, exponential time growth was not an issue, so algorithm 2 might be preferential, but as n could grow to 2 billion this would be intractable, so algorithm 3 was chosen as the most practical. The data set in question had 2 billion items at most so $\log_2 2,000,000,000 = 30$ meant space overheads in algorithm 3 were manageable with modern storage capacities. Exemplar 2

Question			Answer/Indicative content	Marks	Guidance
					 Order is used to evaluate the complexity of an algorithm, in respect to how its performance changes with a change to the size of input. This may be measuring how much longer it takes to complete, or how much additional load is placed on components such as memory. Linear complexity means that the time/space taken is proportional to the size of the input, and will grow at a steady rate as input size increases. Constant complexity means that no matter the size of input, the time / load will always be the same regardless of the data being processed. Exponential complexity grows very quickly as input increases, whilst logarithmic increases, though at a [illegible]  Algorithm 1 scales ~~poorly~~ poorly - though linear time is acceptable for small inputs, it quickly becomes unmanageable ~~for~~ with larger data sets, and the exponential space complexity will mean the data quickly requires much more processing power for just a small increase in input volume. Though Algorithm 2's constant space complexity is ideal, the exponential time complexity means many of the larger possible data inputs will take exceptionally long to complete, and so is impractical and shouldn't be used. Since algorithm 3 has both time and space complexity as logarithmic it will scale [illegible] input volume can increase greatly with [illegible] well, as [illegible] This response clearly demonstrates that a high scoring Level 3 response is possible within the space provided. The second page of the response shows clear AO3 evaluation between the merits of the three different algorithms and the scenario.
			Total	9	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>10</td><td></td><td></td><td>1 mark for final path A, D, G
1 mark for final distance 14
1 mark for each SECTION or equivalent working shown.
<table>
<tr><th>Node</th><th>Distance travelled</th><th>Previous node</th><th>Marking Guidance</th></tr>
<tr><td>A</td><td>0</td><td>- / N/A / blank / None</td><td rowspan="2">1 Mark</td></tr>
<tr><td>B</td><td>5</td><td>A</td></tr>
<tr><td>C</td><td>2</td><td>A</td><td rowspan="2">1 Mark</td></tr>
<tr><td>D</td><td>10</td><td>A</td></tr>
<tr><td>E</td><td>7</td><td>B</td><td rowspan="2">1 Mark</td></tr>
<tr><td>F</td><td>15</td><td>E</td></tr>
<tr><td>G</td><td>~~19~~ 14</td><td>~~E~~ D</td><td>1 Mark</td></tr>
</table></td><td>6</td><td>Nodes should appear in the alphabetical order given if candidates add them as the algorithm progresses but allow other orderings of the nodes.
For the last mark in the table there must be a clear indication that G 19 from E is overwritten by G 14 from D.
Allow equivalent discrete maths approach or textual description.
Check diagram for annotations / solution.
<u>Examiner's Comments</u>
Candidates answering 'by inspection' could gain 2 marks for the final path and distance, and many less successful responses thus scored 1 or 2 marks by doing so, demonstrating little real understanding of how Dijkstra's algorithm operates. A common error was the incorrect solution ACDG that showed ignorance of not continuing the search from the node with the least distance travelled that has not already been marked as visited. For full marks there had to be an indication that the first path to G (19 from E) was overwritten by G (14 from D) for the last mark in the table. Relatively few candidates demonstrated this.</td></tr>
<tr><td></td><td></td><td></td><td>Total</td><td>6</td><td></td></tr>
<tr><td>11</td><td></td><td></td><td>It doesn't point to another **node**
Indicates the end of the linked list</td><td>1</td><td><u>Examiner's Comments</u>
This question was generally well answered, with many candidates gaining marks. A generic 'no assigned value' was not given marks, as the response had to be answered in the context of the linked list, so end of list/not pointing to another node was required, rather than stating pointing to nothing.</td></tr>
<tr><td></td><td></td><td></td><td>Total</td><td>1</td><td></td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance

12			1 mark per bullet to max 6
			• function header taking parameter
			• looping appropriately e.g. until value is 0
			• dividing by 2 and finding remainder e.g. MOD
			• adding 1 or 0 correctly
			• …appending to a value to be returned // final string reversed
			• reducing value to use within loop
			• returning calculated value
			
			e.g.
			`function toBinary(denary)`
			`binaryValue=""`
			`while denary > 0`
			`temp = denary MOD 2`
			`if temp == 1 then`
			`binaryValue = "1" + binaryValue`
			`else`
			`binaryValue = "0" + binaryValue`
			`endif`
			`denary = denary DIV 2`
			`endwhile`
			`return binaryValue`
			`endfunction`
				6	Award a recursive algorithm as equivalent
					
					Examiner's Comments
					
					Very few candidates were able to produce a working function, but many gained some marks.
					
					Some candidates had little idea of the concept of a function and struggled to define one. Many omitted the definition statement or omitted the required parameter and then asked for user input instead.
					
					The standard of pseudocode/code was quite weak. Indentation of constructs was often missing or hard to follow. Mid-range marks were achieved when candidates effectively utilised MOD to determine if the remainder on division by two was odd or even, and then using DIV to find the next term in the sequence.
					
					More successful responses demonstrated an ability to problem solve, think logically, and present clear working functions.

			Total	6	

13			1 mark for each line and correction
			• Line 01
			`total = 0`
			• Line 02
			• `smallest = dataArray[0]`
			• Line 04
			`for x = 0 to 19` (accept 20)
			• Line 07
			`if dataArray[x] > largest then`
			• Line 14
			`print("Average = " + total / 20)`
				4	Do not award a mark for the line number alone without correction.
					
