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	Zac'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()`
	`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]

4.

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 computer game stores tasks that the player has requested. Each task has:

- an identification (ID) number e.g. **Task A**
- a real number to be processed e.g. **123456.789**
- an integer number to represent the order the tasks should be accessed e.g. **1**.

The task that needs to be processed the earliest is given the order number 1.

Two or more tasks can have the same order number. For example, two tasks can have an order number 1.

The data about each task needs to be stored. This will store the ID number, data value and order number for a task.

Explain why a record data structure is suitable for this data.

---**[2]**

(b). The tasks will be stored in a binary search tree before they are processed. They are stored in ascending order by their order number.

i. Give **two** characteristics of a binary search tree.

1 ---

---

2 ---

---**[2]**

ii. Give an advantage of storing the tasks in a binary search tree instead of a 1-dimensional array.

---**[1]**

iii. Tick (✓) **one** column in each row to identify whether each statement applies to a depth-first (post-order) tree traversal, a breadth-first tree traversal, or neither of these two traversals, when performed on a binary search tree.

Statement	Depth-first (post-order)	Breadth-first	Neither of these two traversals
All nodes at the current depth are visited before moving to the next depth			
The algorithm traverses to the end of one branch before moving to another branch			
The algorithm will make use of backtracking			
The traversal can be used to output the contents of the tree in ascending order			
The algorithm will output the root node last			

[5]

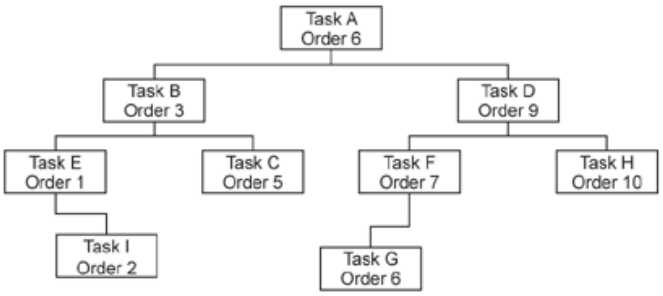
iv. The tasks currently stored in the binary search tree are shown here.

When a new task is inserted with the same order number as a pre-existing task, it is classed as having a higher order number.

For example, task G has the same order number as task A. Since task G was inserted after task A it is classed as a higher number.

Change the diagram to show the contents of the binary search tree after the following tasks are inserted in the order given:

- Task X with order number 12
- Task Y with order number 7
- Task Z with order number 11



[3]

6(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]

7(a). 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.

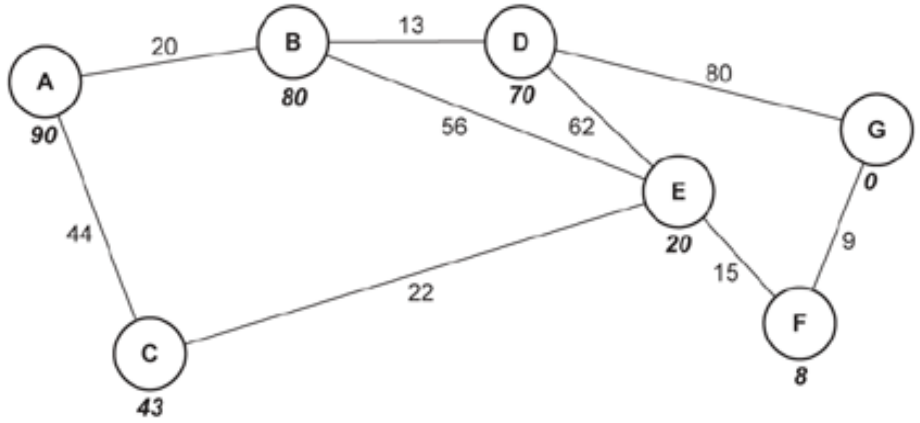


Fig. 3

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]

(b). State **four** ways that a graph data structure is different from a tree data structure.

1

2

3

4

---

[4]

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

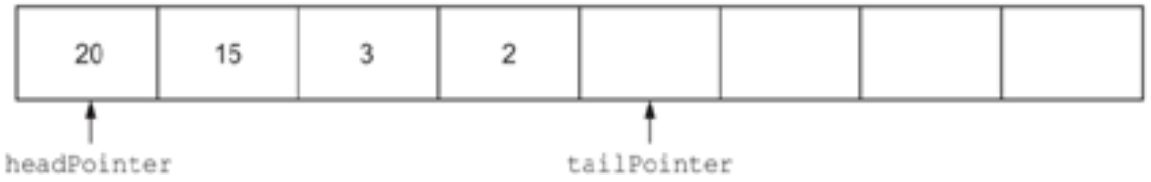


Fig. 4

State the purpose of `headPointer` and `tailPointer` in the queue shown in **Fig. 4**.

`headPointer`

`tailPointer`

---

[2]

(b). `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]

(c). 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]

9(a). The contents of a stack are stored in the 1-dimensional array called `numbers`.

`topStack` stores the index of the next free space in the stack.

The array is declared with space for 100 elements.

The function `pop()` returns the next item from the stack and updates the appropriate pointers.

Describe the steps in the function `pop()`.

