Answer **all** questions.

0 1 . 1 Define the term algorithm.

[2 marks]

0 1 . 2 The following are computer science terms (labelled **A** – **E**).

A abstraction
B data type
C decomposition
D efficiency
E input

For each of the definitions in the table, write the label of the most suitable computer science term. Use a label only once.

[3 marks]

	Label
Breaking a problem down into a number of sub-problems.	
The process of removing unnecessary detail from a problem.	
Defines the range of values a variable may take.	

Turn over for the next question

0 2 The subroutine `CHAR_TO_CODE(character)` returns the integer ASCII value of a character. For example,

`CHAR_TO_CODE('a')` returns the value 97
`CHAR_TO_CODE('z')` returns the value 122
`CHAR_TO_CODE('`')` returns the value 96
`CHAR_TO_CODE('{')` returns the value 123

Develop an algorithm, using either pseudo-code or a flowchart, that:

- asks the user to enter a character
- outputs '`LOWER`' if the user has entered a lowercase character
- outputs '`NOT LOWER`' if the user has entered any other character.

You **must** use the built-in `CHAR_TO_CODE` subroutine in your answer.

[7 marks]

Turn over for the next question

0 3 Develop an algorithm using either pseudo-code or a flowchart that allows a taxi company to calculate how much a taxi fare should be.

The algorithm should:

- prompt the user to enter the journey distance in kilometres
  - the distance entered must be greater than zero
  - the user should be made to re-enter the distance until the distance entered is valid
- prompt the user to enter the number of passengers (no validation is required)
- calculate the taxi fare by
  - charging £2 for every passenger regardless of the distance
  - charging a further £1.50 for every kilometre regardless of how many passengers there are
- output the final taxi fare.

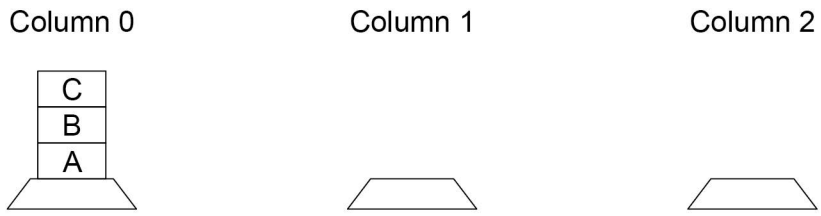
[8 marks]

Turn over for the next question

0 4 The following subroutines control the way that labelled blocks are placed in different columns.

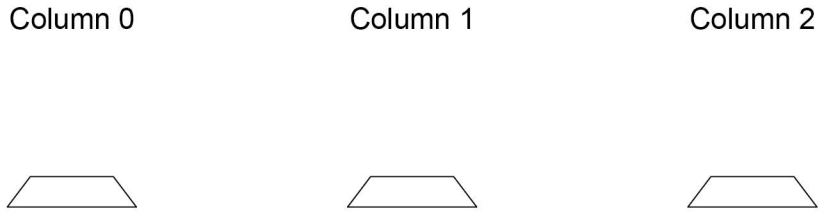
`BLOCK_ON_TOP(column)`	returns the label of the block on top of the column given as a parameter.
`MOVE(source, destination)`	moves the block on top of the `source` column to the top of the `destination` column.
`HEIGHT(column)`	returns the number of blocks in the specified column.

0 4 . 1 This is how the blocks A, B and C are arranged at the start.



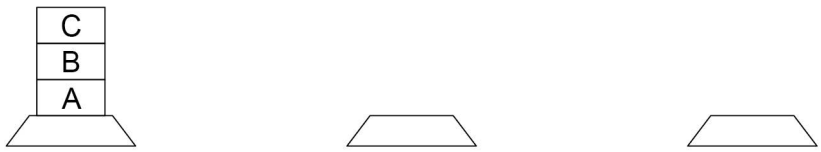
Draw the final arrangement of the blocks after the following algorithm has run.

```
MOVE(0, 1)
MOVE(0, 2)
MOVE(0, 2)
```



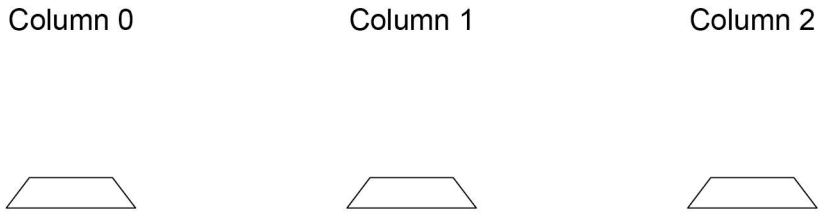
[3 marks]

0 4 . 2 This is how the blocks A, B and C are arranged at the start.



