AS

COMPUTER SCIENCE

Paper 1

Tuesday 16 May 2023 Afternoon Time allowed: 1 hour 45 minutes

Materials

For this paper you must have:

- a computer
- a printer
- appropriate software
- the Electronic Answer Document
- an electronic version and a hard copy of the Skeleton Program
- an electronic version and a hard copy of the Preliminary Material
- an electronic version of the Data Files **prog1.txt**, **prog2.txt** and **prog3.txt**.

You must **not** use a calculator.

Instructions

- Type the information required on the front of your Electronic Answer Document.
- Before the start of the examination make sure your **Centre Number**, **Candidate Name** and **Candidate Number** are shown clearly **in the footer** of every page (not the front cover) of your Electronic Answer Document.
- Enter your answers into the Electronic Answer Document.
- Answer **all** questions.
- Save your work at regular intervals.

Information

- The marks for questions are shown in brackets.
- The maximum mark for this paper is 75.
- No extra time is allowed for printing and collating.
- The question paper is divided into **three** sections.

Advice

You are advised to allocate time to each section as follows:
Section A – 20 minutes; **Section B** – 25 minutes; **Section C** – 60 minutes.

At the end of the examination

Tie together all your printed Electronic Answer Document pages and hand them to the Invigilator.

Warning

It may not be possible to issue a result for this paper if your details are not on every page of your Electronic Answer Document.

Section A

You are advised to spend no more than **20 minutes** on this section.

Enter your answers to **Section A** in your Electronic Answer Document. You **must save** this document at regular intervals.

Question **05** in this section asks you to write program code **starting from a new program/project/file**.

You are advised to save your program at regular intervals.

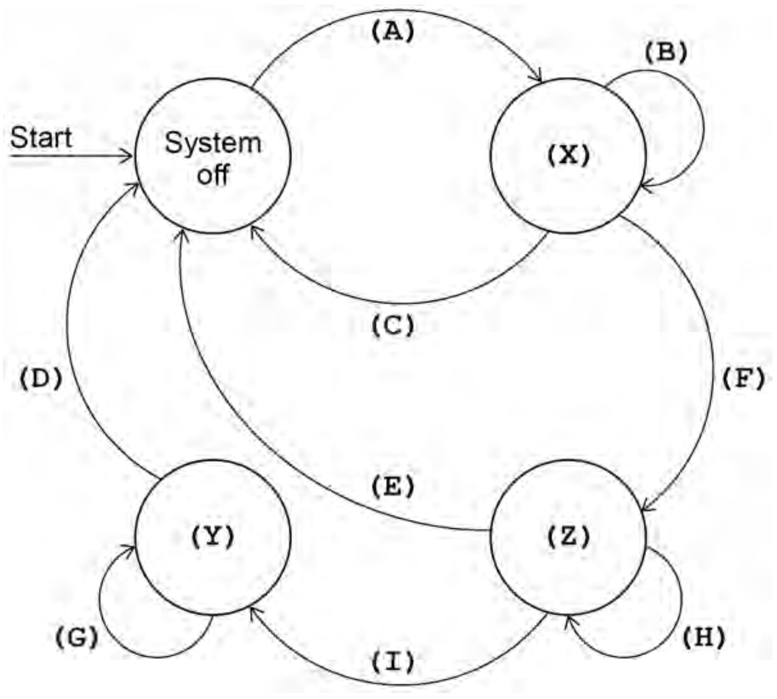
0 1 A security system uses a motion sensor, a keypad and an alarm bell.

The system operates as follows:

- the system is initially off
- when the system is switched on it goes into sensing mode
- the system can be switched off at any time by entering the correct code on the keypad
- if the system detects movement while in sensing mode it goes into alert mode
- after the system has been in alert mode for 10 seconds, it enters the alarm bell ringing mode
- if an incorrect code is entered on the keypad, once the system has been switched on, the system remains in its current mode.

Figure 1 shows a partially completed state transition diagram that represents the operation of the security system. Three of the states are labelled **(X)** to **(Z)** and events are labelled **(A)** to **(I)**.

