Fundamentals of Algorithms - GCSE Computer Science Paper 1

1. No.1

0 1 **Figure 1** shows an algorithm, represented using pseudo-code, which assigns a different value to four variables.

Figure 1

```
country ← 'United States of America'
state ← 'California'
city ← 'San Francisco'
landmark ← 'Alcatraz Island'
```

0 1 . 1 Define the term **algorithm**.

[2 marks]

2. No.2

0 2 **Figure 2** shows an algorithm that uses integer division which has been represented using pseudo-code.

- Line numbers are included but are not part of the algorithm.

Figure 2

```
1   again ← True
2   WHILE again = True
3       a ← USERINPUT
4       IF a > 0 THEN
5           counter ← 0
6           WHILE a > 0
7               a ← a DIV 3
8               counter ← counter + 1
9           ENDWHILE
10      ELSE
11          again ← False
12      ENDIF
13      OUTPUT a
14  ENDWHILE
```

Integer division is the number of times one integer divides into another, with the remainder ignored.

For example:

- `14 DIV 5` evaluates to `2`
- `25 DIV 3` evaluates to `8`

0 2 . 1 Where is iteration **first** used in the algorithm in **Figure 2**?

Shade **one** lozenge.

[1 mark]

A Line number 2 ⬭

B Line number 4 ⬭

C Line number 6 ⬭

D Line number 11 ⬭

0 2 . 2 In the algorithm in **Figure 2**, what will be output when the user input is `10`?

Shade **one** lozenge.

[1 mark]

A 0 ⬭

B 1 ⬭

C 2 ⬭

D 4 ⬭

0 2 . 3 In the algorithm in **Figure 2**, what is the largest possible value of the variable `counter` when the user input is `36`?

Shade **one** lozenge.

[1 mark]

A 0 ⬭

B 2 ⬭

C 4 ⬭

D 5 ⬭

3. No.4

Figure 3 shows a program written in C# that calculates the area of a rectangle or the volume of a box from the user inputs.

Figure 3

```
public static int calculate(int width, int length,
int height) {
    if (height == -1)
    {
        return width * length;
    } else
    {
        return width * length * height;
    }
}

public static void Main() {
    int numOne, numTwo, numThree, answer;
    Console.Write("Enter width: ");
    numOne = Convert.ToInt32(Console.ReadLine());
    Console.Write("Enter length: ");
    numTwo = Convert.ToInt32(Console.ReadLine());
    Console.Write("Enter height, -1 to ignore: ");
    numThree = Convert.ToInt32(Console.ReadLine());

    answer = calculate(numOne, numTwo, numThree);

    if (numThree == -1)
    {
        Console.WriteLine($"Area {answer}");
    } else
    {
        Console.WriteLine($"Volume {answer}");
    }
}
```

0 4 . 1 Complete the trace table using the program in **Figure 3**.

[3 marks]

numOne	numTwo	numThree	Final output
5	6	-1	
10	4	0	
3	5	10	

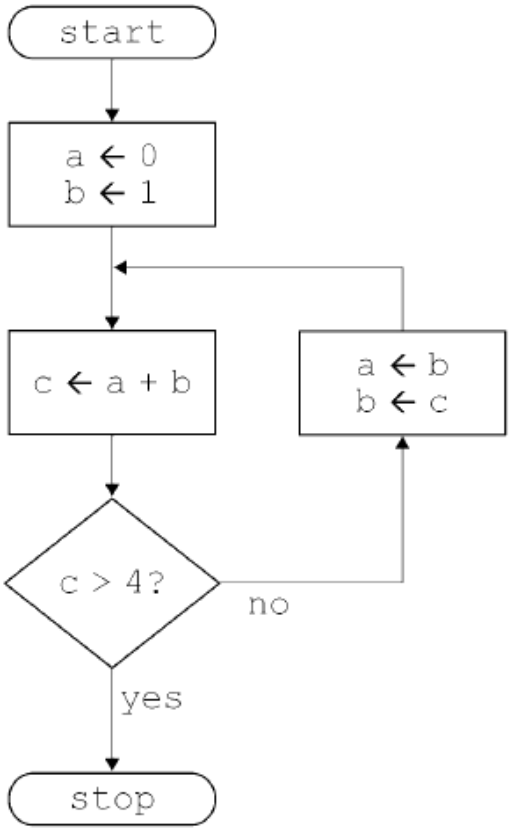
0 4 . 2 Describe **one** way that the program in **Figure 3** could be made more robust.

[1 mark]

4. No.5

Figure 4 shows an algorithm presented as a flowchart.

Figure 4



Complete the trace table for the algorithm in **Figure 4**.

You may not need to use all the rows in the table.

[3 marks]

a	b	c

5. No.6

Figure 5 shows an algorithm represented using pseudo-code.