[4]

(b). The function `push()` inserts its parameter called `dataValue` onto the stack and updates the appropriate pointers.

Complete the function `push()` using pseudocode or program code.

```
function push(... .....................................)
   if ................................. ! = 100 then
     numbers [.....................................] = dataValue
     topStack = topStack + ..................................
     return true
   else
     return false
   endif
endfunction
```

[4]

(c). Write an algorithm, using pseudocode or program code, to call the function `push()` with the value 15 and output a message saying "`Added`" if the value was successfully inserted onto the stack or "`Not Added`" if the stack is full.

[4]

10(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]**

(c). Show how an insertion sort will sort the given data into **ascending** alphabetical order.

“rainbow”	“moon”	“sun”	“stars”	“clouds”	“tornado”

---

[5]

11(a). A treasure game is being programmed using an object-oriented paradigm.

A class, `Treasure`, is used to store the treasure objects.

The design for the `Treasure` class, its attributes and methods is shown here.

class: Treasure
attributes: private value : integer private level : string
methods: new() function getValue() function getLevel()

i. The constructor method takes a value as an integer, e.g. 100, and a level, e.g. "bronze", as parameters and assigns these to the attributes.

Write pseudocode or program code to declare the class `Treasure`.

You should define the attributes and constructor method in your answer.

You do **not** need to write the get methods.

[5]

ii. The get method `getLevel()` will return the appropriate attribute.

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

[2]

iii. Describe the object-oriented programming technique being used in part **9(b)(ii)**.

[2]

(b). 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."

A class, `Board`, is used to store the 10 row (x coordinate) by 20 column (y coordinate) grid.

The design for the `Board` class, its attributes and methods is shown here.

class: Board
attributes: private grid : Array of Treasure
methods: new() function getGridItem(x, y) function setGridItem(x, y, treasureToInsert)

The constructor initialises each space in the grid to a treasure object with `value` as -1 and `level` as an empty string.

Complete the following pseudocode for the constructor method.

```
public procedure new()
  for row = ............................ to 9
    for column = 0 to ..........................
      ........................... [row, column] = new
Treasure(.............................,"")
    next ..........................
  next row
endprocedure
```

[5]

(c). A procedure, `guessGrid()`:

- takes a `Board` object as a parameter
- accepts the row (x) and column (y) coordinates from the user
- outputs "No treasure" if there is no treasure found at the coordinate (level is an empty string)
- if there is treasure at that coordinate, it outputs the level and the value of the treasure in an appropriate message.

Write the procedure `guessGrid()` using either pseudocode or program code.

[7]

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

1 _____

_____

2 _____

_____ **[2]**

13(a). 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)
08      endif
09   endfunction
```

Describe the key features of a recursive algorithm.

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

_____

_____

_____

_____

_____

_____ **[3]**

(b). 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]

14. Octal is a base 8 number system.

To convert a denary number to base 8:

- the denary value is divided by 8 and the remainder is stored
- the integer value after division is divided by 8 repeatedly until 0 is reached
- the remainders are then displayed in reverse order.

Example 1:

Denary 38		
	38 / 8 = 4 remainder 6	6
	4 / 8 = 0 remainder 4	4
Octal = 46		

Example 2:

Denary 57		
	57 / 8 = 7 remainder 1	1
	7 / 8 = 0 remainder 7	7
Octal = 71		

Write an algorithm to:

- take a denary value as input from the user
- convert the number to octal
- output the octal value.

You do **not** need to validate the input from the user.

Write your algorithm using pseudocode or program code.

[6]

15(a). 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]

(b). A programmer needs to use a merge sort in one part of the problem to sort items in ascending order.

i. Describe how a merge sort works.

[5]

ii. Give **one** benefit and **one** drawback of the programmer using a merge sort instead of a bubble sort.

Benefit

Drawback

[2]

16(a). A tree is one example of a data structure.

i. Give **two** characteristics of a tree data structure.

1 _____

_____

2 _____

_____

[2]

ii. The following data is entered into a binary search tree.

22 13 5 36 55 14 8

Draw the binary search tree when the given data is entered in the order given.

[4]

iii. Describe how a **leaf node** is deleted from a binary search tree.

_____

_____

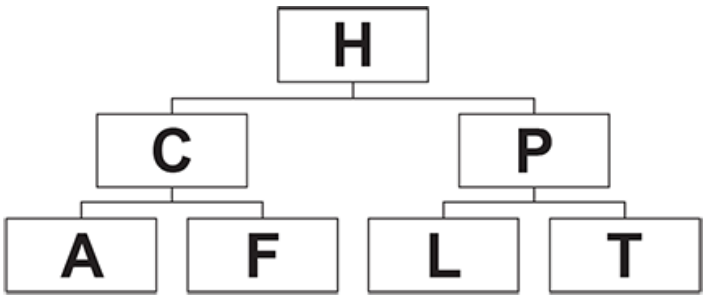
_____

[2]

iv. Describe how a binary search tree can be searched for a value.

[4]

v. Identify the order that the nodes will be visited in a **depth-first (post-order)** traversal of this binary search tree.



[4]

vi. Explain how backtracking is used in depth-first (post-order) traversals.

[2]

(b). A graph is a type of data structure.

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

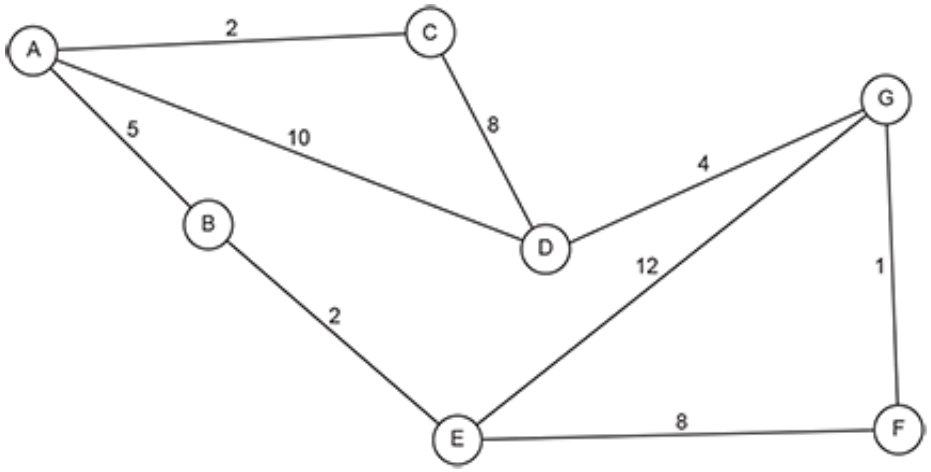


Fig. 1

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]

17(a). 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 purpose of `headPointer` and `freeListPointer` in the linked list shown in **Fig. 3**.

`headPointer` _____

_____

`freeListPointer` _____

_____ **[2]**

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

[1]

(c). A procedure outputs the data in the linked list shown in **Fig. 3** from the first item in the list, to the last item.

Give the output from the procedure.

[2]

(d). A new item needs to be added to the linked list.

Describe how a new item is added to a linked list.

[4]

(e). The function `findNode` will search the linked list and return either the position of the node that contains the data item, or -1 if the data item is not found.

The data held in a node at location x can be accessed with `linkedList[x].data`. The pointer of the node at location x can be accessed with `linkedList[x].pointer`.

For example, using the linked list shown in **Fig. 3:**
`linkedList[2].data` returns `green`.
`linkedList[2].pointer` returns `8`.

Complete the function, using pseudocode or program code.