Figure 1



Complete **Table 1** by filling in the unshaded cells with the correct labels from **Figure 1**. You should write:

- which labels, **(X)** to **(Z)**, represent which state
- which labels **(A)** to **(I)** represent which event(s).

Some of the events will be assigned more than one label.

Each label **must** only be used once.

Table 1

Event / State	**Label(s): (A) to (I), (X) to (Z)**
Alarm bell ringing mode	
Alert mode	
Detect movement	
Enter correct code	
Enter incorrect code	
Sensing mode	
Switch on	
10 second delay elapsed	

Copy the contents of all the unshaded cells in **Table 1** into your Electronic Answer Document.

[6 marks]

Turn over for the next question

0 2 The algorithm, represented using pseudo-code in **Figure 2**, describes a method to rearrange four numbers in a data structure.

Figure 2

```
Numbers[0] ← 45
Numbers[1] ← 19
Numbers[2] ← 62
Numbers[3] ← 12
FOR X ← 1 TO 3
  Y ← X - 1
  N ← Numbers[X]
  WHILE Y > -1 AND N < Numbers[Y]
    Numbers[Y + 1] ← Numbers[Y]
    Y ← Y - 1
  ENDWHILE
  Numbers[Y + 1] ← N
ENDFOR
```

Complete **Table 2** by hand-tracing the algorithm in **Figure 2**.

You may not need to use all the rows in **Table 2**.

The first row of **Table 2** has already been completed for you.

Table 2

X	Y	N	Numbers			
			[0]	[1]	[2]	[3]
			45	19	62	12

Copy the contents of all the unshaded cells in **Table 2** into your Electronic Answer Document.

[5 marks]

0 3 Describe **one** difference between a global variable and a local variable.

[2 marks]

0 4 Define the term algorithm.

[2 marks]

Turn over for the next question

0 5 The algorithm, represented using pseudo-code, in **Figure 3** outputs a series of integers. The output depends upon the value entered by the user.

Figure 3

```
OUTPUT "Enter an integer greater than 1: "
INPUT Number
X ← 2
Count ← 0
WHILE Number > 1
  Multi ← FALSE
  WHILE (Number MOD X) = 0
    IF NOT Multi THEN
      OUTPUT X
    ENDIF
    Count ← Count + 1
    Multi ← TRUE
    Number ← Number DIV X
  ENDWHILE
  X ← X + 1
ENDWHILE
OUTPUT Count
```

Table 3 lists the `MOD` and `DIV` operators for each of the available programming languages. You should refer to the row for your programming language.

Table 3

Programming language	**MOD**	**DIV**
C#	`%`	`/`
Java	`%`	`/`
Pascal	`mod`	`div`
Python	`%`	`//`
VB.NET	`Mod`	`\`

What you need to do:

Task 1
Write a program to implement the algorithm in **Figure 3**.

Task 2
Test that your program works:

- run your program, then enter the number `23`
- run your program, then enter the number `25`
- run your program, then enter the number `1260`

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

0 5 . 1 Your PROGRAM SOURCE CODE for **Task 1**.

[9 marks]

0 5 . 2 SCREEN CAPTURE(S) showing the tests described in **Task 2**.

[1 mark]

Turn over for the next section

Turn over ►

Section B

You are advised to spend no more than **25 minutes** on this section.

Enter your answers to **Section B** in your Electronic Answer Document. You **must save** this document at regular intervals.

These questions refer to the **Preliminary Material** and the **Skeleton Program**, but do **not** require any additional programming.

Refer **either** to the **Preliminary Material** issued with this question paper **or** your electronic copy.

0 6 State the identifier for:

0 6 . 1 a variable that is used to store a Boolean value.

[1 mark]

0 6 . 2 a user-defined subroutine that returns only **one** value that is **an integer**.

[1 mark]

0 6 . 3 a user-defined subroutine that returns only **one** value that is **a single character**.

[1 mark]

0 6 . 4 a user-defined subroutine that contains exception handling.

[1 mark]

0 7 The **Skeleton Program** uses a number of data structures.

0 7 . 1 State the identifier of a data structure that stores values of **more than one data type**.

[1 mark]

0 7 . 2 State the identifier of a data structure that stores values of **only one data type**.