The algorithm is for a simple authentication routine.

The pseudo-code uses a subroutine `getPassword` to check a username:

- If the username exists, the subroutine returns the password stored for that user.
- If the username does not exist, the subroutine returns an empty string.

Parts of the algorithm are missing and have been replaced with the labels L1 to L4.

Figure 5

```
login ← False
REPEAT
    username ← ''
    WHILE username = ''
        OUTPUT 'Enter username: '
        username ← L1
    ENDWHILE
    password ← ''
    WHILE password = ''
        OUTPUT 'Enter password: '
        password ← USERINPUT
    ENDWHILE
    storedPassword ← getPassword( L2 )
    IF storedPassword = L3 THEN
        OUTPUT ' L4 '
    ELSE
        IF password = storedPassword THEN
            login ← True
        ELSE
            OUTPUT 'Try again.'
        ENDIF
    ENDIF
UNTIL login = True
OUTPUT 'You are now logged in.'
```

Figure 6

-1	OUTPUT	0
username	True	SUBROUTINE
1	User not found	''
USERINPUT	password	Wrong password

State the items from **Figure 6** that should be written in place of the labels in the algorithm in **Figure 5**.

You will not need to use all the items in **Figure 6**.

[4 marks]

L1 _____

L2 _____

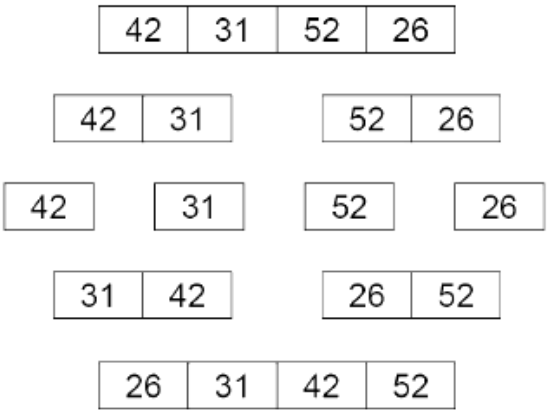
L3 _____

L4 _____

6. No.8

Figure 7 shows a merge sort being carried out on a list.

Figure 7



Explain how the merge sort algorithm works.

[4 marks]

7. No.9

Figure 8 shows an algorithm, written using pseudo-code, that uses a RECORD data structure for storing information about a film.

Each record stores four pieces of information about a film:

- film title
- certificate (eg 12A, PG)
- year the film was made
- if the film is currently being shown at a cinema.

There are records for three films and these films are stored alphabetically in an array called `filmCollection`.

The pseudo-code outputs the title of the newest of the three films.

- Part of the algorithm has been replaced by the label L1.

Figure 8

```
RECORD Film
    title : String
    certificate : String
    year : Integer
    beingShown : Boolean
ENDRECORD

hulk ← Film('Hulk', '12A', 2005, False)
ironMan ← Film('Iron Man', '12A', 2008, False)
antMan ← Film('Ant-Man', '12A', 2015, False)
filmCollection ← [antMan, hulk, ironMan]
year ← 0
position ← 0

FOR i ← 0 TO L1
    IF filmCollection[i].year > year THEN
        year ← filmCollection[i].year
        position ← i
    ENDIF
ENDFOR

OUTPUT filmCollection[position].title, ' is the
newest film'
```

0 9 . 1 How many different values can the field `beingShown` have?

Shade **one** lozenge.

[1 mark]

A 2 ⬭

B 3 ⬭

C 128 ⬭

D 256 ⬭

0 9 . 2 Which assignment statement changes the year the film *Hulk* was made to 2003?

Shade **one** lozenge.

[1 mark]

A `hulk.year ← 2003` ⬭

B `filmCollection[0].year ← 2003` ⬭

C `Film(year) ← 2003` ⬭

D `hulk(year) ← 2003` ⬭

0 9 . 3 What should the label L1 in **Figure 8** be replaced by?

Shade **one** lozenge.

[1 mark]

A `3` ⬭

B `LEN(filmCollection)` ⬭

C `LEN(filmCollection) - 1` ⬭

D `Position` ⬭

0 9 . 4 Write a pseudo-code statement that updates the `antMan` record to show that the film is currently being shown at the cinema.

[1 mark]

8. No.10

Figure 9 shows an algorithm, represented in pseudo-code, used to display students' test scores. The algorithm does not work as expected and the teacher wants to find the error.

The algorithm should display three test scores for each student:

- Natalie has results of 78, 81 and 72
- Alex has results of 27, 51 and 54
- Roshana has results of 52, 55 and 59.

- Line numbers are included but are not part of the algorithm.