```
function findNode(toFind, headPointer, linkedList)

  currentNode = ……

  while(currentNode != ……)

     if linkedList[currentNode]. …… == toFind then

        return currentNode

     else

        currentNode = linkedList[……].pointer

     endif

endwhile

return ……

endfunction
```

[5]

18. 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)
```

`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]**

19(a). A program stores data in a 1-dimensional array.

The program needs to search the array for a number that is input by the user.

i. Describe how a linear search will search the data in the array for a number that has been input.

[5]

ii. State why you would use a linear search rather than a binary search.

[1]

(b). Describe how an array can be used to store and access data in a stack data structure.

[4]

20.

i. The array `numbers` contains 356 numbers to be sorted by the bubble sort algorithm.

State the maximum number of passes a bubble sort would need to complete to sort 356 numbers into order.

[1]

ii. State the name of **one** other sorting algorithm.

[1]

21(a). 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]

(b). 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]

22(a). 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)
```

i. Identify the construct used on lines 01 to 03 in the algorithm.

_____

[1]

ii. Identify the construct used on lines 10 to 12 in the algorithm.

_____

[1]

(b). Identify **two** variables used in this algorithm.

1 _____

2 _____

[2]

(c). 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]**

23(a). 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]

(b). A programmer has partially developed a bubble sort algorithm in pseudocode.

This will partially sort an array of numbers called `numbers` that is passed as a parameter.

```
01 procedure bubbleSort(numbers : byRef)
02   flag = true
03   for x = 0 to numbers.length - 1
04     if numbers[x] > numbers[x + 1] then
05       holdValue = numbers[x]
06       numbers[x] = numbers[x + 1]
```

```
07          numbers[x + 1] = holdValue
08          flag = false
09        endif
10      next x
11  endprocedure
```

i. Explain why the procedure `bubbleSort` accepts the array `numbers` by reference and not by value.

[3]

ii. The programmer has used a `for` loop on line 3 in the procedure `bubbleSort`. A `for` loop is a count controlled loop.

State what is meant by the term 'count controlled loop'.

[1]

iii. State the purpose of the variable `holdValue` in the procedure `bubbleSort`.

[3]

iv. The procedure `bubbleSort` will only partially sort the array `numbers` into order.

Describe what the programmer would need to add to the algorithm to enable it to fully sort the numbers into order.

[2]

24. 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]**

25(a). 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]

(b). Trace the recursive function `calculate()` and give the final return value, when the following function call is run:

```
calculate(5)
```

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

Function call	**number**	**return**
`calculate(5)`		

[5]

(c). Give the pseudocode function call that would return 55 from the recursive function `calculate()`.

---

[1]

26(a). 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:

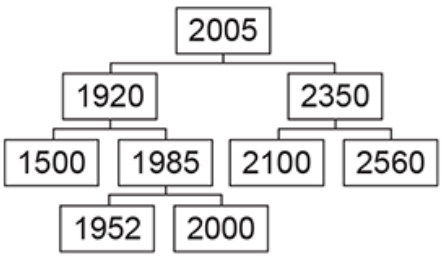


Fig. 2

Identify the root node in the binary tree shown in **Fig. 2**.

_____

_____ **[1]**

(b). Identify **two** leaf nodes in the binary tree shown in **Fig. 2**.

1 _____

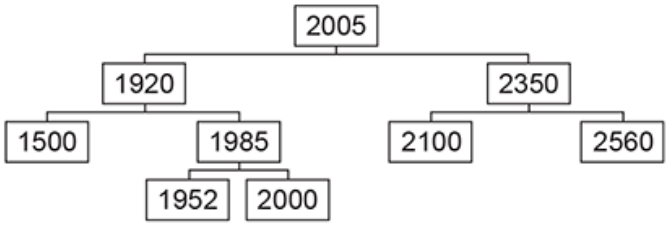
2 _____

[2]

(c). Four more students are added to the binary search tree shown in **Fig. 2** in this order:

1420 2050 2780 2600

Complete the binary search tree here by adding the new students to it.



[4]

(d). * 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]

27(a). A bubble sort will sort an array of 50 integer values called `numberArray`.

This bubble sort algorithm is written to sort `numberArray` into ascending numerical order.

Complete this bubble sort algorithm.

```
arrayLength = ................................................
tempValue = 0
do
  flag = false
  for y = 0 to arrayLength - ................................................
    if numberArray[y] > numberArray[y + 1] then
     ................................................ = numberArray[y]
     numberArray[................................................] =
numberArray[y + 1]
     numberArray[y + 1] = ................................................
     flag = true
    endif
  next y
until flag == false
```

[5]

(b). One section of `numberArray` is shown here.

2	12	1	9	3	5	15	7

A second sorting algorithm that could be used to sort this data is a merge sort.

Show how a merge sort will sort this section of the array `numberArray` into ascending numerical order.

[4]

(c). * Another sorting algorithm is insertion sort.

The number of values stored in the array `numberArray` has been reduced to 10.

Compare the use of bubble, merge and insertion sorts on the array `numberArray`.

You should include the following in your answer:

- how each algorithm works
- the Big O complexities for each algorithm
- the suitability of each algorithm for sorting the 10 values.

[12]

28(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.

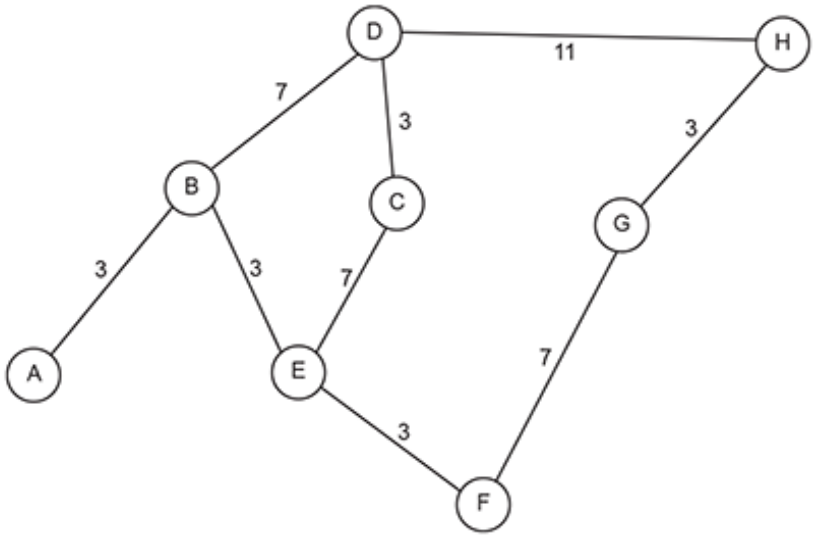


Fig. 5

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]

29. A computer program stores data in an array named `words`.

The data in the array needs to be searched for a value that the user inputs.

i. One example of a searching algorithm is a binary search.

Identify the precondition for a binary search.

[1]

ii. A second example of a searching algorithm is a linear search.

Describe how a linear search works.

[4]

30(a). 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.

When customers make an appointment they often ask for the first available booking slot.

Describe how a linear search could be used for this purpose.

[3]

(b). 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]

31(a). Trudi would like to sort an array of numbers into order.

The numbers before they have been sorted can be seen here.

89	25	75	37	45

Trudi will use a bubble sort to put these numbers into order from smallest to largest.

Show the first pass of the bubble sort. You should clearly show each comparison made.

[4]

(b). Trudi has written a procedure, `bubbleSort`.