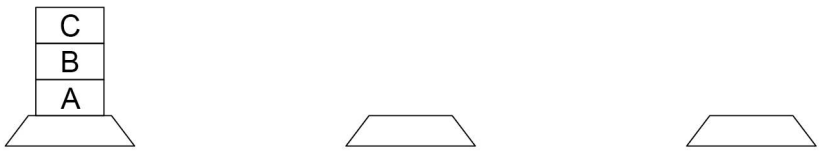
Draw the final arrangement of the blocks after the following algorithm has run.

```
WHILE HEIGHT(0) > 1
   MOVE(0, 1)
ENDWHILE
MOVE(1, 2)
```



[3 marks]

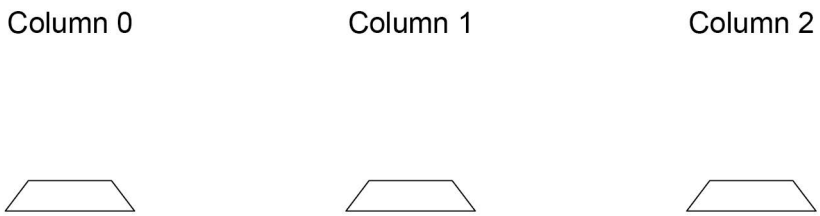
0 4 . 3 This is how the blocks A, B and C are arranged at the start.



Draw the final arrangement of the blocks after the following algorithm has run.

```
FOR c ← 0 TO 2
   IF BLOCK_ON_TOP(0) = 'B' THEN
      MOVE(0, (c+1) MOD 3)
   ELSE
      MOVE(0, (c+2) MOD 3)
   ENDIF
ENDFOR
```

This algorithm uses the MOD operator which calculates the remainder resulting from integer division. For example, `13 MOD 5 = 3`.



[3 marks]

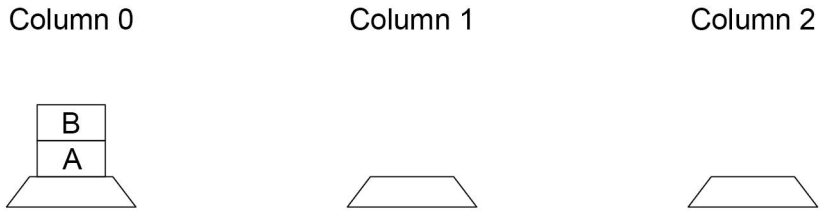
0 4 . 4 Develop an algorithm using either pseudo-code or a flowchart that will move every block from column 0 to column 1.

Your algorithm should work however many blocks start in column 0. You may assume there will always be at least one block in column 0 at the start and that the other columns are empty.

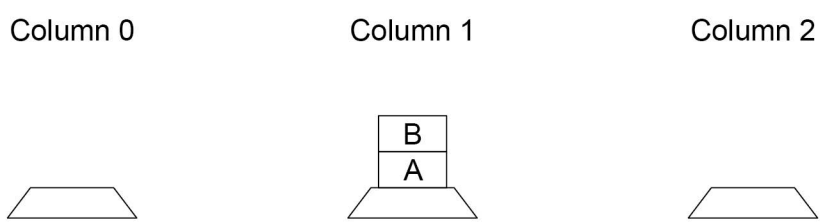
The order of the blocks must be preserved.

The `MOVE` subroutine must be used to move a block from one column to another. You should also use the `HEIGHT` subroutine in your answer.

For example, if the starting arrangement of the blocks is:



Then the final arrangement should have block B above block A:



[5 marks]

Turn over for the next question

0 5 . 1 Complete the trace table for the algorithm shown in **Figure 4** for when the user enters the value 750 when prompted.