Figure 9

```
1   names ← ['Natalie', 'Alex', 'Roshana']
2   scores ← [78, 81, 72, 27, 51, 54, 52, 55, 59]
3   count ← 0
4   FOR i ← 0 TO 2
5       person ← names[i]
6       OUTPUT 'Student: ', person
7       FOR j ← 0 TO 1
8           OUTPUT j + 1
9           result ← scores[i * 3 + j]
10          OUTPUT result
11          count ← count + 1
12      ENDFOR
13  ENDFOR
```

1 0 . 1 Complete the trace table for the algorithm shown in **Figure 9**.

You may not need to use all the rows in the table.

[5 marks]

count	i	person	j	result

1 0 . 2 How could the error in the algorithm in **Figure 9** be corrected?

Shade **one** lozenge.

[1 mark]

A Change line number 3 to: `count ← -1` ⬭

B Change line number 4 to: `FOR i ← 1 TO 4` ⬭

C Change line number 7 to: `FOR j ← 0 TO 2` ⬭

D Change line number 9 to: `result ← scores[j * 3 + i]` ⬭

9. No.11

Figure 10 shows part of an algorithm that has been written in pseudo-code.

There is an error in the algorithm.

The algorithm should:

- get the start year and end year from the user
- check that the start year is before the end year
- check that the start year is before 2000
- calculate the difference between the two years after a valid start year has been entered.

- Line numbers are included but are not part of the algorithm.

Figure 10

```
1    validChoice ← False
2    REPEAT
3       difference ← -1
4       OUTPUT 'Enter a start year '
5       startYear ← USERINPUT
6       OUTPUT 'Enter an end year '
7       endYear ← USERINPUT
8       IF startYear ≥ endYear THEN
9          OUTPUT 'Start year must be before end year'
10      ELSE
11         IF startYear < 2000 THEN
12            OUTPUT 'Start year must be before 2000'
13         ELSE
14            validChoice ← True
15         ENDIF
16      ENDIF
17   UNTIL validChoice = True
18   difference ← endYear - startYear
19   OUTPUT difference
```

1 1 . 1 **Table 1** shows three tests used to check the algorithm in **Figure 10**.

Complete the table to show what the values of the `validChoice` and `difference` variables would be for the given test data.

[4 marks]

Table 1

Test type	**Test data**		`validChoice`	`difference`
Normal	`startYear`	`1995`		
	`endYear`	`2010`		
Erroneous	`startYear`	`2015`		
	`endYear`	`2000`		
Boundary	`startYear`	`2000`		
	`endYear`	`2023`		

1 1 . 2 The algorithm in **Figure 10** contains a logic error on **line 11**.

Describe how the error on **line 11** can be corrected.

[1 mark]

10. No.12

1 2 . 1 **Figure 11** shows a binary search algorithm that has been programmed in C#.

The `CompareTo` method is used to compare two strings. It returns:

- `-1` if the first string is less than the second
- `0` if the first string is equal to the second
- `1` if the first string is greater than the second.

Figure 11

```
string[] animals = {"cat", "dog", "hippo", "llama",
"ox", "rat", "tiger", "wolf"};
Console.Write("What animal would you like to find? ");
string animalToFind = Console.ReadLine();
bool validAnimal = false;
int start = 0;
int finish = animals.Length - 1;
while (validAnimal == false && start <= finish) {
   int mid = (start + finish) / 2;
   if (animals[mid] == animalToFind) {
      validAnimal = true; }
   else if (animalToFind.CompareTo(animals[mid]) == 1)
   {
      start = mid + 1;
   } else {
      finish = mid - 1;
   }
}
Console.WriteLine(validAnimal);
```

Complete the trace table for the program in **Figure 11** if the user input is `wolf`

Part of the table has already been filled in.
You may not need to use all the rows in the table.

[4 marks]

animalToFind	validAnimal	start	finish	mid
wolf	False	0	7	3

1 2 . 2 **Figure 12** shows a line of C# code that creates an array of fruit names.

Figure 12

```
string[] fruits = {"banana", "apple", "orange",
                   "pear", "grape", "pineapple"};
```

Extend the program in **Figure 12**. Your answer must be written in C#.

The program should get the user to enter a word and perform a **linear** search on the array `fruits` to find if the word is in the array or not.

The program should:

- ask the user what word they would like to find
- output the message `True` if the word is found
- output the message `False` if the word is not found.

You must write your own linear search routine and **not** use any built-in search function available in C#.

You **should** use meaningful variable name(s) and C# syntax in your answer.

The answer grid below contains vertical lines to help you indent your code.

[7 marks]

```
string[] fruits = {"banana", "apple", "orange",
                   "pear", "grape", "pineapple"};
```


12.3 State why a binary search cannot be used on the array `fruits`

[1 mark]

1 2 . 4 **Figure 13** shows an algorithm, represented using pseudo-code, that should display currency names in reverse alphabetical order, starting with `yen`.