[1 mark]

0 8 The **Skeleton Program** models a very simple processor. Many details of how a real processor works were removed because they did not need to be included for the purposes of the simulation.

State the computing term that best describes the concept of removing detail that is unnecessary from a problem when developing a solution.

[1 mark]

0 9 When the **Skeleton Program** detects an error it outputs an error code which is a number between 1 and 11.

The error codes could be replaced by more helpful error messages.

Complete **Table 4** by writing the most appropriate error code from the **Skeleton Program** next to each proposed error message.

Table 4

Proposed error message	**Error code**
Duplicate label found	
File not found	
No assembled code to run	
No source code to display	
Unknown opcode	

Copy the contents of all the unshaded cells in **Table 4** into your Electronic Answer Document.

[5 marks]

Turn over for the next question

1 0 This question refers to the subroutine `PassTwo`.

State **two** conditions that need to be met for error code 6 to be reported during the assembly process of the program.

[2 marks]

1 1 This question refers to the subroutine `ExtractOperand`.

1 1 . 1 Describe the purpose of the FOR loop **and** the IF statement **inside** it.

[2 marks]

1 1 . 2 Describe the purpose of the IF statement **after** the FOR loop.

[2 marks]

1 1 . 3 Describe why a WHILE loop might have been a better choice than a FOR loop.

[2 marks]

Section C

You are advised to spend no more than **60 minutes** on this section.

Enter your answers to **Section C** in your Electronic Answer Document. You **must save** this document at regular intervals.

These questions require you to load the **Skeleton Program** and to make programming changes to it.

1 2 This question extends the functionality of the **Skeleton Program**.

The functionality of the SKP opcode is to be changed so that it increments the value stored in the accumulator by 1

For example, if the accumulator contained 4, then executing the instruction

```
SKP
```

would change the value in the accumulator to 5

What you need to do:

Task 1
Amend the subroutine `ExecuteSKP` so that it adds 1 to the accumulator and then updates the status register if required. The `Registers` data structure should be passed as a parameter.

Task 2
Amend the call to `ExecuteSKP` in the `Execute` subroutine as required.

Task 3
Test that the changes you have made work by conducting the following test:
- run your amended **Skeleton Program**
- enter L
- load **prog2**
- enter A
- enter R

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 2 . 1 Your PROGRAM SOURCE CODE for the entire subroutine `ExecuteSKP` and the entire subroutine `Execute`.

[4 marks]

1 2 . 2 SCREEN CAPTURE(S) showing the requested test described in **Task 3**.

The SCREEN CAPTURE(S) only need to show all of Frame 0 and all of the final frame.

[1 mark]

1 3 This question adds validation to the **Skeleton Program**. The subroutine `EditSourceCode` asks the user to enter a line number. The line number must be an integer and the number of an existing line in the file containing the program.

For example, the last line (line 11) of `prog2.txt` is:

```
        FINAL:      0
```

Therefore, a valid line number for `prog2.txt` would be an integer between 1 and 11 inclusive.

What you need to do:

Task 1
Amend the subroutine `EditSourceCode` to check that the line number entered by the user is a valid existing line number. If an invalid value is entered the subroutine should output an appropriate error message. The program must not continue until a valid line number has been entered.

Task 2
Test that the changes you have made work by conducting the following test:

- run your amended **Skeleton Program**
- enter `L`
- load **prog2**
- enter `E`
- enter `Q`
- enter `22`
- enter `0`
- enter `2`

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 3 . 1 Your PROGRAM SOURCE CODE for the entire subroutine `EditSourceCode`.
[5 marks]

1 3 . 2 SCREEN CAPTURE(S) showing the requested test described in **Task 2**.
[1 mark]

1 4 This question adds a memory address check to the **Skeleton Program**.

Each time a JSR opcode is executed the return address is stored in memory. The memory location used for this could already contain an instruction or data.

If storing a return address will overwrite an instruction or data, then an appropriate error message should be output using the `ReportRunTimeError` subroutine. The original code in the `ExecuteJSR` subroutine should only be carried out if there is no error.

