Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME

CENTRE NUMBER

CANDIDATE NUMBER

0594894312

COMPUTER SCIENCE **9608/43**

Paper 4 Further Problem-solving and Programming Skills **May/June 2018**

2 hours

Candidates answer on the Question Paper.

No Additional Materials are required.

No calculators allowed.

READ THESE INSTRUCTIONS FIRST

Write your Centre number, candidate number and name in the spaces at the top of this page.
Write in dark blue or black pen.
You may use an HB pencil for any diagrams, graphs or rough working.
Do not use staples, paper clips, glue or correction fluid.
DO **NOT** WRITE IN ANY BARCODES.

Answer **all** questions.
No marks will be awarded for using brand names of software packages or hardware.

At the end of the examination, fasten all your work securely together.
The number of marks is given in brackets [ ] at the end of each question or part question.

The maximum number of marks is 75.

This document consists of **17** printed pages and **3** blank pages.

© UCLES 2018

CAMBRIDGE International Examinations

1 A declarative language is used to represent facts and rules about flights.

```
01 direct(edinburgh, paris).
02 direct(palma, rome).
03 direct(glasgow, palma).
04 direct(glasgow, vienna).
05 direct(glasgow, salzburg).
06
07 flies(paris, fly_jet).
08 flies(mumbai, british_air).
09 flies(palma, ciebe).
10 flies(vienna, fly_jet).
11 flies(salzburg, ciebe).
12
13 can_fly(X, Y) IF direct(X, Z) AND direct(Z, Y).
```

These clauses have the following meaning:

Clause	Explanation
01	There is a direct route from Edinburgh to Paris.
07	Fly Jet operates flights to Paris.
13	It is possible to fly from X to Y if there is a direct flight from X to Z and a direct flight from Z to Y.

(a) More facts need to be included.

There is a direct flight from London to Rome and British Air flies to Rome.

14 .....

15 .....

[2]

(b) Using the variable `Q`, the goal

```
flies(Q, fly_jet).
```

returns

```
Q = paris, vienna
```

Write the result returned by the goal

```
flies(K, ciebe).
```

K = ...[2]

(c) Use the variable `M` to write the goal to find where you can fly direct from Glasgow.

.....[2]

(d) If an airline flies to an airport, that airline also flies every direct route out of that airport.

Write a rule to represent this condition.

```
flies(Y, X)

IF .....
```

.....[3]

(e) State what the following goal returns.

```
can_fly(glasgow, rome).
```

.....[1]

2 The array `ItemList[1:20]` stores data. A **bubble sort** sorts these data.

(a) Complete the pseudocode algorithm for a bubble sort.

```
01   MaxIndex ← 20
02   NumberItems ← .....
03   FOR Outer ← 1 TO .....
04      FOR Inner ← 1 to NumberItems
05         IF ItemList[Inner] > .....
06            THEN
07               Temp ← ItemList[.....]
08               ItemList[Inner] ← ItemList[.....]
09               ItemList[Inner + 1] ← .....
10         ENDIF
11      ENDFOR
12      NumberItems ← .....
13   ENDFOR
```

[7]

(b) The algorithm in **part (a)** is inefficient.

(i) Explain why the algorithm in **part (a)** is inefficient.

.....

.....

.....

.....[2]

(ii) Explain how you would improve the efficiency of this algorithm.

.....

.....

.....

.....

.....[3]

© UCLES 2018

(c) An insertion sort is another sorting algorithm.

State **two** situations when an insertion sort is more efficient than a bubble sort. Give a reason for each.

Situation 1 .....

.....

Reason .....

.....

.....

Situation 2 .....

.....

Reason .....

.....

.....

[4]

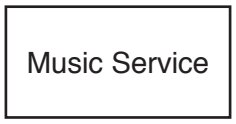
3 An internet based music streaming service provides access to an unlimited number of songs for members to play.

The following pseudocode represents the operation of the service.

```
CALL OpenAccount()
CALL OperateAccount()
CALL CloseAccount()

PROCEDURE OperateAccount()
   WHILE RequestCloseAccount() = FALSE
      IF SubscriptionDue() = TRUE
         THEN
            CALL MakePayment()
         ELSE
             CALL PlaySong()
      ENDIF
   ENDWHILE
ENDPROCEDURE
```

(a) Complete the JSP structure diagram for this music service from the pseudocode given.



[5]

(b) The service needs extending so that members can download songs to play offline.

- When a member selects a song, the service checks if the song has already been downloaded.
- If the member has already downloaded the song, the member has the option to delete or play it.
- If the member has not already downloaded the song they have the option to download or stream it.