There are errors in the logic of the algorithm.

- Line numbers are included but are not part of the algorithm.

Figure 13

```
1    SUBROUTINE diffCurrencies(currencies)
        currencies ← ['baht', 'dollar', 'euro',
2                     'koruna', 'lira', 'rand',
                      'rupee', 'yen']
3       RETURN currencies[x]
4    ENDSUBROUTINE
5
6    FOR i ← 8 TO 0 STEP 1
7       OUTPUT(diffCurrencies(i))
8    ENDFOR
```

Rewrite **line 1** and **line 6** from **Figure 13** to make the algorithm work as intended.

[3 marks]

Line 1 _____

_____

Line 6 _____

_____

11. No.1(1.1)

0 1 **Figure 1** shows an algorithm, represented using pseudo-code, which assigns a different value to four variables.

Figure 1

```
country ← 'United States of America'
state ← 'California'
city ← 'San Francisco'
landmark ← 'Alcatraz Island'
```

0 1 . 1 Define the term **algorithm**.

[2 marks]

12. No.2

0 2 **Figure 2** shows an algorithm that uses integer division which has been represented using pseudo-code.

- Line numbers are included but are not part of the algorithm.

Figure 2

```
1   again ← True
2   WHILE again = True
3       a ← USERINPUT
4       IF a > 0 THEN
5           counter ← 0
6           WHILE a > 0
7               a ← a DIV 3
8               counter ← counter + 1
9           ENDWHILE
10      ELSE
11          again ← False
12      ENDIF
13      OUTPUT a
14  ENDWHILE
```

Integer division is the number of times one integer divides into another, with the remainder ignored.

For example:

- `14 DIV 5` evaluates to `2`
- `25 DIV 3` evaluates to `8`

0 2 . 1 Where is iteration **first** used in the algorithm in **Figure 2**?

Shade **one** lozenge.

[1 mark]

A Line number 2 ⬭

B Line number 4 ⬭

C Line number 6 ⬭

D Line number 11 ⬭

0 2 . 2 In the algorithm in **Figure 2**, what will be output when the user input is `10`?

Shade **one** lozenge.

[1 mark]

A 0 ⬭

B 1 ⬭

C 2 ⬭

D 4 ⬭

0 2 . 3 In the algorithm in **Figure 2**, what is the largest possible value of the variable `counter` when the user input is `36`?

Shade **one** lozenge.

[1 mark]

A 0 ⬭

B 2 ⬭

C 4 ⬭

D 5 ⬭

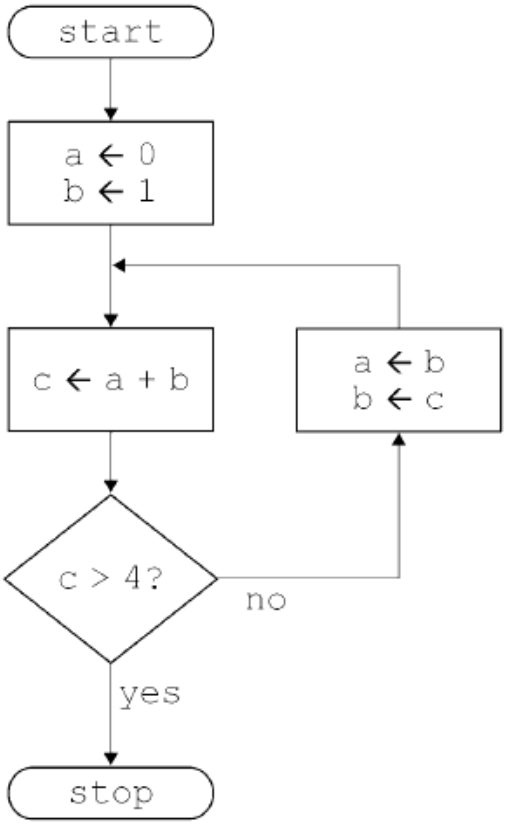
0 3 Explain **one** advantage of the structured approach to programming.

[2 marks]

13. No.5

Figure 4 shows an algorithm presented as a flowchart.

Figure 4



Complete the trace table for the algorithm in **Figure 4**.

You may not need to use all the rows in the table.

[3 marks]

a	b	c

14. No.6

Figure 5 shows an algorithm represented using pseudo-code.

The algorithm is for a simple authentication routine.

The pseudo-code uses a subroutine `getPassword` to check a username:

- If the username exists, the subroutine returns the password stored for that user.
- If the username does not exist, the subroutine returns an empty string.

Parts of the algorithm are missing and have been replaced with the labels L1 to L4.