					Examiner's Comments
					
					Most candidates were given some marks, but fewer achieved full marks.
					
					The context of the question indicated that the numbers input were positive integers, but candidates did not always appreciate this.

			Total	4	

Question			Answer/Indicative content	Marks	Guidance
14			**Mark Band 3 – High level** **(7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of search traversals; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated*. **Mark Band 2 – Mid level** **(4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of search traversals; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence*. **Mark Band 1 – Low Level** **(1-3 marks)** The candidate demonstrates a basic knowledge of search traversals with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be*	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	**AO1: Knowledge and Understanding Indicative content** • Breadth first takes first value then visits all nodes connected to it. It then takes all nodes connected to those nodes. • Depth first goes to the left node, this becomes a new tree. It continues going to the left until a leaf. It then returns this, then goes right and repeats from the start. Follow left, follow right, take root. **AO2: Application** • Breadth will return 2005 1920 2350 1500 1985 2100 2560 (1420) 1952 2000 (2050) (2780) (2600) • Post-order / Depth will return (1420) 1500 1952 2000 1985 1920 (2050) 2100 (2600) (2780) 2560 2350 2005 **AO3: Evaluation** **Evaluations may vary and include one or more of the following points:** • Breadth is more efficient when the data searched for is closer to the root. • Depth is more efficient when data to be search for is further down. • Depth memory requirement is linear • Depth can be written recursively to aid understanding. • Breadth in general is better time complexity • In large trees depth may never return a value Candidates are not expected to know the complexities for the search traversals, however credit should be awarded if candidates choose to include these. Limit to band 2 if there is no evaluation of BFS/DFS Examiner's Comments Many candidates achieved some marks, but a few did confuse Breadth-first search (BFS) and Depth-first search (DFS), getting them the wrong way round. Most had a much clearer understanding of

Question			Answer/Indicative content	Marks	Guidance
			clear. **0 marks** No attempt to answer the question or response is not worthy of credit.		BFS than DFS and were able to score marks in Level 1 for showing how a BFS would be carried out. Fewer managed to accurately show how a DFS would work. While there was an understanding that DFS would traverse leftward to the lowest leftmost node, descriptions of back tracking were not given as many marks. A common mistake made with DFS was to output the nodes in the order they were first visited rather than the order in which they are output. Few could describe the mechanics of the algorithms with BFS using a queue and DFS using a stack. Those who did achieved the top of Level 2. Very few candidates could give exemplar uses of BFS or DFS (e.g. deleting nodes in a tree) or compare cases where they might be used (e.g. distance of target node(s) from root), with very little evaluation. This meant there were very few Level 3 responses seen.
			Total	9	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>15</td><td>a</td><td></td><td>1 mark e.g.
• Symbols are used to represent the address
• The edges represent possible connections between addresses not the actual physical routes</td><td>1</td><td>Allow other suitable answers that are in context of the problem

<u>Examiner's Comments</u>

Very few candidates were able to give suitable answers within the context of the problem. The question was asking why the graph in Fig 5 was a visualisation. Few candidates identified that it was because the letters at the nodes represented delivery addresses, while the weights on the edges represented the road distances between the addresses. Most candidates gave descriptions of visualisation in general rather than answering in context.</td></tr>
<tr><td></td><td>b</td><td>i</td><td>
<table>
<tr><th>Node</th><th>Distance travelled</th><th>Previous node</th><th>Marking Guidance</th></tr>
<tr><td>A</td><td>0 / -</td><td>N/A / -</td><td>1 Mark</td></tr>
<tr><td>B</td><td>3</td><td>A</td><td>1 Mark</td></tr>
<tr><td>C</td><td>13</td><td>E</td><td></td></tr>
<tr><td>D</td><td>10</td><td>B</td><td>1 Mark</td></tr>
<tr><td>E</td><td>6</td><td>B</td><td></td></tr>
<tr><td>F</td><td>9</td><td>E</td><td>1 Mark</td></tr>
<tr><td>G</td><td>16</td><td>F</td><td></td></tr>
<tr><td>H</td><td>~~21~~ 19</td><td>~~D~~ G</td><td>1 Mark</td></tr>
</table>

Final Path = A,B,E,F,G,H, Distance = 19 (1 Mark)</td><td>6</td><td>Order of previous nodes visited must be clear

Note that nodes in the table do not have to be given in alphabetical order by candidates

<u>Examiner's Comments</u>

Most candidates gave the final path and the total distance correctly by inspection if nothing else. All nodes are initially set to infinity, so A is updated to 0 and has a distance 0 from A as the start node and many candidates missed this. Those who gave an answer by inspection wrote ABEFGH without knowing Dijkstra's algorithm but gained some marks by giving the distances to BEFGH along the way, but distances to nodes C and D were omitted. Few candidates clearly showed that the initial calculation for the path distance to H (from D, distance 21) was later updated and overwritten with the more optimal path length from G with distance 19.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
		ii	1 mark per bullet Similarity: • Both always find the shortest route • Both are pathfinding algorithms Differences: • A* is (usually) more efficient // dijkstra's is (usually) slower • A* uses heuristics to find a solution faster // Dijkstra's does not use heuristics	2	Must contain a similarity and a difference for both marks. Examiner's Comments Those candidates who scored well in c(i) frequently gained full marks for describing the similarities and differences between Dijkstra's algorithm and the A* algorithm. The most common responses were that both give the shortest path and that A* uses heuristics.
			Total	9	