What you need to do:

Task 1

Amend the `ExecuteJSR` subroutine. If storing a return address would overwrite an instruction or data, the `ReportRunTimeError` subroutine should be called with an appropriate error message.

One method of completing this task would require the addition of extra parameter(s) to the `ExecuteJSR` subroutine.

Task 2

If your solution requires additional parameter(s) for the `ExecuteJSR` subroutine amend the call to `ExecuteJSR` in the `Execute` subroutine.

Task 3

Test that the changes you have made work by conducting the following test:

- run your amended **Skeleton Program**
- enter L
- load **prog3**
- enter A
- enter R

Evidence that you need to provide

Include the following evidence in your Electronic Answer Document.

1 4 . 1 Your PROGRAM SOURCE CODE for the entire subroutine `ExecuteJSR` and the entire subroutine `Execute` if you have made changes in **Task 2**.

[4 marks]

1 4 . 2 SCREEN CAPTURE(S) showing the requested test described in **Task 3**.

The SCREEN CAPTURE(S) only need to show the final frame and the contents of the stack before execution terminates.

[1 mark]

1 5 This question extends the functionality of the **Skeleton Program**. The option to edit the source code is to be extended to allow lines to be deleted or inserted within a loaded source code program.

The options within the `EditSourceCode` subroutine should now be:

`E – Edit this line`

`D – Delete the current line`

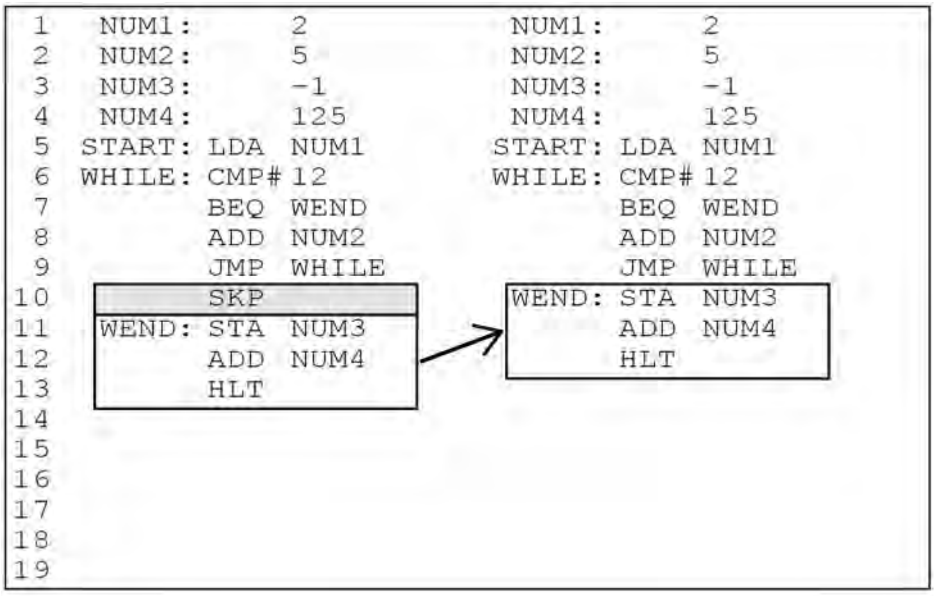
`I – Insert a new line above this line`

`C – Cancel edit`

Option D should delete the current line without changing the size of the data structure. The lines after the deleted line need to be moved within the data structure so that there is no gap in the source code.

Figure 4 shows an example where line 10 is being deleted. As a result lines 11 to 13 move. The size of the data structure does not change.

Figure 4



```
 1   NUM1:      2                 NUM1:      2
 2   NUM2:      5                 NUM2:      5
 3   NUM3:      -1                NUM3:      -1
 4   NUM4:      125               NUM4:      125
 5  START: LDA  NUM1             START: LDA  NUM1
 6  WHILE: CMP# 12               WHILE: CMP# 12
 7         BEQ  WEND                    BEQ  WEND
 8         ADD  NUM2                    ADD  NUM2
 9         JMP  WHILE                   JMP  WHILE
10         SKP                   WEND: STA  NUM3
11   WEND: STA  NUM3                    ADD  NUM4
12         ADD  NUM4                    HLT
13         HLT
14
15
16
17
18
19
```