```
01 procedure bubbleSort(numbers)
02         do
03               sorted = true
04               for count = 0 to numbers.length -2
05                     if numbers[count] > numbers[count+1] then
06                           temp = numbers[count+1]
07                           numbers[count+1] = numbers[count]
08                           numbers[count] = temp
09                           sorted = false
10                     endif
11               next count
12         until sorted == true
13 endprocedure
```

i. Identify a line in the procedure `bubbleSort` where a decision is taken.

[1]

ii. Identify the name of the parameter used in the procedure `bubbleSort`.

[1]

iii. Describe the purpose of the `temp` variable in the procedure `bubbleSort`.

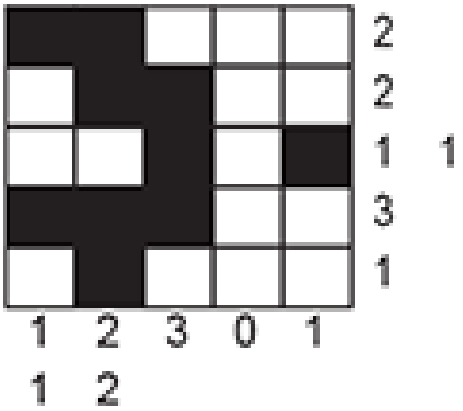
[2]

iv. Describe the purpose of the `sorted` variable in the procedure `bubbleSort`.

[2]

32(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:

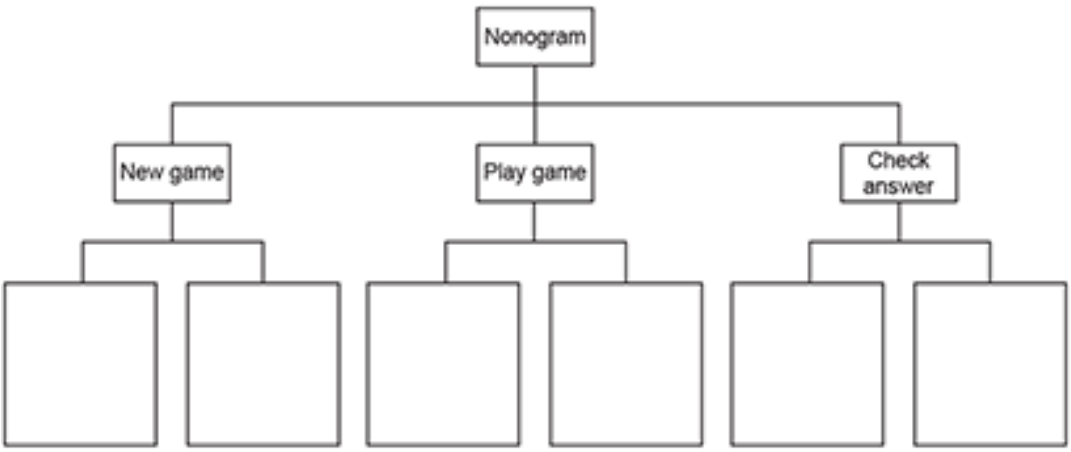
- the first column has 1 1, this means there must be two single coloured boxes in this column. There must be at least 1 white box between them.
- the first row has 2, this means there must be two consecutively coloured boxes in the row.

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()`.

```
01    function countRow(puzzle:byref, rowNum:byval)
02    count = 0
03    output = " "
04    for i = 0 to ……
05      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`.

```
01  function checkWon(puzzle)
02    for row = 0 to 4
03      for column = 0 to 4
04        if puzzle[row, column] == answerGrid[row, column] then
05          return false
06        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.

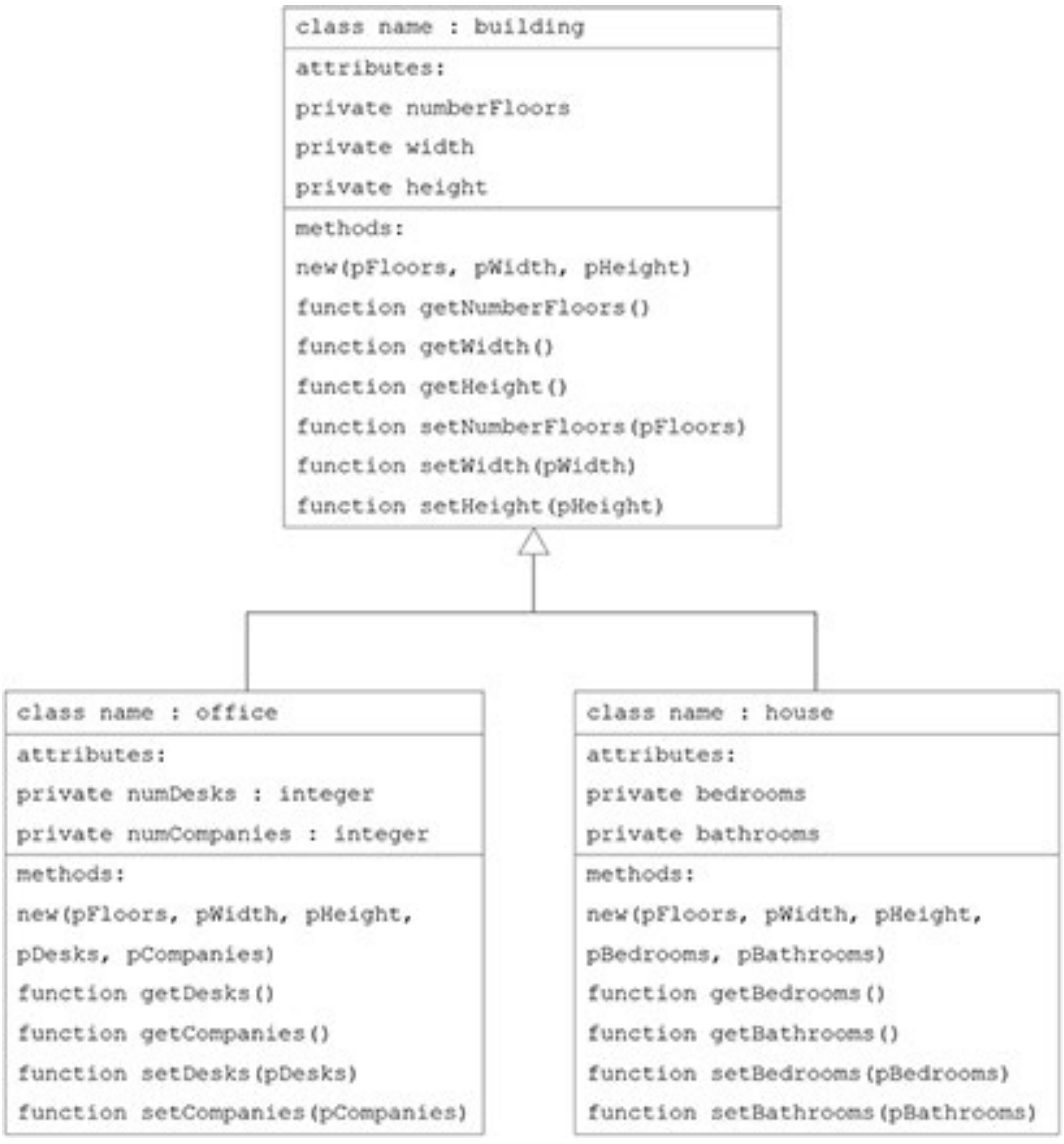
[9]

(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]

33(a). Christoff is writing a program to simulate a city using object-oriented programming. He is designing classes to store different types of buildings and their location on the road. He has created the following plan for some of the buildings:



Part of the declaration for the class `building` is shown.

Complete the pseudocode declaration by filling in the missing statements.