Question			Answer/Indicative content	Marks	Guidance
16			1 mark per bullet up to a maximum of 7 marks, e.g.: • Defining the `findFirst` function correctly • Suitable logic for checking the first time slot • Suitable logic for checking the next time slot… • Suitable loop to check all time slots • Suitable logic for returning the available time slot • Suitable logic for returning -1 if no time slots available • Suitable use of variable names and indentation	7 (AO3.1) (7)	

Example solution:

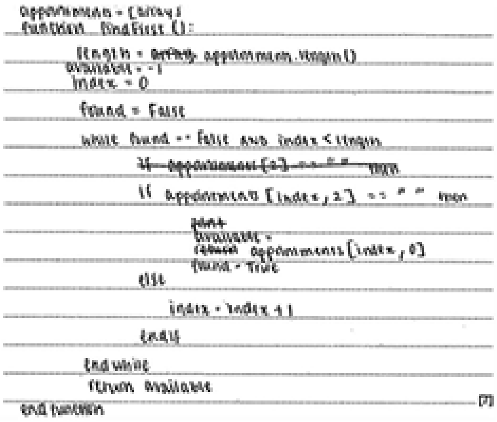
```
function findFirst()
    count = 0
    do
        found = false
        if customerID[2, count] == "" then
            found = true
        else
            count = count + 1
        endif
    until count == 10 or found == true
    if found == True
        return customerID[0,count]
    else
        return -1
    endif
endfunction
```

There are many different ways that this function could have been achieved. Therefore other alternative methods should be given credit.

Examiner's Comments

A number of candidates started by defining a procedure rather than defining a function. Indentation was not always consistent with the constructs being used. Candidates seemed to have difficulty with referencing two dimensional structures. Algorithms that require two dimensional data frequently appear on this paper and candidates need to have extensive practical programming experience solving problems using these structures.

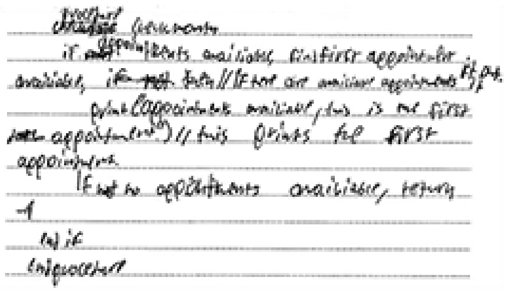
Exemplar 2



```
appointments = [array]
function findFirst ():
        length = ~~array~~ appointments.length()
      available = -1
        index = 0
        found = False
        while found == False and index < length
                ~~if appointments[2] == "" then~~
                if appointments [index, 2] == " " then
                        ~~print~~
                        available =
                        ~~return~~ appointments[index, 0]
                        found = True
                else
                    index = index + 1
                endif
        endwhile
        return available
end function                                          [7]
```

A well-structured pseudocode response

Question			Answer/Indicative content	Marks	Guidance

					that uses indentation and variable naming well. The logic of the loop to check each item in sequence for an empty entry is clear, as is the indexing into the 2-Dimensional table structure.
					Exemplar 3
					
					procedure ~~[illegible]~~ [illegible]
					if ~~[illegible]~~ appointments available, find first appointment available, if ~~not~~ then // If there are available appointments then
					print(appointments available, this is the first ~~[illegible]~~ appointment) // this prints the first appointment
					If not no appointments available, return -1
					end if
					endprocedure
					To contrast with Exemplar 2, this exemplar shows a lack of pseudocode with a response written mostly in prose English, with inconsistent indentation and a lack of clear variable naming.
					Assessment for learning
					Candidates frequently benefit from having extensive practical programming experience when answering pseudocode questions.
					Past paper questions can provide a context for problems for implementation. For this question candidates could either be asked to code a two dimensional table structure for the data in **Fig 1** and to then implement the linear search, or they code be provided with a scaffolded partially complete solution with the table defined and the stem of a function given.
					Routinely taking past paper questions that can be practically implemented is an effective way to make sure that candidates have relevant experience.