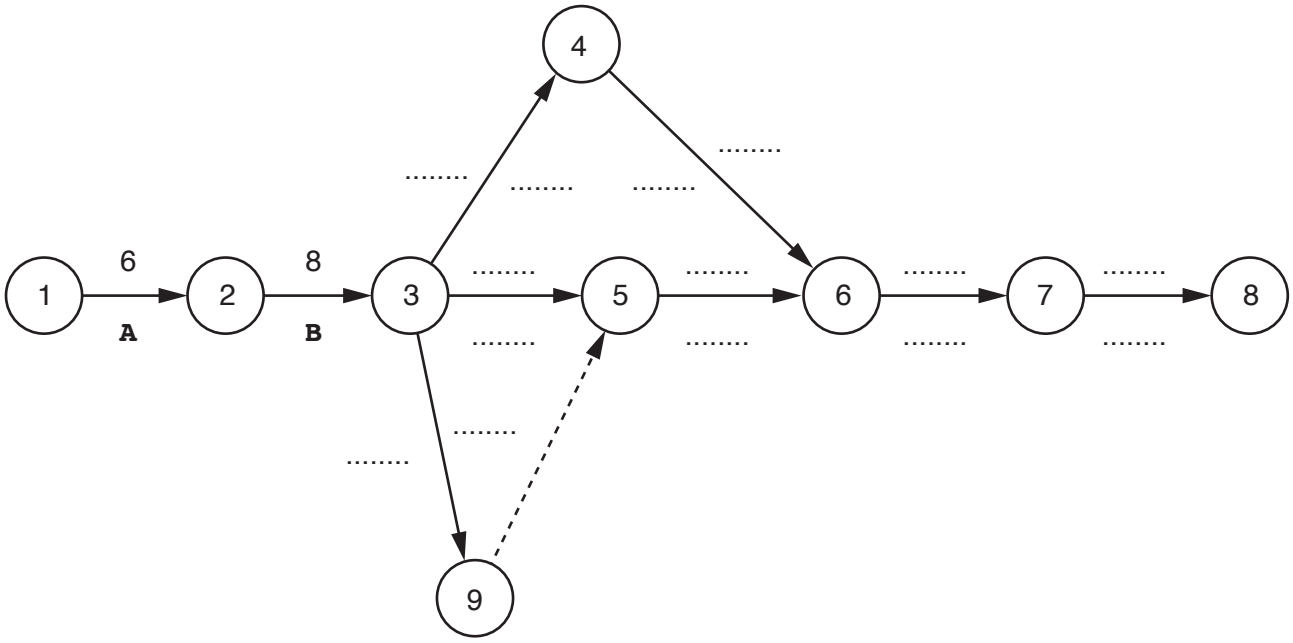
Complete the following JSP structure diagram to represent these new requirements.

[4]

4 A software company is developing a new application. The project manager has created a work breakdown structure, as shown in the following table.

	Activity	Days to complete	Predecessor
A	Gather user requirements	6	
B	Design work	8	A
C	Develop server code	4	B
D	Develop application code	5	B
E	User interface development	6	B
F	Test server code	2	C
G	Test application	2	D,E
H	Test application/server integration	5	F,G
I	Roll out mobile application	3	H

(a) Use the data in the table to complete the following Program Evaluation Review Technique (PERT) chart.



[5]

(b) Calculate the critical path (CP). State the:

activities that form the CP .....

duration of the CP .....

[2]

(c) For activity F, state the:

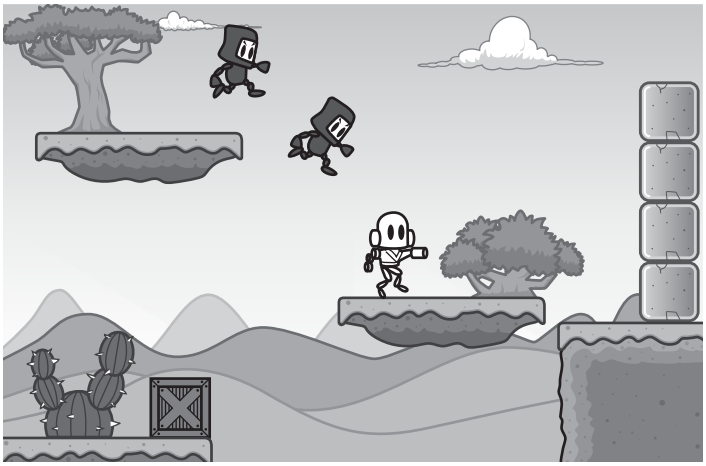
earliest start time .....

latest finish time .....

[2]

Question 5 begins on the next page.

5 A computer game is being developed using object-oriented programming. The following image is a screenshot from the game.



There are scenery elements and animated elements. The player's character is one of the animated elements.

Each game element has the attributes:

Attribute	Description	Example value
`PositionX`	The x coordinate of the game element.	`92`
`PositionY`	The y coordinate of the game element.	`106`
`Width`	The width of the game element.	`150`
`Height`	The height of the game element.	`200`
`ImageFilename`	The filename of the image file for the game element.	`GameElementFrame1.png`

Each game element has a method, `GetDetails()` that returns a string containing all the element's attributes.