```
class building
    private numberFloors
    private width
    private ……
    public procedure new(pFloors, pWidth, pHeight)
       numberFloors = ……
       width = pWidth
       height = pHeight
    endprocedure
    public function getNumberFloors()
       return ……
    endfunction
    public function setNumberFloors(pFloors)
       /sets the value of numberFloors when the parameter is >= 1
       /returns true if numberFloors is successfully changed,
       /returns false otherwise
       if pFloors >= 1 then
         numberFloors = ……
         return true
       else
         return ……
       endif
    endfunction
endclass
```

[5]

(b). Write program code or pseudocode to declare the class house.
Define the attributes and constructor method in your answer. You do **not** need to write the get or set methods.

[6]

(c). Christoff develops a new class to store the houses in one road. His class design is shown:

class : houseRoad
attributes: private buildings(100) /array of class house private numberBuildings /records the number /of houses currently stored in the array /buildings
methods: new(building) function getBuilding(buildingNum) procedure newbuilding(pBuilding)

The method `newbuilding()` takes a new building as a parameter, and stores this in the next free space in the array `buildings`.

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

[4]

(d). Christoff wants to create a new `house` called `houseOne`. It has the properties: 2 floors, 8(m) width, 10(m) height, 3 bedrooms and 2 bathrooms.

The house is located on a road with the identifier `limeAvenue` of type `houseRoad`, `houseOne` is the first house in this road.

Write pseudocode or program code to declare the house `houseOne`, road `limeAvenue` and assign `houseOne` to the first array position in the road.

[4]

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

[1]

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

Describe how a quicksort would sort data into ascending order.

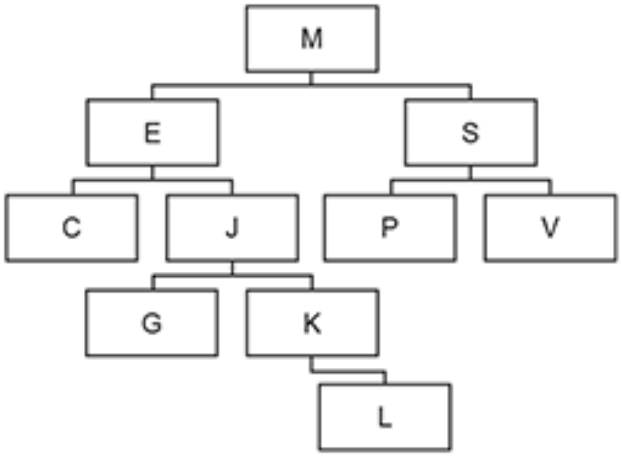
[5]

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

[2]

36. A breadth-first traversal can be performed on both a tree and a graph.

Show how a breadth-first traversal is performed on the following binary tree.



[6]

37(a). The pseudocode function `binarySearch()` performs a binary search on the array `dataArray` that is passed as a parameter. The function returns the array index of `searchValue` within the array, and `-1` if it is not in the array.

The pseudocode binary search algorithm is incomplete.

i. Complete the algorithm by filling in the missing statements.