Figure 5

```
login ← False
REPEAT
    username ← ''
    WHILE username = ''
        OUTPUT 'Enter username: '
        username ← L1
    ENDWHILE
    password ← ''
    WHILE password = ''
        OUTPUT 'Enter password: '
        password ← USERINPUT
    ENDWHILE
    storedPassword ← getPassword( L2 )
    IF storedPassword = L3 THEN
        OUTPUT ' L4 '
    ELSE
        IF password = storedPassword THEN
            login ← True
        ELSE
            OUTPUT 'Try again.'
        ENDIF
    ENDIF
UNTIL login = True
OUTPUT 'You are now logged in.'
```

Figure 6

-1	OUTPUT	0
username	True	SUBROUTINE
1	User not found	''
USERINPUT	password	Wrong password

State the items from **Figure 6** that should be written in place of the labels in the algorithm in **Figure 5**.

You will not need to use all the items in **Figure 6**.

[4 marks]

L1 _____

L2 _____

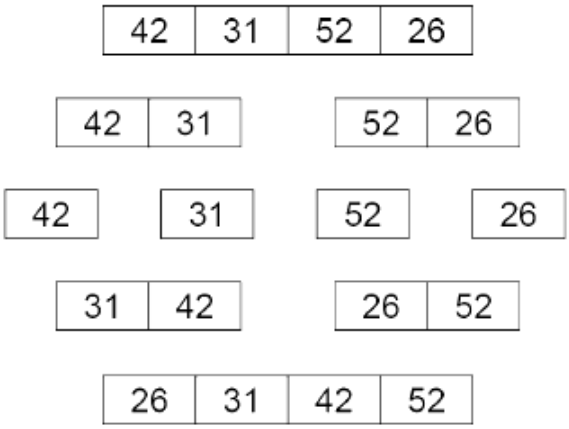
L3 _____

L4 _____

15. No.8

Figure 7 shows a merge sort being carried out on a list.

Figure 7



Explain how the merge sort algorithm works.

[4 marks]

16. No.9

Figure 8 shows an algorithm, written using pseudo-code, that uses a RECORD data structure for storing information about a film.

Each record stores four pieces of information about a film:

- film title
- certificate (eg 12A, PG)
- year the film was made
- if the film is currently being shown at a cinema.

There are records for three films and these films are stored alphabetically in an array called filmCollection.

The pseudo-code outputs the title of the newest of the three films.

- Part of the algorithm has been replaced by the label L1.

Figure 8

```
RECORD Film
   title : String
   certificate : String
   year : Integer
   beingShown : Boolean
ENDRECORD

hulk ← Film('Hulk', '12A', 2005, False)
ironMan ← Film('Iron Man', '12A', 2008, False)
antMan ← Film('Ant-Man', '12A', 2015, False)
filmCollection ← [antMan, hulk, ironMan]
year ← 0
position ← 0

FOR i ← 0 TO L1
   IF filmCollection[i].year > year THEN
      year ← filmCollection[i].year
      position ← i
   ENDIF
ENDFOR

OUTPUT filmCollection[position].title, ' is the
newest film'
```

0 9 . 1 How many different values can the field `beingShown` have?

Shade **one** lozenge.

[1 mark]

A 2 ⬭

B 3 ⬭

C 128 ⬭

D 256 ⬭

0 9 . 2 Which assignment statement changes the year the film *Hulk* was made to 2003?

Shade **one** lozenge.

[1 mark]

A `hulk.year ← 2003` ⬭

B `filmCollection[0].year ← 2003` ⬭

C `Film(year) ← 2003` ⬭

D `hulk(year) ← 2003` ⬭

0 9 . 3 What should the label **L1** in **Figure 8** be replaced by?

Shade **one** lozenge.

[1 mark]

A `3` ⬭

B `LEN(filmCollection)` ⬭

C `LEN(filmCollection) - 1` ⬭

D `Position` ⬭

0 9 . 4 Write a pseudo-code statement that updates the `antMan` record to show that the film is currently being shown at the cinema.

[1 mark]

17. No.10

Figure 9 shows an algorithm, represented in pseudo-code, used to display students' test scores. The algorithm does not work as expected and the teacher wants to find the error.

The algorithm should display three test scores for each student:

- Natalie has results of 78, 81 and 72
- Alex has results of 27, 51 and 54
- Roshana has results of 52, 55 and 59.

- Line numbers are included but are not part of the algorithm.

Figure 9

```
1    names ← ['Natalie', 'Alex', 'Roshana']
2    scores ← [78, 81, 72, 27, 51, 54, 52, 55, 59]
3    count ← 0
4    FOR i ← 0 TO 2
5       person ← names[i]
6       OUTPUT 'Student: ', person
7       FOR j ← 0 TO 1
8          OUTPUT j + 1
9          result ← scores[i * 3 + j]
10         OUTPUT result
11         count ← count + 1
12      ENDFOR
13   ENDFOR
```

10.1 Complete the trace table for the algorithm shown in **Figure 9**.

You may not need to use all the rows in the table.