			Total	7	

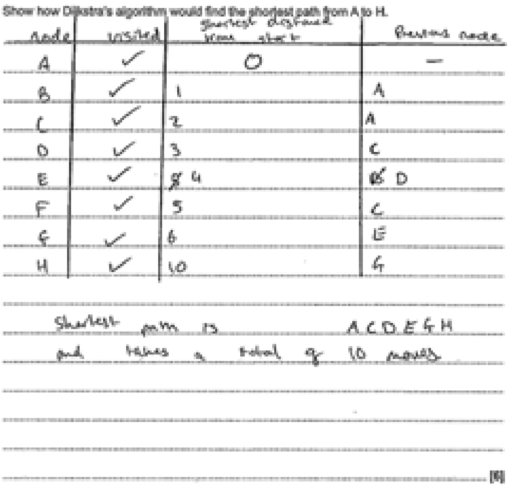
Question			Answer/Indicative content	Marks	Guidance
17			1 mark for any example e.g. • Data is not sorted • Item you are looking for is the first item in the list • Small number of items	1 AO1.2 (1)	
			Total	**1**	
18	a		1 mark per bullet, each must be applied correctly to data • Choose a pivot // identify start and end pointers • Compare each element to the pivot… // compare start and end pointers • Put items < pivot in the left sublist • Put items > pivot in the right sublist • Choose a pivot in each sublist • If start pointer is larger than end pointer… • …then swap data items around • And repeat the process until each item becomes a pivot	5 AO1.2 (2) AO2.2 (3)	
	b		1 mark per bullet to max 2 • decomposing data sets into smaller subsets • and then sorting each split subset • until each subset is sorted • and then combining the subsets to provide a solution	2 AO1.1 (1) AO2.1 (1)	
			Total	**7**	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>19</td><td></td><td>i</td><td>1 mark per bullet to max 2
e.g:
• A rule of thumb / estimate / guess
• That estimates the distance / cost from each node to the destination node
• To speed up the process of finding a solution
• …by identify which paths to follow first</td><td>2 AO1.1
(1)
AO1.2
(1)</td><td></td></tr>
<tr><td></td><td></td><td>ii</td><td><table>
<tr><th>Node</th><th>Distance travelled</th><th>Heuristic</th><th>Distance travelled + Heuristic</th><th>Previous node</th><th>MARKING GUIDANCE</th></tr>
<tr><td>A (✓)</td><td>0</td><td>90</td><td>90</td><td></td><td>1 MARK</td></tr>
<tr><td>B (✓)</td><td>∞ 21</td><td>80</td><td>101</td><td>A</td><td>1 MARK</td></tr>
<tr><td>C (✓)</td><td>∞ 42</td><td>65</td><td>107</td><td>A</td><td>1 MARK</td></tr>
<tr><td>D (✓)</td><td>∞ 42+12=54</td><td>50</td><td>104</td><td>C</td><td>1 MARK</td></tr>
<tr><td>E</td><td>∞ 21+40=61</td><td>50</td><td>111</td><td>B</td><td>1 MARK</td></tr>
<tr><td>F (✓)</td><td>∞ 42+12+23=77</td><td>30</td><td>107</td><td>D</td><td>1 MARK</td></tr>
<tr><td>G</td><td>∞ 42+12+23+33=110</td><td>0</td><td>110</td><td>F</td><td>1 MARK</td></tr>
</table>
Final path = A,C,D,F,G and Distance = 110 (1 Mark)</td><td>8 AO1.2
(3)
AO2.1
(3)
AO2.2
(2)</td><td></td></tr>
<tr><td></td><td></td><td></td><td><b>Total</b></td><td><b>10</b></td><td></td></tr>
<tr><td>20</td><td></td><td></td><td>1 mark per bullet
• Take <b>A</b> as starting node
• Visit <b>B</b>, <b>C</b> and <b>E</b>
• Visit <b>D, F, G</b> and <b>H</b>
• Visit <b>I</b> and <b>J</b></td><td>4
AO1.2
(2)
AO2.2
(2)</td><td>Allow the reverse ordering from right to left e.g. A; E, C, B; H, G, F, D; J, I</td></tr>
<tr><td></td><td></td><td></td><td><b>Total</b></td><td><b>4</b></td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>21</td><td></td><td></td><td>1 mark for final solution, max 5 for showing the stages
<ul>
<li>Mark A as the current node initially</li>
<li>Record B = 1, C = 2 (mark A as visited)</li>
<li>Record E = 5 (and mark B as visited)</li>
<li>(Record D = 3, F = 5 (and mark B as visited)</li>
<li>Change E to 4 (overriding previous value, and mark D as visited)</li>
<li>Record G = 6 (and mark E as visited)</li>
<li>…Do not change G as greater than current (mark F as visited)</li>
<li>(G as visited) H = 10 (Mark G as visited)</li>
<li>Solution: A-C-D-E-G-H path length 10</li>
</ul>
<table>
<tr><th>Node</th><th>Visited</th><th>From A</th><th>Previous Node</th><th></th></tr>
<tr><td>A</td><td>✓</td><td>0</td><td>-</td><td>1 Mark</td></tr>
<tr><td>B</td><td>✓</td><td>1</td><td>A</td><td rowspan="2">1 Mark</td></tr>
<tr><td>C</td><td>✓</td><td>2</td><td>A</td></tr>
<tr><td>D</td><td>✓</td><td>3</td><td>C</td><td rowspan="2">1 Mark</td></tr>
<tr><td>F</td><td>✓</td><td>5</td><td>C</td></tr>
<tr><td>E</td><td>✓</td><td>~~5~~ 4</td><td>~~B~~ D</td><td>2 Marks Initial visit, plus override values</td></tr>
<tr><td>G</td><td>✓</td><td>6</td><td>E</td><td>1 Mark</td></tr>
<tr><td>H</td><td></td><td>10</td><td>G</td><td>1 Mark</td></tr>
</table>
</td><td>6
AO1.2
(1)
AO2.2
(3)
AO2.2
(2)</td><td>Guidance – 1 mark only for stating the solution of A-C-D-E-G-H length 10

<u>Examiner’s Comments</u>

<b>Exemplar 2</b>


Show how Dijkstra's algorithm would find the shortest path from A to H.
<table>
<tr><th>Node</th><th>visited</th><th>shortest distance from start</th><th>Previous node</th></tr>
<tr><td>A</td><td>✓</td><td>0</td><td>–</td></tr>
<tr><td>B</td><td>✓</td><td>1</td><td>A</td></tr>
<tr><td>C</td><td>✓</td><td>2</td><td>A</td></tr>
<tr><td>D</td><td>✓</td><td>3</td><td>C</td></tr>
<tr><td>E</td><td>✓</td><td>~~5~~ 4</td><td>~~B~~ D</td></tr>
<tr><td>F</td><td>✓</td><td>5</td><td>C</td></tr>
<tr><td>G</td><td>✓</td><td>6</td><td>E</td></tr>
<tr><td>H</td><td>✓</td><td>10</td><td>G</td></tr>
</table>
Shortest path is ACDEGH and takes a total of 10 moves
[6]

Dijkstra's shortest path algorithm has appeared in previous sessions and candidates generally had a good grasp of the principles of the algorithm. Verbose text often made some responses difficult to follow. The clearest responses (as exemplified) tabulated the steps in the algorithm.</td></tr>
<tr><td></td><td></td><td></td><td><b>Total</b></td><td>6</td><td></td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>22</td><td></td><td>i</td><td>1 mark per bullet

