OCR
Oxford Cambridge and RSA

Tuesday 13 October 2020 – Afternoon

A Level Computer Science

H446/02 Algorithms and programming

Time allowed: 2 hours 30 minutes

You can use:
- a ruler (cm/mm)
- an HB pencil

Do not use:
- a calculator

8330013161

H 4 4 6 0 2

Please write clearly in black ink. **Do not write in the barcodes.**

Centre number | | | | | | Candidate number | | | | |

First name(s) _____

Last name _____

INSTRUCTIONS
- Use black ink. You can use an HB pencil, but only for graphs and diagrams.
- Write your answer to each question in the space provided. If you need extra space use the lined pages at the end of this booklet. The question numbers must be clearly shown.
- Answer **all** the questions.

INFORMATION
- The total mark for this paper is **140**.
- The marks for each question are shown in brackets **[ ]**.
- Quality of extended response will be assessed in questions marked with an asterisk (*).
- This document has **32** pages.

ADVICE
- Read each question carefully before you start your answer.

Section A

Answer **all** the questions.

1 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.

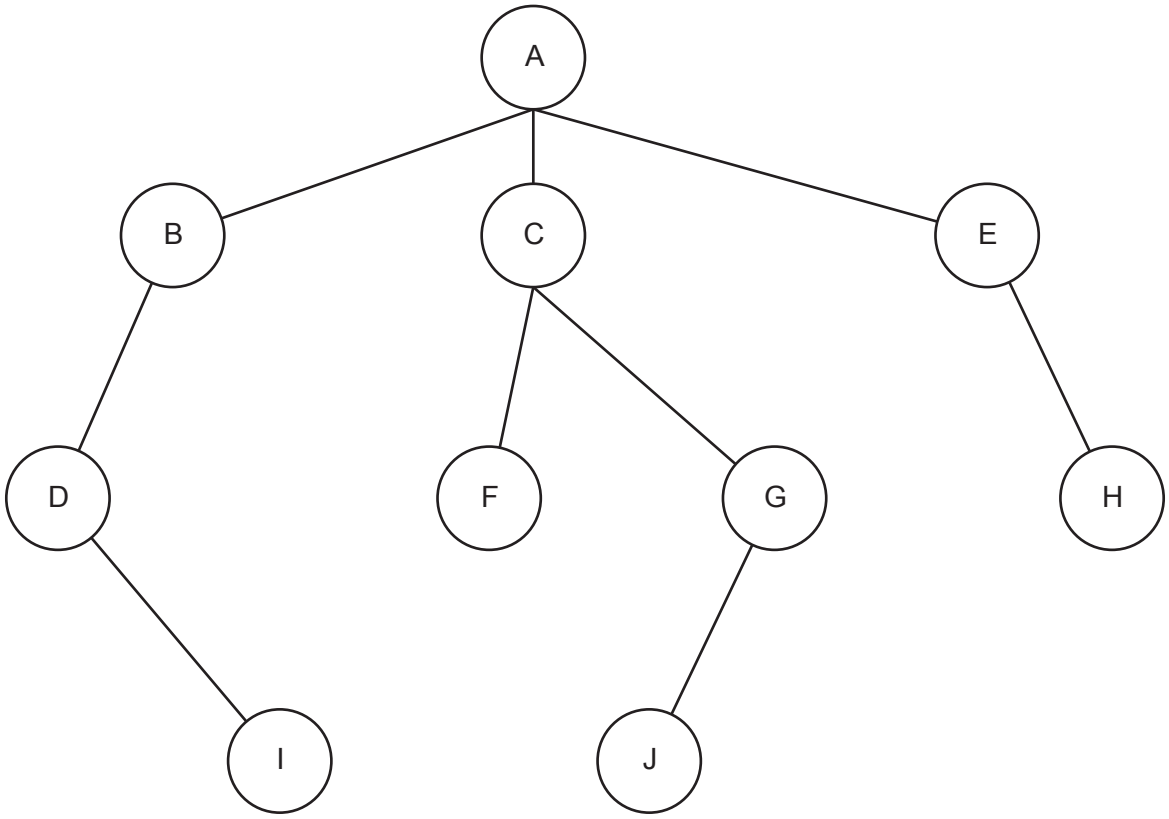

Fig. 1

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

.....

.....

.....

.....

.....

..... **[3]**

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

.....

..... **[1]**

(c) 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]**

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.

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

.....

.....

.....

.....

.....

.....

.....

..... **[4]**

(e) 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]**

(f) 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]

2 OCR-Tickets wants to sell tickets for their concerts, plays and other events online. A customer should be able to create an account and then be able to log into their account. Once logged in, customers should be able to carry out actions such as setting their preferences and purchase tickets.

OCR-Tickets have hired a software development company to create the system for them.

(a) The system requirements have a number of features that mean they are solvable by computational methods, such as decomposition.

Explain why decomposition can help the development of the program.

.....

.....

.....

.....

.....

.....

.....

..... **[4]**

(b)* Discuss the need for concurrent processing in OCR-Ticket's system and the benefits and drawbacks of using this.

.... **[9]**

3 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

(a) `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
```

11

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

You should write your answer using pseudocode or program code.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... **[6]**

4 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
```

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

.....

.....

.....

..... **[2]**

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

.....

.....

.....

.....

.....

..... **[3]**

(c)* Two other 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]**

(d) A fourth sorting algorithm is a bubble sort.