[5 marks]

count	i	person	j	result

10.2 How could the error in the algorithm in **Figure 9** be corrected?

Shade **one** lozenge.

[1 mark]

A Change line number 3 to: `count ← -1` ⬭

B Change line number 4 to: `FOR i ← 1 TO 4` ⬭

C Change line number 7 to: `FOR j ← 0 TO 2` ⬭

D Change line number 9 to: `result ← scores[j * 3 + i]` ⬭

18. No.11

Figure 10 shows part of an algorithm that has been written in pseudo-code.

There is an error in the algorithm.

The algorithm should:

- get the start year and end year from the user
- check that the start year is before the end year
- check that the start year is before 2000
- calculate the difference between the two years after a valid start year has been entered.

- Line numbers are included but are not part of the algorithm.

Figure 10

```
1    validChoice ← False
2    REPEAT
3       difference ← -1
4       OUTPUT 'Enter a start year '
5       startYear ← USERINPUT
6       OUTPUT 'Enter an end year '
7       endYear ← USERINPUT
8       IF startYear ≥ endYear THEN
9          OUTPUT 'Start year must be before end year'
10      ELSE
11         IF startYear < 2000 THEN
12            OUTPUT 'Start year must be before 2000'
13         ELSE
14            validChoice ← True
15         ENDIF
16      ENDIF
17   UNTIL validChoice = True
18   difference ← endYear - startYear
19   OUTPUT difference
```

1 1 . 1 **Table 1** shows three tests used to check the algorithm in **Figure 10**.

Complete the table to show what the values of the `validChoice` and `difference` variables would be for the given test data.

[4 marks]

Table 1

Test type	**Test data**		`validChoice`	`difference`
Normal	`startYear`	1995		
	`endYear`	2010		
Erroneous	`startYear`	2015		
	`endYear`	2000		
Boundary	`startYear`	2000		
	`endYear`	2023		

1 1 . 2 The algorithm in **Figure 10** contains a logic error on **line 11**.

Describe how the error on **line 11** can be corrected.

[1 mark]

19. No.14

50 students have voted for the music genre they like best.

Figure 16 shows an **incomplete** algorithm, represented using pseudo-code, designed to output the highest or lowest results of the vote.

The programmer has used a two-dimensional array called `results` to store the genre and the number of votes for each genre.

Parts of the algorithm are missing and have been replaced with the labels L1 to L3.

Figure 16

```
SUBROUTINE showResults(method, numberOfGenres)
    results ← [['Pop', 'Post-Punk', 'Techno', 'Metal',
                'Dance'], ['7', '19', '14', '1', '9']]
    pos ← 0
    high ← -1
    IF method = 'HIGHEST' THEN
        FOR i ← 0 TO numberOfGenres - 1
            Votes ← STRING_TO_INT(results[ L1 ][i])
            IF votes > high THEN
                high ← votes
                pos ← L2
            ENDIF
        ENDFOR
    ELSE
        OUTPUT 'not yet working'
    ENDIF
    IF high ≠ -1 THEN
        OUTPUT results[0][pos], ' with ', results[1][pos]
    ENDIF
ENDSUBROUTINE

OUTPUT 'Show the genre with the HIGHEST or LOWEST number
of votes? '
method ← USERINPUT
showResults( L3 , 5)
```

State what should be written in place of the labels L1 to L3 in the algorithm in **Figure 16**.

[3 marks]

L1 _____

L2 _____

L3 _____

20. No.1(1.1)

Figure 1 shows an algorithm, represented using pseudo-code, which assigns a different value to four variables.

Figure 1

```
country ← 'United States of America'
state ← 'California'
city ← 'San Francisco'
landmark ← 'Alcatraz Island'
```

0 1 . 1 Define the term **algorithm**.

[2 marks]

_____

_____

_____

21. No.2(2.1 _ 2.2)

Figure 2 shows an algorithm that uses integer division which has been represented using pseudo-code.

- Line numbers are included but are not part of the algorithm.

Figure 2

```
1   again ← True
2   WHILE again = True
3       a ← USERINPUT
4       IF a > 0 THEN
5           counter ← 0
6           WHILE a > 0
7               a ← a DIV 3
8               counter ← counter + 1
9           ENDWHILE
10      ELSE
11          again ← False
12      ENDIF
13      OUTPUT a
14  ENDWHILE
```

Integer division is the number of times one integer divides into another, with the remainder ignored.

For example:

- `14 DIV 5` evaluates to `2`
- `25 DIV 3` evaluates to `8`

0 2 . 1 Where is iteration **first** used in the algorithm in **Figure 2**?