• Correct NextPointer values
• Suitable end/null pointer

<table><tr><th>Data item</th><th>Data</th><th>NextPointer</th></tr><tr><td>0</td><td>Begonia</td><td>1</td></tr><tr><td>1</td><td>Daisy</td><td>2</td></tr><tr><td>2</td><td>Hosta</td><td>3</td></tr><tr><td>3</td><td>Lily</td><td>4</td></tr><tr><td>4</td><td>Peony</td><td>5</td></tr><tr><td>5</td><td>Rose</td><td>6</td></tr><tr><td>6</td><td>Sunflower</td><td>null</td></tr></table></td><td>2

AO2.1
(2)</td><td>Exact values only. Allow -1 for null pointer or equivalent such as Φ.
Do not allow a blank or 0.

<u>Examiner's Comments</u>

Most candidates displayed some understanding of the use of pointers in a linked list and successfully gave the correct values. Some candidates erroneously gave 0 as a null pointer value or left the NextPointer value for Sunflower empty. Null, -1 and Ø were all accepted null pointer values.</td></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark per bullet

• Lavender added in position 7
• …Hosta points to 7
• …Lavender points to 3

<table><tr><th>Data item</th><th>Data</th><th>NextPointer</th></tr><tr><td>0</td><td>Begonia</td><td>1</td></tr><tr><td>1</td><td>Daisy</td><td>2</td></tr><tr><td>2</td><td>Hosta</td><td>7</td></tr><tr><td>3</td><td>Lily</td><td>4</td></tr><tr><td>4</td><td>Peony</td><td>5</td></tr><tr><td>5</td><td>Rose</td><td>6</td></tr><tr><td>6</td><td>Sunflower</td><td>null</td></tr><tr><td>7</td><td>Lavender</td><td>3</td></tr></table></td><td>3

AO1.2
(1)
AO2.2
(2)</td><td>Do not credit answers that do not place lavender in position 7 and then update pointer positions

<u>Examiner's Comments</u>

The majority of candidates appreciated that a new entry could be made to the list by using the last available space, and then updating the relevant pointers. It was pleasing to see few candidates tried to shift the data items down, rather than updating the pointers.</td></tr>
<tr><td></td><td></td><td>iii</td><td>1 mark per bullet

• Traverse the list to the item immediately prior to the item to be removed (1)
• … which is DataItem 1 - Daisy
• Find the value of the NextPointer of the item to be removed
• … which is the NextPointer of DataItem 2 - Hosta, value 7
• Set the nextPointer of the item prior to the item to be removed to the NextPointer value of the DataItem to be removed
• … update the NextPointer of DataItem 1 - Daisy from 2 to 7 (Lavender)</td><td>4

AO1.1
(1)
AO1.2
(1)
AO2.2
(2)</td><td>Find the <i>item</i> before item to be deleted (Daisy)
Find nextPtr of item to be deleted (Hosta)
Update nextPtr of the <i>item</i> before (<i>Daisy</i>) to the nextPtr of item to be deleted (Hosta) i.e. Daisy 2 is updated to Daisy 7

Allow FT from 2b(ii/iii) if candidate has used table in fig 2.1 (e.g. Daisy would now point to Lily at position 3)

<u>Examiner's Comments</u>

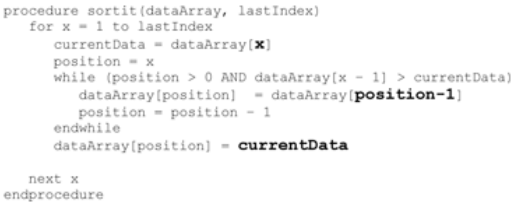
Candidates sometimes struggled to break their explanations into a clear sequence of steps, using the specific data items given. Those candidates who scored well made clear reference to the relevant data items Daisy, Hosta and Lavender. A few candidates gave annotated diagrams that explained the process particularly clearly.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance

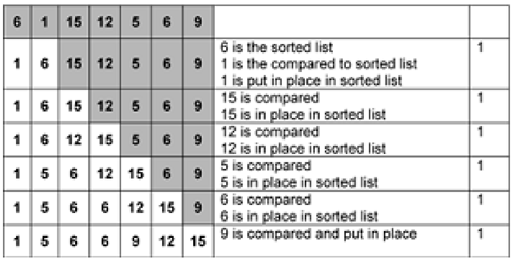
		iv	1 mark per bullet
			• Start at the `firstElement` in the list
			• Correctly looping until null pointer found / end of list
			• Outputting the data element
			• Accessing the pointer to the next element
			• Appropriate comment(s)
			e.g.
			`currentElement = firstElement`
			`while(currentElement != null)`
			`//Continue until last node`
			`print(plantList[currentElement,0])`
			`  currentElement = plantList[currentElement,1]`
			`endwhile`
				5
				AO2.1 (1)
				AO2.2 (2)
				AO3.2 (2)
					Note: Solution must utilise pointers in a linked list; it cannot use a FOR loop as the number of elements is not known and the data is not in order by index number
					Note: identifiers given in the question as `plantList` and `firstElement` should be used accurately in the solution
					Note: allow credit for answers that interpret the data structure as an array of records/structures with data/pointer fields
					Examiner's Comments
					Candidates need to be familiar with a range of data structures and the associated algorithms for performing basic functions on the data structures. Many candidates erroneously thought that the individual elements of a linked list could be indexed and accessed with a for loop. Relatively few candidates appreciated that the items in the linked list could only be traversed by following each item's next pointer until the end of the list was located.
					Many candidates continue to show limited ability to write pseudocode.