```
function binarySearch(dataArray:byref, upperbound, lowerbound, ……)
  while true
    middle = lowerbound + ((upperbound - lowerbound) ……)
    if upperbound < lowerbound then
      return ……
    else
      if dataArray[middle] < searchValue then
        lowerbound = ……
      elseif dataArray[middle] > searchValue then
        upperbound = ……
      else
        return ……
      endif
    endif
  endwhile
endfunction
```

[6]

ii. The algorithm uses a while loop.

State a different type of loop that could be used instead of the while loop in the given algorithm.

---

[1]

(b). The tables below show possible Big O complexities for the worst-case space, best-case space and average time for search algorithms.

Tick the worst-case space complexity for a binary and linear search.

	Binary search	**Linear search**
O(log(n))		
O(1)		
O(n)		

Tick the best-case space complexity for a binary and linear search.

	Binary search	**Linear search**
O(log(n))		
O(1)		
O(n)		

Tick the average time complexity for a binary and linear search.

	Binary search	**Linear search**
O(log(n))		
O(1)		
O(n)		

[6]

38. 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.

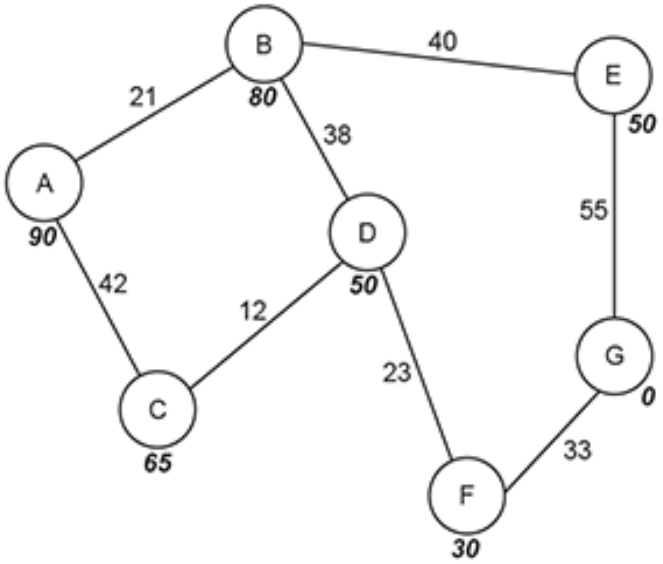


Fig. 1

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]

39(a). The following pseudocode procedure performs an insertion sort on the array parameter.

```
01 procedure insertionSort(dataArray:byRef)
02   for i = 1 to dataArray.Length - 1
03     temp = dataArray[i]
04     tempPos = i - 1
05     exit = false
06     while tempPos >= 0 and exit == false
07        if dataArray[tempPos] < temp then
08           dataArray[tempPos + 1] = dataArray[tempPos]
09           tempPos = tempPos - 1
10        else
11           exit = true
12        endif
13     endwhile
14     dataArray[tempPos + 1] = temp
15   next i
16 endprocedure
```

* Two sorting algorithms are merge sort and quick sort.

Compare the use of merge sort, quick sort and insertion sort on an array with a small number of elements, and on an array with a very large number of elements.

You should make reference to the time complexities of each algorithm using the Big O notation in your answer.

[9]

(b). Describe how a bubble sort will sort an array of 10 elements.

[6]

40(a). 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.

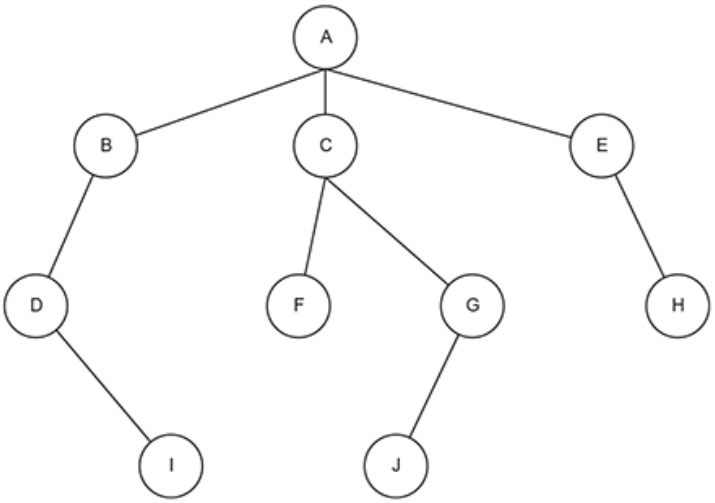


Fig. 1

Kira wants to make some changes to the data that is stored in the tree structure shown in **Fig. 1**.

i. The move represented by node 'E' needs to be deleted.

Describe the steps an algorithm will follow to delete node 'E' from the tree.

[3]

ii. The move represented by the node 'K' needs to be added. Node 'K' needs to be joined to node 'G.'

Describe the steps the algorithm will follow to add node 'K' to the right of node 'G'.

[3]

(b). Kira could have used a graph data structure to represent the moves in her game.

Give **two** similarities and **two** differences between a tree and a graph data structure.

Similarity 1 _____

_____

Similarity 2 _____

_____

Difference 1 _____

_____

Difference 2 _____

_____

[4]

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