Shade **one** lozenge.

[1 mark]

A Line number 2 ⬭

B Line number 4 ⬭

C Line number 6 ⬭

D Line number 11 ⬭

0 2 . 2 In the algorithm in **Figure 2**, what will be output when the user input is 10?

Shade **one** lozenge.

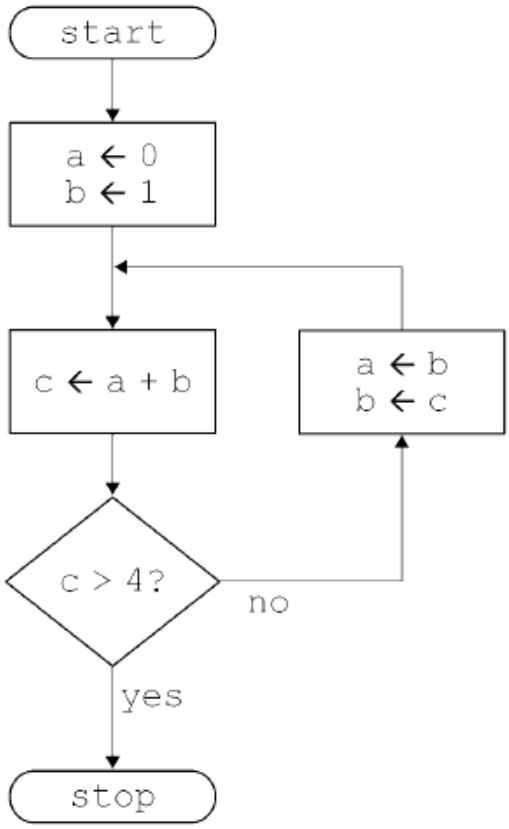
[1 mark]

A	0	⬭
B	1	⬭
C	2	⬭
D	4	⬭

22. No.5

Figure 4 shows an algorithm presented as a flowchart.

Figure 4



Complete the trace table for the algorithm in **Figure 4**.

You may not need to use all the rows in the table.

[3 marks]

a	b	c

23. No.6

Figure 5 shows an algorithm represented using pseudo-code.

The algorithm is for a simple authentication routine.

The pseudo-code uses a subroutine `getPassword` to check a username:

- If the username exists, the subroutine returns the password stored for that user.
- If the username does not exist, the subroutine returns an empty string.

Parts of the algorithm are missing and have been replaced with the labels L1 to L4.

Figure 5

```
login ← False
REPEAT
    username ← ''
    WHILE username = ''
        OUTPUT 'Enter username: '
        username ← L1
    ENDWHILE
    password ← ''
    WHILE password = ''
        OUTPUT 'Enter password: '
        password ← USERINPUT
    ENDWHILE
    storedPassword ← getPassword( L2 )
    IF storedPassword = L3 THEN
        OUTPUT ' L4 '
    ELSE
        IF password = storedPassword THEN
            login ← True
        ELSE
            OUTPUT 'Try again.'
        ENDIF
    ENDIF
UNTIL login = True
OUTPUT 'You are now logged in.'
```

Figure 6

-1	OUTPUT	0
username	True	SUBROUTINE
1	User not found	''
USERINPUT	password	Wrong password

State the items from **Figure 6** that should be written in place of the labels in the algorithm in **Figure 5**.

You will not need to use all the items in **Figure 6**.

[4 marks]

L1 _____

L2 _____

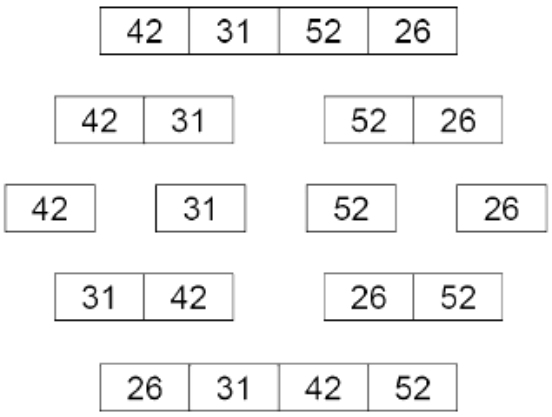
L3 _____

L4 _____

24. No.8

Figure 7 shows a merge sort being carried out on a list.

Figure 7



Explain how the merge sort algorithm works.

[4 marks]

25. No.9

Figure 8 shows an algorithm, written using pseudo-code, that uses a RECORD data structure for storing information about a film.

Each record stores four pieces of information about a film:

- film title
- certificate (eg 12A, PG)
- year the film was made
- if the film is currently being shown at a cinema.

There are records for three films and these films are stored alphabetically in an array called filmCollection.