			Total	14	

Question			Answer/Indicative content	Marks	Guidance
23			1 mark per bullet to max 3 • Array size defined • A stack pointer is used to point to the top of the stack • When an item is pushed the stack pointer is incremented • When an item is popped the stack pointer is decremented	3 AO1.2 (1) AO2.1 (1) AO2.2 (1)	<u>Examiner's Comments</u> Those candidates with experience of languages other than Python appreciated that a 1D array is a static structure that needs to be declared with a given size, and that a stack pointer variable would be required. It is of concern that significant numbers of candidates have only had experience of lists and their associated methods in Python. A number of candidates also confused their descriptions with those for a queue rather than a stack.
			Total	3	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>24</td><td>a</td><td>i</td><td>1 mark for each correct item in <b>bold</b>
<pre>
procedure sortit(dataArray, lastIndex)
   for x = 1 to lastIndex
      currentData = dataArray[x]
      position = x
      while (position > 0 AND dataArray[x - 1] > currentData)
         dataArray[position] = dataArray[position-1]
         position = position - 1
      endwhile
      dataArray[position] = currentData

   next x
endprocedure
</pre></td><td>3
AO1.1
(3)</td><td>answers must be in the correct case as given
e.g. <b>current<u>D</u>ata</b>

<b>Examiner’s Comment:</b>
Many candidates found it difficult to apply the logic required to calculate the correct solution. Stronger candidates could do so even if they did not know the algorithm for insertion sort.</td></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark for contents of each row in table
<table>
<tr><td>6</td><td>1</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td></td><td></td></tr>
<tr><td>1</td><td>6</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td>6 is the sorted list
1 is the compared to sorted list
1 is put in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>6</td><td>15</td><td>12</td><td>5</td><td>6</td><td>9</td><td>15 is compared
15 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>6</td><td>12</td><td>15</td><td>5</td><td>6</td><td>9</td><td>12 is compared
12 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>12</td><td>15</td><td>6</td><td>9</td><td>5 is compared
5 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>6</td><td>12</td><td>15</td><td>9</td><td>6 is compared
6 is in place in sorted list</td><td>1</td></tr>
<tr><td>1</td><td>5</td><td>6</td><td>6</td><td>9</td><td>12</td><td>15</td><td>9 is compared and put in place</td><td>1</td></tr>
</table></td><td>6
AO2.1
(6)</td><td>… each row is dependent upon the preceding row being correct

<b>Examiner’s Comment:</b>
Some candidates confused insertion sort with other sorting algorithms, but many candidates gave good answers in diagrammatic form. Answers in diagrammatic form after each pass of the loop were often far clearer than prose descriptions. This form of answer should be encouraged.</td></tr>
<tr><td></td><td>b</td><td>i</td><td>O(n)</td><td>1
AO1.1
(1)</td><td></td></tr>
<tr><td></td><td></td><td>ii</td><td>1 mark per bullet to max 3

• The best case is for a sorted list (O(n))
• As the number of elements increases
• … the number of steps increases in a <u>linear</u> fashion</td><td>3
AO1.2
(3)</td><td>B(ii) dependent upon b(i) being correct i.e. answers for O(n) only

Accept appropriate graph for bullet points 2 and 3

<b>Examiner’s Comment:</b>
Whilst many candidates had some knowledge of 'Big O' notation fewer could apply it correctly within the context given.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
	c		**Mark Band 3 - High level** **(7–9 marks)** The candidate demonstrates a **thorough** knowledge and understanding of how bubble sort works and Big O complexity; the material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence / examples will be explicitly relevant to the explanation. *There is a well-developed line of reasoning which is clear and logically structured.* *The information presented is relevant and substantiated.* **Mark Band 2 - Mid level** **(4–6 marks)** The candidate demonstrates **reasonable** knowledge and understanding of of how bubble sort works and Big O complexity; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence / examples are for the most part implicitly relevant to the explanation. The candidate provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. *There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.* **Mark Band 1 - Low Level** **(1–3 marks)** The candidate demonstrates a **basic** knowledge of of how bubble sort works and Big O complexity with limited understanding shown; the material is basic and contains some inaccuracies. The candidates make a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	**AO1: Knowledge and Understanding Indicative content** • Description of bubble sort: ◊Starting at the beginning of the list Items are swapped with their neighbour if they are out of order. ◊Each pair of neighbours is checked in order. ◊When a swap is made a flag is set. ◊If at the end of the list the flag has been set the flag is unset and the algorithm starts from the beginning of the list again. ◊When the algorithm gets to the end of the list and the flag is unset the list is sorted and the algorithm finishes. • $O(n^2)$ denotes as the data size increases the time the list takes to sort increases in a quadratic manner. • O(1) denotes the space used is constant **AO2: Application** • As data set gets bigger, bubble sort's time gets larger at an increasing rate.. • Complexity doesn't denote the actual time but the order with which the time / space grows. • O(1) space complexity means no matter how big the data set becomes the amount of space (extra to the data itself) remains the same. • $O(n^2)$ time complexity means as n increases time increases by n^2 / if n doubles the time taken is squared. • Bubble sort can be tweaked with improvements (e.g. checking one less item per iteration and alternating sorting directions). • These optimisations don't change the complexity. IT will run a little quicker on smaller sets but time taken increases rapidly with data size. • When choosing an algorithm we may also want to take into account the average and best case scenarios. (in this case they are also the same for both algorithms.)