The player's character is one of a number of animated elements. All animated elements have the attributes:

Attribute	Description	Example value
`AnimationFrames`	An array of `GameElement`	
`Direction`	A string giving the direction the object is travelling in.	`"Left"`
`Strength`	A value for the strength that indicates the power of the object.	`2000`
`Health`	A value for the health that indicates the health of the object.	`100`

The player's character can either move left or right, or jump.

© UCLES 2018

(a) Complete the following class diagram for the game.

You do not need to include any additional get or set methods.

GameElement
PositionX: INTEGER PositionY: INTEGER Width: INTEGER Height: INTEGER ImageFilename: STRING
Constructor() GetDetails()

AnimatedElement
AnimationFrames: ARRAY OF GameElement ....
Constructor() AdjustHealth() AdjustStrength() DisplayAnimation()

Scenery
CauseDamage: BOOLEAN DamagePoints: INTEGER
Constructor() GiveDamagePoints()

Player
....

[3]

(b) Write **program code** to define the `GameElement` class.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....[6]

© UCLES 2018

(c) The `Scenery()` class has two attributes, `CauseDamage` and `DamagePoints`.

If the attribute `CauseDamage` is `TRUE`, then the scenery element can cause damage.

The method `GiveDamagePoints()` checks whether the object can cause damage. If the object can cause damage, the method returns the integer value of the `DamagePoints` attribute.

Write **program code** for the `Scenery` class.

Programming language .....

Program code

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[6]

(d) A new scenery object, `GiftBox`, is to be created.

(i) The attributes of `GiftBox` are as follows:

Attribute	Value
`PositionX`	`150`
`PositionY`	`150`
`Width`	`50`
`Height`	`75`
`ImageFilename`	`"box.png"`
`CauseDamage`	`TRUE`
`DamagePoints`	`50`

Write **program code** to create an instance of `GiftBox`.

Programming language .....

Program code

.....

.....

.....

.....

.....

.....

.....[3]

(ii) An additional method, `GetScenery()`, returns all the attributes of the `Scenery` class.

Write **program code** for the `GetScenery()` method.

You should use the `GetDetails()` method that the `Scenery` class inherits from the `GameElement` class.

Programming language .....

Program code

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[3]

6 An Abstract Data Type (ADT) is used to create a linked list. The linked list is created as an array of records. The records are of type `ListNode`.

An example of a record of `ListNode` is shown in the following table.

Data Field	Value
`Player`	`"Alvaro"`
`Pointer`	`1`

(a) (i) Use **pseudocode** to write a definition for the record type, `ListNode`.

.....

.....

.....

.....

.....[3]

(ii) An array, `Scorers`, will hold 10 nodes of type `ListNode`. Use **pseudocode** to write an array declaration for this array. The lower bound subscript is `0`.

.....[2]

(b) The linked list stores `ListNode` records in alphabetical order of player. The last node in the linked list always has a `Pointer` value of `-1`. The position of the first node in the linked list is held in the variable `ListHead`.

After some processing, the array and variables are in the state as follows:

ListHead
0

Scorers

	Player	Pointer
0	"Alvaro"	1
1	"Antoine"	3
2	"Dimitri"	7
3	"Cristiano"	2
4	"Gareth"	5
5	"Graziano"	6
6	"Olivier"	8
7	"Erik"	4
8	"Yaya"	9
9	"Zoto"	-1

A **recursive** function traverses the linked list to search for a player.

An example of calling the function, using pseudocode, is:

```
Position ← SearchList("Gareth", ListHead)
```

© UCLES 2018

Complete the following **pseudocode** to implement the function `SearchList()`.

The function will return a value of `99` when a player is not found.

```
FUNCTION SearchList(Find : STRING, Position : INTEGER) RETURNS INTEGER
   IF Scorer[Position].Player = .....
      THEN
         RETURN .....
      ELSE
         IF Scorer[Position].Pointer <> -1
            THEN
               Position ← SearchList(Find, .....)
               RETURN .....
            ELSE
               RETURN .....
         ENDIF
   ENDIF
ENDFUNCTION
```

[5]

BLANK PAGE

© UCLES 2018

BLANK PAGE

© UCLES 2018

BLANK PAGE

Permission to reproduce items where third-party owned material protected by copyright is included has been sought and cleared where possible. Every reasonable effort has been made by the publisher (UCLES) to trace copyright holders, but if any items requiring clearance have unwittingly been included, the publisher will be pleased to make amends at the earliest possible opportunity.

To avoid the issue of disclosure of answer-related information to candidates, all copyright acknowledgements are reproduced online in the Cambridge International Examinations Copyright Acknowledgements Booklet. This is produced for each series of examinations and is freely available to download at www.cie.org.uk after the live examination series.

Cambridge International Examinations is part of the Cambridge Assessment Group. Cambridge Assessment is the brand name of University of Cambridge Local Examinations Syndicate (UCLES), which is itself a department of the University of Cambridge.

© UCLES 2018