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... **[6]**

BLANK PAGE

PLEASE DO NOT WRITE ON THIS PAGE

5 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.

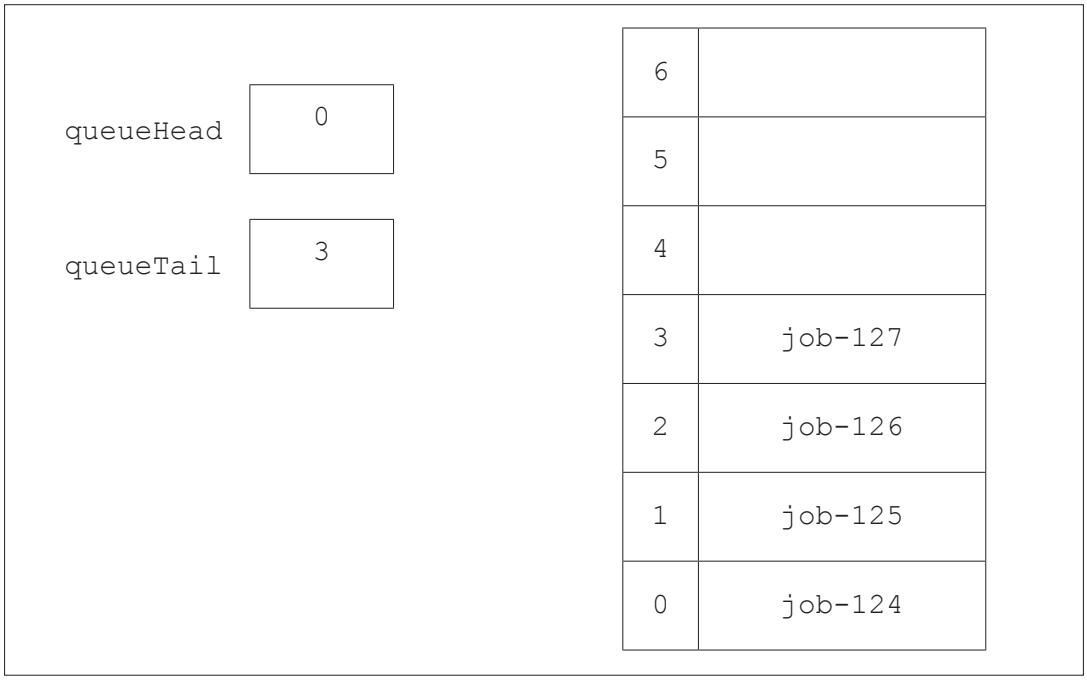

Fig. 2

(a) 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]**

Section B

Answer **all** the questions.

6 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`.

(a) **Fig. 3** shows the current contents of the linked list including the head and free list pointer values.

`headPointer` 0

`freeListPointer` 4

index	data	pointer
0	2.6	3
1	3.5	-1
2	1.8	1
3	6.9	2
4		5
5		6
6		7
7		8
8		9
9		-1

Fig. 3

(i) Describe the purpose of `freeListPointer`.

.....

.....

.....

..... **[2]**

(ii) State the purpose of `headPointer`.

.....

..... **[1]**

(iii) Show the contents of the linked list from **Fig. 3** and the pointer values when the node with data `6.9` is deleted.

`headPointer` [ ]

`freeListPointer` [ ]

index	data	pointer
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

[4]

(b) Barney wants the nodes to be stored as objects using object-oriented programming. He designs the following class.

class: node
attributes: private data : Real private pointer : Integer
methods: new (newData, newPointer) getData() getPointer() setData(newData) setPointer(newPointer)

The constructor assigns the parameters to the attributes to create an object.

(i) Write an algorithm, using pseudocode or program code, to create the class `node`, its attributes and constructor.

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... **[4]**

(ii) The class `node`, uses get methods and set methods.

Describe **one** difference between get methods and set methods.

.....

.....

.....

..... **[2]**

(c) The function `findNodePath()` takes the data item to find in the linked list as a parameter and follows the pointers to find the required node.

The function returns the array indexes of all the nodes it visits and joins this to a suitable message stating whether the data was found or not found and then returns this as one string.

Describe how the function `findNodePath()` will search for the data item and return the required message.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... **[6]**

(d) The procedure `printLinkedList()` follows the pointers to print all of the elements in the linked list.

```
01  procedure printLinkedList(headPointer)
02    tempPointer = headPointer - 1
03    dataToPrint = ""
04    if tempPointer == -1 then
05      print("List is full")
06    else
07      while linkedList[pointer].getPointer() != -1
08        dataToPrint = dataToPrint + " " + linkedList[tempPointer,0]
09         linkedList[tempPointer].getPointer() = tempPointer
10      endwhile
11     print(dataToPrint + " " + linkedList[tempPointer].getData()
12    endif
13  endprocedure
```

The procedure has a number of errors.

(i) Identify the line of each error and write 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]

(ii) Barney will use an Integrated Development Environment (IDE) to debug his program code.

Describe **three** features commonly found in IDEs that Barney could use to debug his program code.

1 .....

.....

.....

.....

2 .....

.....

.....

.....

3 .....

.....

.....

.....

[6]

(e)* 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]**

END OF QUESTION PAPER

ADDITIONAL ANSWER SPACE

If you need extra space you should use the following lined pages. The question numbers must be clearly shown in the margins.