Question			Answer/Indicative content	Marks	Guidance
			discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. *The information is basic and comunicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.* **0 marks** No attempt to answer the question or response is not worthy of credit.		AO3: Evaluation - The algorithms may have the same time complexity but this does not mean they take the same time to execute on the same data set. - Insertion sort generally performs quicker than bubble sort and is therefore preferable. (Neither scale well however.) - Both algorithms have a space complexity of O(1). This is because both algorithms are inplace (i.e. all sorting takes place within the actual data). - Both have a time complexity of $O(n^2)$ as a consequence of their nested loops. (NB last two points are only likely to appear in the very highest mark answers.) **Examiner's Comment:** Most candidates achieved some credit, especially for a description of the bubble sort. Fewer candidates could compare the relative merits of both bubble and insertion sort in terms of the best / average / worse case.
			Total	22	

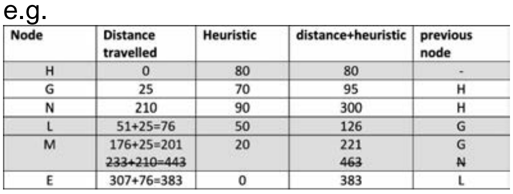
Question			Answer/Indicative content	Marks	Guidance
25		i	O(n)	1 AO1.1 (1)	Examiner's Comment: Most candidates scored well for this section.
		ii	1 mark per bullet to max 2 • 20(ms) • … showing working	2 AO1.2 (1) AO2.1 (1)	**Examiner's Comment:** Most candidates scored well for this section.
			Total	3	
26			1 mark per bullet, to max 2, e.g. • Orders can be processed in the order they are in the queue • Orders can be inserted at any place in the list e.g. high priority item inserted earlier in the list • Orders can be deleted from any position in the list once they are complete • List is dynamic… • … to allow orders to be added / deleted	2 AO1.2 (1) AO2.1 (1)	**Examiner's Comment:** Many candidates struggled to apply the context given to computer science concepts and hence answer with the relevant properties of a linked list that would be relevant in context.
			Total	2	

Question			Answer/Indicative content	Marks	Guidance
27		i	1 mark per bullet • How the times scales as data size increases • O(n) = linear complexity • Increases at the same rate as the number of data items increases	3	
		ii	1 mark each A = exponential B = logarithmic	2	
		iii	1 mark for recommendation, max 3 for explanation • Recommend: Solution B Justification • A's space does not scale well when increased in number of items • B's space scales well / does not increase significantly with number of items • As n increases at some point a will require significantly more space than B • Both have same time complexity so need to look at space	4	
			Total	9	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>28</td><td>a</td><td></td><td>1 mark per bullet

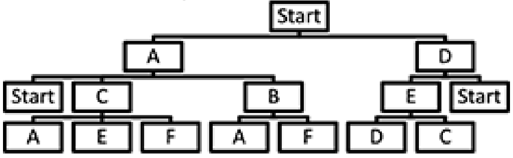
• Weighted/Undirected
• Graph</td><td>2</td><td></td></tr>
<tr><td></td><td>b</td><td>i</td><td>E.g. Weighting/cost based on estimated distance from final node</td><td>1</td><td></td></tr>
<tr><td></td><td></td><td>ii</td><td>Used to speed up process of finding solution</td><td>1</td><td></td></tr>
<tr><td></td><td></td><td>iii</td><td>1 mark per bullet, max 7 for calculations/explanation, max 1 for correct final path

• Visiting H with correct heuristic
• Visiting G and N from H
◦ Calculating correct distance+heuristic for G and N
• Identifying G as the smallest value
• Visiting L and M from G
◦ Calculating distance+heuristic for L and M
• Identifying L as the smallest value
• Visiting E
◦ Calculating distance+heuristic for E
• Final path: H-G-L-E

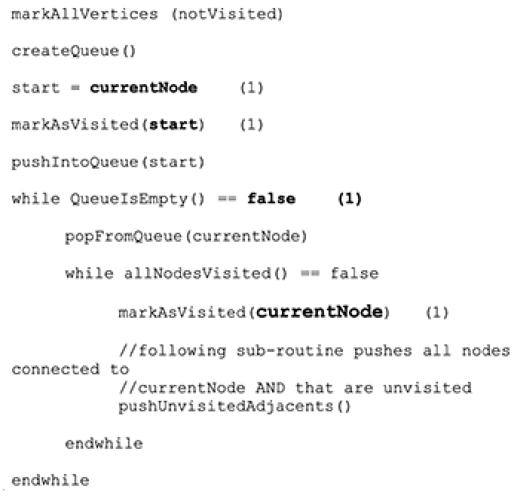
e.g.

<table>
<tr><th>Node</th><th>Distance travelled</th><th>Heuristic</th><th>distance+heuristic</th><th>previous node</th></tr>
<tr><td>H</td><td>0</td><td>80</td><td>80</td><td>-</td></tr>
<tr><td>G</td><td>25</td><td>70</td><td>95</td><td>H</td></tr>
<tr><td>N</td><td>210</td><td>90</td><td>300</td><td>H</td></tr>
<tr><td>L</td><td>51+25=76</td><td>50</td><td>126</td><td>G</td></tr>
<tr><td>M</td><td>176+25=201
~~233+210=443~~</td><td>20</td><td>221
~~463~~</td><td>G
~~N~~</td></tr>
<tr><td>E</td><td>307+76=383</td><td>0</td><td>383</td><td>L</td></tr>
</table></td><td>8</td><td></td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance
		iv	1 mark for decision, 2 marks for effect e.g. Decision: • Choosing which node to take next • The shortest distance+heuristic is taken Effect: • All adjoining nodes from this new node are taken • Other nodes are compared again in future checks • Assumed that this node is a shorter distance • Adjoining nodes may not be shortest path … • ….may need to backtrack to previous nodes	3	
			Total	**15**	