[4 marks]

Figure 4

```
constant PAYLOAD_SIZE ← 250
constant HEADER_SIZE ← 50
OUTPUT 'Enter the number of bits of data to be sent'
dataToBeSent ← USERINPUT
totalSize ← PAYLOAD_SIZE + HEADER_SIZE
numberOfPackets ← 0
REPEAT
   dataToBeSent ← dataToBeSent - totalSize
   numberOfPackets ← numberOfPackets + 1
UNTIL dataToBeSent ≤ 0
```

totalSize	dataToBeSent	numberOfPackets
	750	

0 5 . 2 State why both PAYLOAD_SIZE and HEADER_SIZE from the algorithm in **Figure 4** did not need to be included in the trace table.

[1 mark]

0 5 . 3 Shade **one** lozenge to show which of the following best represents the input and output to/from the algorithm in **Figure 4**.

[1 mark]

A Input: dataToBeSent, output: numberOfPackets ⬭

B Input: numberOfPackets, output: totalSize ⬭

C Input: totalSize, output: dataToBeSent ⬭

0 5 . 4 A developer looks at the algorithm in **Figure 4** and realises that the use of iteration is unnecessary if they use a combination of the `DIV` and `MOD` operators.

- `DIV` calculates integer division, eg `11 DIV 4 = 2`
- `MOD` calculates the remainder after integer division, eg `11 MOD 4 = 3`

The programmer realises that she can rewrite the algorithm by replacing the `REPEAT-UNTIL` structure with code that uses selection, `MOD` and `DIV` instead.

Complete this new algorithm by stating the code that should be written in the boxes labelled **A**, **B** and **C**. This new algorithm should calculate the same final result for the variable `numberOfPackets` as the original algorithm in **Figure 4**.

[3 marks]

```
constant PAYLOAD_SIZE ← 250
constant HEADER_SIZE ← 50
OUTPUT 'Enter the number of bits of data to be sent'
dataToBeSent ← USERINPUT
totalSize ← PAYLOAD_SIZE + HEADER_SIZE
numberOfPackets ← dataToBeSent DIV totalSize
IF [ A ] MOD [ B ] > 0 THEN
  numberOfPackets ← [ C ]
ENDIF
```

A _____

B _____

C _____

Turn over for the next question

0 6 Run length encoding (RLE) is a form of compression that creates frequency/data pairs to describe the original data.

For example, an RLE of the bit pattern `00000011101111` could be `6 0 3 1 1 0 4 1` because there are six `0`s followed by three `1`s followed by one `0` and finally four `1`s.

The algorithm in **Figure 7** is designed to output an RLE for a bit pattern that has been entered by the user.

Five parts of the code labelled **L1**, **L2**, **L3**, **L4** and **L5** are missing.

- Note that indexing starts at zero.

Figure 7

```
pattern ← L1
i ← L2
count ← 1
WHILE i < LEN(pattern)-1
   IF pattern[i] L3 pattern[i+1] THEN
      count ← count + 1
   ELSE
      L4
      OUTPUT pattern[i]
      count ← 1
   ENDIF
   L5
ENDWHILE
OUTPUT count
OUTPUT pattern[i]
```

0 6 . 1 Shade **one** lozenge to show what code should be written at point **L1** of the algorithm.

[1 mark]

A `OUTPUT` ⬭

B `'RLE'` ⬭

C `True` ⬭

D `USERINPUT` ⬭

0 6 . 2 Shade **one** lozenge to show what value should be written at point **L2** of the algorithm.

[1 mark]

A -1 ⬭

B 0 ⬭

C 1 ⬭

D 2 ⬭

0 6 . 3 Shade **one** lozenge to show what operator should be written at point **L3** of the algorithm.

[1 mark]

A = ⬭

B ≤ ⬭

C < ⬭

D ≠ ⬭

0 6 . 4 Shade **one** lozenge to show what code should be written at point **L4** of the algorithm.

[1 mark]

A `count` ⬭

B `count ← count - 1` ⬭

C `count ← USERINPUT` ⬭

D `OUTPUT count` ⬭

0 6 . 5 Shade **one** lozenge to show what code should be written at point **L5** of the algorithm.

[1 mark]

A `i ← i * 2` ⬭

B `i ← i + 1` ⬭

C `i ← i + 2` ⬭

D `i ← i DIV 2` ⬭

0 6 . 6 State a run length encoding of the series of characters `ttjjeeess`

[2 marks]

0 6 . 7 A developer implements the algorithm shown in **Figure 7** and tests their code to check that it is working correctly. The developer tests it only with the input bit pattern that consists of six zeros and it correctly outputs `6 0`.