```
01 function thisFunction(theArray, num1, num2, num3)
02   result = num1 + ((num2 - num1) DIV 2)
03   if num2 < num1 then
04     return -1
05   else
06     if theArray[result] < num3 then
07        return thisFunction(theArray, result + 1, num2, num3)
08     elseif theArray[result] > num3 then
09        return thisFunction(theArray, num1, result - 1, num3)
10     else
11        return result
12     endif
13   endif
14 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.

```
01 function thisFunction(theArray, num1, num2, num3)
02   result = num1 + ((num2 - num1) DIV 2)
03   if num2 < num1 then
04     return -1
05   else
06     if theArray[result] < num3 then
07        return thisFunction(theArray, result + 1, num2, num3)
08     elseif theArray[result] > num3 then
09        return thisFunction(theArray, num1, result - 1, num3)
10     else
11        return result
12     endif
13   endif
14 endfunction
```

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

You should write your answer using pseudocode or program code.

[6]

42(a). 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.

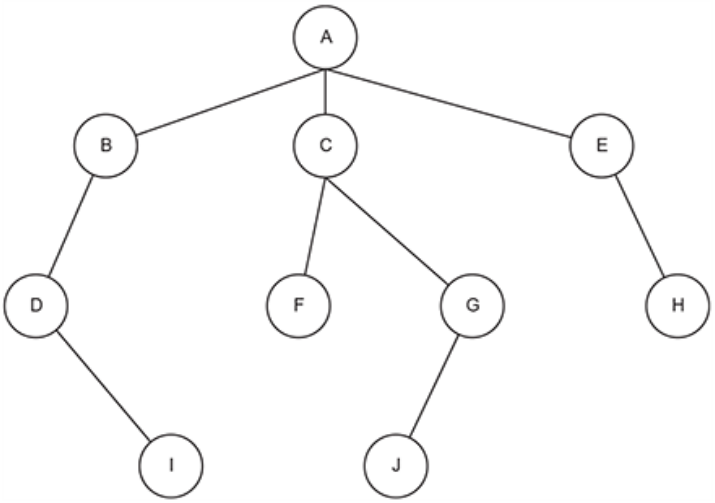


Fig. 1

State why the tree shown in **Fig. 1** is **not** an example of a binary search tree.

[1]

(b). State what type of pointers are used to store nodes I, F, J and H so they do not point to any other nodes.

[1]

(c). 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]

43(a). Poppy would like to use a bubble sort to sort 250 000 numbers into order from lowest to highest.

Currently the first five numbers before they have been sorted are:

195 584	167 147	158 187	160 125	184 236

Currently the last five numbers before they have been sorted are:

1058	19 558	1915	20 215	15

State the name of **one** other sorting algorithm that Poppy could have used.

[1]

(b). * Discuss how a bubble sort works and how efficient it will be when sorting these 250 000 items into order from lowest to highest.

[9]

(c). State the number of comparisons that will need to be made in the first pass of the bubble sort.

[1]

44(a). Oscar owns a taxi company. He would like a program to handle taxi bookings from customers.

When a customer makes a booking, they are placed into a queue data structure until a taxi driver is available.

i. Explain why Oscar uses a queue data structure rather than a stack data structure.

[4]

ii. Oscar has written a procedure, `enqueue`, to be able to add a customer number to the queue. The queue is not circular.

```
01   procedure enqueue(custNumber)
02       maxElements = 10
03       if (tail + 1) > maxElements then
04          print ("Error, queue is full")
05       else
06          head = head + 1
07          queue[head] = custNumber
08       endif
09   endprocedure
```

State the name of the parameter used in the procedure `enqueue`.

[1]

iii. The procedure `enqueue` contains an error on line 06 and line 07.

Rewrite lines 06 and 07 of the procedure `enqueue` so that the queue works correctly.

[2]

iv. Identify the logical condition in the procedure `enqueue` that affects whether a new Item can be added to the queue.

[1]

(b). Some of Oscar's customers are rated as gold. Customers who are rated as gold are given priority when they make a taxi booking. Some customers rated as gold are shown here.

Arshad	**Betty**	**Dave**	**Freddie**	**Harry**	**Jimmy**	**Kanwal**	**Lynn**	**Siad**	**Tommy**	**Will**

When a customer makes a booking, Oscar will make use of a binary search to check if they are gold rated.

Oscar would like to know if 'Tommy' is gold rated.

i. State the **three** values that will be set as the midpoints and then checked against 'Tommy' on each iteration of the binary search.

Show your working here.

Midpoint 1

Midpoint 2

Midpoint 3

[3]

ii. Oscar has 75 000 customers stored in his program.

Describe the benefit to Oscar of using binary searches in his program.

Benefit _____

_____

_____

[2]

iii. State **one** other search algorithm that Oscar could have used.

_____

[1]

iv. State the pre-condition which has been met, which meant that Oscar did not need to use the search algorithm you stated in the part above.

_____

[1]

END OF QUESTION PAPER