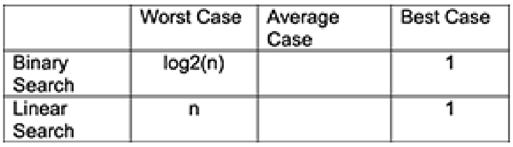
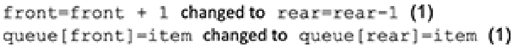
Question			Answer/Indicative content	Marks	Guidance
29		i	Class diagram	1	cao Examiner's Comments An easy question which most correctly answered. There was only one possible answer.
		ii	Method / operation	1	Examiner's Comments This was also well answered, there were two possible answers, both equally correct.
		iii	Shows inheritance Eg Nurse is a subclass of Staff / Staff is a superclass of Nurse	2	Accept any relevant example from this diagram Examiner's Comments This question asked for examples and was one of the few questions where just about every candidate managed to give an example.
		iv	Jones is the value of an attribute / not defined as an object in Staff	1	Examiner's Comments This was a stretch and challenge type question and the expected target candidates were able to answer correctly.
		v	Staff includes setSalary( ) Nurse is a subclass of Staff SeniorNurse is a subclass of Nurse SeniorNurse inherits setSalary( ) from Staff via Nurse *	3	*Final statement listed is equivalent of the 2 previous statements Examiner's Comments Most candidates were able to pick up at least two of the three marks for this, showing a good understanding of class diagrams.
			Total	**8**	

Question			Answer/Indicative content	Marks	Guidance
30	a		1 mark per bullet to max 2 e.g. • Places have been replaced with variables … (1) • … e.g. a place has been replaced with A (1) • Irrelevant information has been removed … (1) • … e.g. only the routes and places are shown (1) • Time is given as a numeric value (1) … • … e.g. 1 rather than 1 hour, or 1 minute (1) • Relative geographic location may not be accurate (1) • … e.g. positions of the towns may not be proportional to actual distance (1)	2	
	b	i	1 mark for completing A, E, F below C 1 mark for completing A, F below B 1 mark for completed D, C below E  Start A D Start C B E Start A E F A F D C	3	
		ii	In a binary tree a node can only have two children	1	
	c	i	1 mark per bullet to max 2 • Collection of data nodes / vertices (1) • Connections / edges are set between nodes / vertices (1) • Graph (edges) can be directional or bi-directional (1) • Graphs (edges) can be directed or undirected (1)	2	

Question			Answer/Indicative content	Marks	Guidance

		ii	1 mark each
			
			markAllVertices (notVisited)
			createQueue()
			start = **currentNode** (1)
			markAsVisited(**start**) (1)
			pushIntoQueue(start)
			while QueueIsEmpty() == **false** **(1)**
			popFromQueue(currentNode)
			while allNodesVisited() == false
			markAsVisited(**currentNode**) (1)
			//following sub-routine pushes all nodes connected to
			//currentNode AND that are unvisited
			pushUnvisitedAdjacents()
			endwhile
			endwhile
				4	

	d		Max 6. 1 mark for final solution, max 5 for showing the stages • Mark A as the current node (1) • Record B is 5, C is 3, D is 3 (1) • Mark A as visited (1) • C is shortest distance from A (1) • (C as current) Record E as 6, F as 6 (1) • Mark C as visited (1) • (D as current) Record E as 5 (1) • Mark D as visited (1) • (B as current) Record F as 7, do not update table as longer (1) • Mark B as visited (1) • (E as current) Record D as 8, do not update table as longer and E as visited (1) • A-C-F found as shortest (1)	6	
			Total	**18**	

<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>31</td><td>a</td><td></td><td>
<table>
<tr><th></th><th>Worst Case</th><th>Average Case</th><th>Best Case</th></tr>
<tr><td>Binary Search</td><td>log2(n)</td><td></td><td>1</td></tr>
<tr><td>Linear Search</td><td>n</td><td></td><td>1</td></tr>
</table></td><td>4</td><td>For 4 marks – 1 mark for each correct entry.</td></tr>
<tr><td></td><td>b</td><td></td><td>• As x (or the size of the array) increases, the rate at which y (or the number of checks needed) increases more slowly (1).
• The inverse of exponential growth (1).
• The rate of increase is a logarithmic function of the size of the array (1).</td><td>1</td><td>For 1 mark.</td></tr>
<tr><td></td><td>c</td><td></td><td>• If the array is very small. (1)
• If the item being searched for is very close to the start of the array. (1)</td><td>1</td><td>For 1 mark.</td></tr>
<tr><td></td><td></td><td></td><td><b>Total</b></td><td>6</td><td></td></tr>
<tr><td>32</td><td>a</td><td></td><td>• Item (1).</td><td>1</td><td>For 1 mark.</td></tr>
<tr><td></td><td>b</td><td>i</td><td>• The front of the queue is incremented (1), an item is placed at (near) the front of the queue (1), it should be placed at the rear (1).</td><td>2</td><td>Up to 2 marks for a valid explanation.</td></tr>
<tr><td></td><td></td><td>ii</td><td>
front=front + 1 <b>changed to</b> rear=rear-1 <b>(1)</b>
queue[front]=item <b>changed to</b> queue[rear]=item <b>(1)</b>

rear = rear – 1
queue[rear] = item</td><td>2</td><td>For 2 marks.</td></tr>
<tr><td></td><td></td><td></td><td><b>Total</b></td><td>5</td><td></td></tr>
</table>