Using example test data, state **three** further tests that the developer could use to improve the testing of their code.

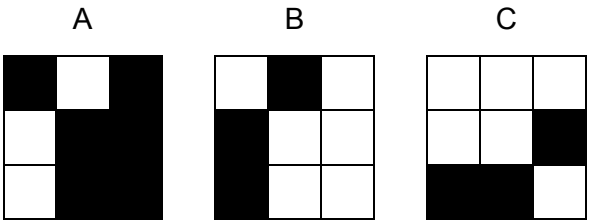
[3 marks]

0 7

A black and white image can be represented as a two-dimensional array where:
- 0 represents a white pixel
- 1 represents a black pixel.

Two images are exact inverses of each other if:
- every white pixel in the first image is black in the second image
- every black pixel in the first image is white in the second image.

For example, B is the inverse of A but C is not the inverse of A:



A developer has started to create an algorithm that compares two 3x3 black and white images, `image1` and `image2`, to see if they are exact inverses of each other.

Complete the algorithm in pseudo-code, ensuring that, when the algorithm ends, the value of the variable `inverse` is `true` if the two images are inverses of each other or `false` if they are not inverses of each other.

The algorithm should work for any 3x3 black and white images stored in `image1` and `image2`.

- Note that indexing starts at zero.

```
image1 ← [ [0, 0, 0], [0, 1, 1], [1, 1, 0] ]
image2 ← [ [1, 1, 1], [1, 1, 0], [0, 0, 1] ]
inverse ← true
i ← 0
WHILE i ≤ 2
   j ← 0
   WHILE j ≤ 2
```

[6 marks]

0 8 The algorithms shown in **Figure 4** and **Figure 5** both have the same purpose.

The operator `LEFTSHIFT` performs a binary shift to the left by the number indicated.

For example, `6 LEFTSHIFT 1` will left shift the number `6` by one place, which has the effect of multiplying the number `6` by two giving a result of 12

Figure 4

```
result ← number LEFTSHIFT 2
result ← result - number
```

Figure 5

```
result ← 0
FOR x ← 1 TO 3
   result ← result + number
ENDFOR
```

0 8 . 1 Complete the trace table for the algorithm shown in **Figure 4** when the initial value of `number` is 4

You may not need to use all rows of the trace table.

[2 marks]

result

0 8 . 2 Complete the trace table for the algorithm shown in **Figure 5** when the initial value of `number` is 4

You may not need to use all rows of the trace table.

[2 marks]

x	result

0 8 . 3 The algorithms in **Figure 4** and **Figure 5** have the same purpose.

State this purpose.

[1 mark]

0 8 . 4 Explain why the algorithm shown in **Figure 4** can be considered to be a more efficient algorithm than the algorithm shown in **Figure 5**.

[1 mark]

Turn over for the next question

0 9 . 1 Four subroutines are shown in **Figure 7**.

Figure 7

```
SUBROUTINE main(k)
   OUTPUT k
   WHILE k > 1
      IF isEven(k) = True THEN
         k ← decrease(k)
      ELSE
         k ← increase(k)
      ENDIF
      OUTPUT k
   ENDWHILE
ENDSUBROUTINE

SUBROUTINE decrease(n)
   result ← n DIV 2
   RETURN result
ENDSUBROUTINE

SUBROUTINE increase(n)
   result ← (3 * n) + 1
   RETURN result
ENDSUBROUTINE

SUBROUTINE isEven(n)
   IF (n MOD 2) = 0 THEN
      RETURN True
   ELSE
      RETURN False
   ENDIF
ENDSUBROUTINE
```

Complete the table showing **all** of the outputs from the subroutine call `main(3)`

The first output has already been written in the trace table. You may not need to use all rows of the table.

[4 marks]

Output
3

0 9 . 2 Describe how the developer has used the structured approach to programming in **Figure 7**.

[2 marks]

1 0 An application allows only two users to log in. Their usernames are stated in **Table 1** along with their passwords.

Table 1

username	password
gower	9Fdg3
tuff	888rG

Develop an algorithm, using either pseudo-code **or** a flowchart, that authenticates the user. The algorithm should:

- get the user to enter their username and password
- check that the combination of username and password is correct and, if so, output the string `'access granted'`
- get the user to keep re-entering their username and password until the combination is correct.