The pseudo-code outputs the title of the newest of the three films.

- Part of the algorithm has been replaced by the label L1.

Figure 8

```
RECORD Film
   title : String
   certificate : String
   year : Integer
   beingShown : Boolean
ENDRECORD

hulk ← Film('Hulk', '12A', 2005, False)
ironMan ← Film('Iron Man', '12A', 2008, False)
antMan ← Film('Ant-Man', '12A', 2015, False)
filmCollection ← [antMan, hulk, ironMan]
year ← 0
position ← 0

FOR i ← 0 TO L1
   IF filmCollection[i].year > year THEN
      year ← filmCollection[i].year
      position ← i
   ENDIF
ENDFOR

OUTPUT filmCollection[position].title, ' is the
newest film'
```

0 9 . 1 How many different values can the field `beingShown` have?

Shade **one** lozenge.

[1 mark]

A 2 ⬭

B 3 ⬭

C 128 ⬭

D 256 ⬭

0 9 . 2 Which assignment statement changes the year the film *Hulk* was made to 2003?

Shade **one** lozenge.

[1 mark]

A `hulk.year ← 2003` ⬭

B `filmCollection[0].year ← 2003` ⬭

C `Film(year) ← 2003` ⬭

D `hulk(year) ← 2003` ⬭

0 9 . 3 What should the label L1 in **Figure 8** be replaced by?

Shade **one** lozenge.

[1 mark]

A `3` ⬭

B `LEN(filmCollection)` ⬭

C `LEN(filmCollection) - 1` ⬭

D `Position` ⬭

0 9 . 4 Write a pseudo-code statement that updates the `antMan` record to show that the film is currently being shown at the cinema.

[1 mark]

26. No.10

Figure 9 shows an algorithm, represented in pseudo-code, used to display students' test scores. The algorithm does not work as expected and the teacher wants to find the error.

The algorithm should display three test scores for each student:

- Natalie has results of 78, 81 and 72
- Alex has results of 27, 51 and 54
- Roshana has results of 52, 55 and 59.

- Line numbers are included but are not part of the algorithm.

Figure 9

```
1    names ← ['Natalie', 'Alex', 'Roshana']
2    scores ← [78, 81, 72, 27, 51, 54, 52, 55, 59]
3    count ← 0
4    FOR i ← 0 TO 2
5       person ← names[i]
6       OUTPUT 'Student: ', person
7       FOR j ← 0 TO 1
8          OUTPUT j + 1
9          result ← scores[i * 3 + j]
10         OUTPUT result
11         count ← count + 1
12      ENDFOR
13   ENDFOR
```

10.1 Complete the trace table for the algorithm shown in **Figure 9**.

You may not need to use all the rows in the table.

[5 marks]

count	i	person	j	result

10.2 How could the error in the algorithm in **Figure 9** be corrected?

Shade **one** lozenge.

[1 mark]

A Change line number 3 to: `count ← -1` ⬭

B Change line number 4 to: `FOR i ← 1 TO 4` ⬭

C Change line number 7 to: `FOR j ← 0 TO 2` ⬭

D Change line number 9 to: `result ← scores[j * 3 + i]` ⬭

27. No.11

Figure 10 shows part of an algorithm that has been written in pseudo-code.

There is an error in the algorithm.

The algorithm should:

- get the start year and end year from the user
- check that the start year is before the end year
- check that the start year is before 2000
- calculate the difference between the two years after a valid start year has been entered.

- Line numbers are included but are not part of the algorithm.

Figure 10

```
1    validChoice ← False
2    REPEAT
3       difference ← -1
4       OUTPUT 'Enter a start year '
5       startYear ← USERINPUT
6       OUTPUT 'Enter an end year '
7       endYear ← USERINPUT
8       IF startYear ≥ endYear THEN
9          OUTPUT 'Start year must be before end year'
10      ELSE
11         IF startYear < 2000 THEN
12            OUTPUT 'Start year must be before 2000'
13         ELSE
14            validChoice ← True
15         ENDIF
16      ENDIF
17   UNTIL validChoice = True
18   difference ← endYear - startYear
19   OUTPUT difference
```

1 1 . 1 **Table 1** shows three tests used to check the algorithm in **Figure 10**.

Complete the table to show what the values of the `validChoice` and `difference` variables would be for the given test data.

[4 marks]

Table 1

Test type	**Test data**		`validChoice`	`difference`
Normal	`startYear`	`1995`		
	`endYear`	`2010`		
Erroneous	`startYear`	`2015`		
	`endYear`	`2000`		
Boundary	`startYear`	`2000`		
	`endYear`	`2023`		

1 1 . 2 The algorithm in **Figure 10** contains a logic error on **line 11**.

Describe how the error on **line 11** can be corrected.

[1 mark]