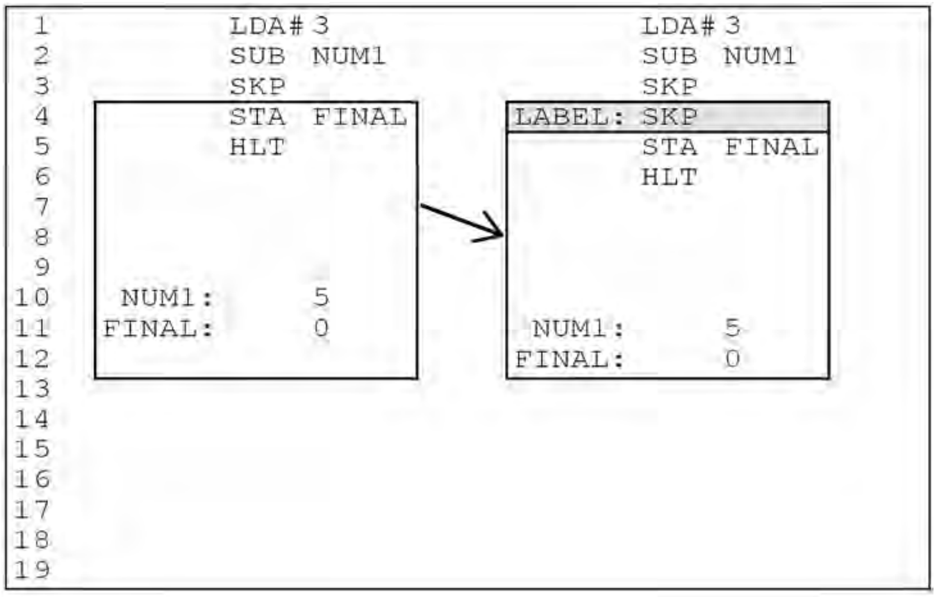
You **must write your own delete routine** and not use any built-in delete function that might be available in the programming language you are using.

Option I should allow the user to enter a new line to be inserted above the chosen line without changing the size of the data structure. The lines after the inserted line need to be moved within the data structure so they are not overwritten.

You should check that there is sufficient space in the data structure to accommodate a new line. If there is not sufficient space, an error message should be displayed.

Figure 5 shows an example where lines 4 to 11 move and a new line 4 is inserted. The size of the data structure does not change.

Figure 5



```
1           LDA# 3
2           SUB  NUM1
3           SKP
4           STA  FINAL
5           HLT
6
7
8
9
10   NUM1:       5
11  FINAL:       0
12
13
14
15
16
17
18
19
```

```
            LDA# 3
            SUB  NUM1
            SKP
     LABEL: SKP
            STA  FINAL
            HLT




     NUM1:       5
    FINAL:       0
```

You **must write your own insert routine** and not use any built-in insert function that might be available in the programming language you are using.

What you need to do:

Task 1
Amend the `EditSourceCode` subroutine to include the delete line option.

Task 2
Test that the changes you have made work by conducting the following test:

- run your amended **Skeleton Program**
- enter `L`
- load **prog1**
- enter `E`
- enter `10`
- enter `D`

Task 3
Amend the `EditSourceCode` subroutine to include the insert line option.

Task 4
Test that the changes you have made work by conducting the following test:

- run your amended **Skeleton Program**
- enter `L`
- load **prog2**
- enter `E`
- enter `4`
- enter `I`
- enter `LABEL: SKP`

Question 15 continues on the next page

Turn over ►

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 5 . 1 Your PROGRAM SOURCE CODE for the entire subroutine `EditSourceCode`.
[12 marks]

1 5 . 2 SCREEN CAPTURE(S) showing the requested test described in **Task 2**.

The SCREEN CAPTURE(S) only need to show the test from entering option E until after the edited source code has been displayed.
[1 mark]

1 5 . 3 SCREEN CAPTURE(S) showing the requested test described in **Task 4**.

The SCREEN CAPTURE(S) only need to show the test from entering option E until after the edited source code has been displayed.
[1 mark]

END OF QUESTIONS