[6 marks]

1 1 Develop an algorithm, using either pseudo-code **or** a flowchart, that helps an ice cream seller in a hot country calculate how many ice creams they are likely to sell on a particular day. Your algorithm should:

- get the user to enter whether it is the weekend or a weekday
- get the user to enter the temperature forecast in degrees Celsius (they should enter a number between 20 and 45 inclusive; if the number falls outside of this range then they should be made to re-enter another number until they enter a valid temperature)
- calculate the number of ice creams that are likely to be sold using the following information:
  - 100 ice creams are likely to be sold if the temperature is between 20 and 30 degrees inclusive,
  - 150 ice creams are likely to be sold if the temperature is between 31 and 38 degrees inclusive,
  - and 120 ice creams are likely to be sold if the temperature is higher than 38 degrees
- double the estimate if it is a weekend
- output the estimated number of ice creams that are likely to be sold.

[9 marks]

1 2 A program is being developed that allows users to rate and review movies. A user will enter their rating (out of 10) and a written review for each movie they have watched.

Computational thinking skills are used during the development of the program.

1 2 . 1 Define the term **abstraction**.

[1 mark]

1 2 . 2 A user will be able to register, log in and log out of the program. When registering, a new user will enter their details, before confirming their email address.

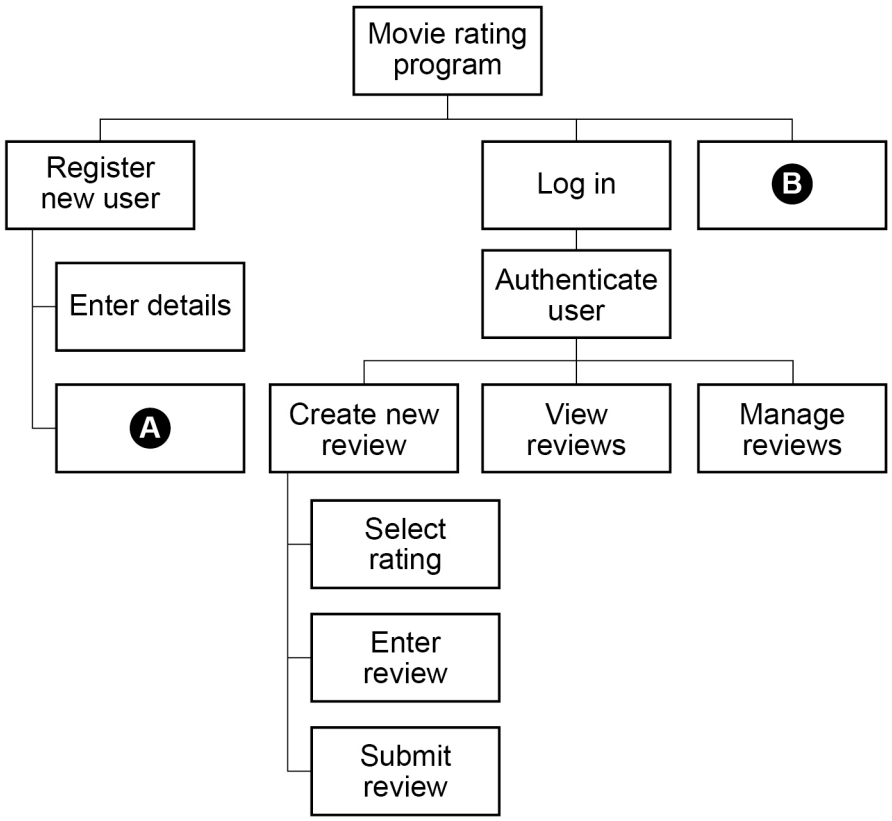
Decomposition has been used to break the problem down into smaller sub-problems.

Figure 7 represents the design of the program.

Complete the decomposition of this program by stating what should be written in boxes **A** and **B**.

[2 marks]

Figure 7



A

B

Turn over for the next question

1 3 Write a C# program to check if an email address has been entered correctly by a user.

Your program must:

- get the user to input an email address
- get the user to input the email address a second time
- output the message `Match` **and** output the email address if the email addresses entered are the same
- output the message `Do not match` if the email addresses entered are not the same.

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.

[5 marks]

1 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}");
   }
}
```

1 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	

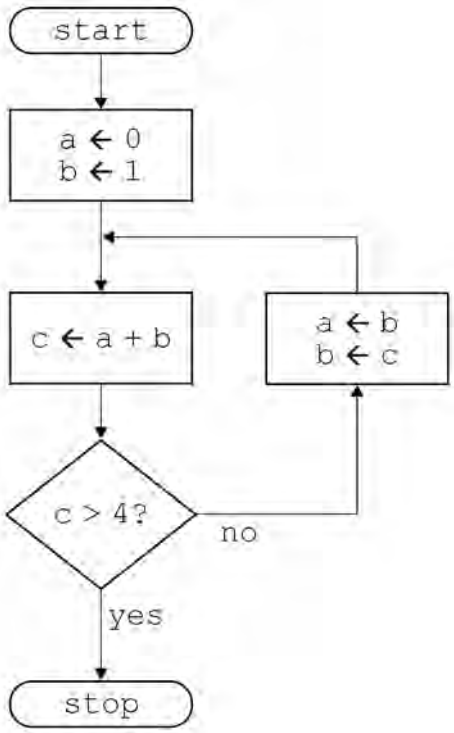
1 4 . 2 Describe **one** way that the program in **Figure 3** could be made more robust.

[1 mark]

Turn over for the next question

1 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

1 6 **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 6 . 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 6 . 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]` ⬭

Turn over for the next question

1 7 . 1 Define the term algorithm.

[2 marks]

1 7 . 2 The following are computer science terms (labelled **A** – **E**).

A assignment
B data type
C decomposition
D efficiency
E input

For each of the definitions in the table, write the label of the most suitable computer science term. Use a label only once.

[3 marks]

	Label
Breaking a problem down into a number of sub-problems	
The process of setting the value stored in a variable	
Defines the range of values a variable may take	

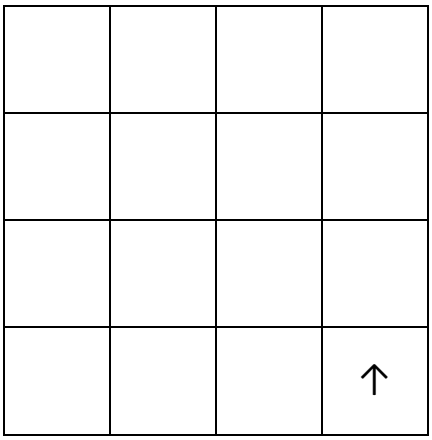
1 8 Four separate subroutines have been written to control a robot.

- `Forward(n)` moves the robot `n` squares forward.
- `TurnLeft()` turns the robot 90 degrees left.
- `TurnRight()` turns the robot 90 degrees right.
- `ObjectAhead()` returns `true` if the robot is facing an object in the next square or returns `false` if this square is empty.

1 8 . 1 Draw the path of the robot through the grid below if the following program is executed (the robot starts in the square marked by the ↑ facing in the direction of the arrow).

```
Forward(2)
TurnLeft()
Forward(1)
TurnRight()
Forward(1)
```

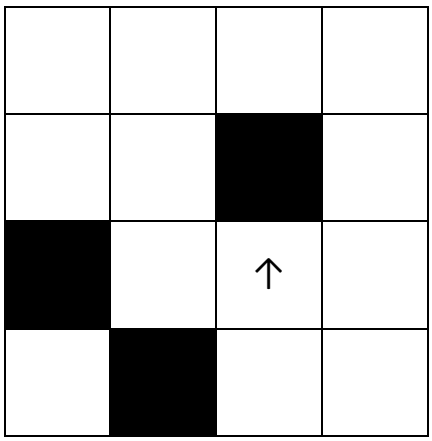
[3 marks]



1 8 . 2 Draw the path of the robot through the grid below if the following program is executed (the robot starts in the square marked by the ↑ facing in the direction of the arrow). If a square is black then it contains an object.

```
WHILE ObjectAhead() = true
  TurnLeft()
    IF ObjectAhead() = true THEN
      TurnRight()
      TurnRight()
    ENDIF
  Forward(1)
ENDWHILE
Forward(1)
```

[3 marks]



Turn over for the next question

1 9 A developer is developing a program for a client. The developer is given the following instructions.

"Many of my friends ask me to walk their dogs for them. All of these friends pay me to do this and the amount I get paid depends on how long I walk their dogs for. If they have more than one dog then I don't charge the owner any extra. I like to walk the dogs in the afternoon when the weather is normally best because I often get colds. I need you to help me keep track of how much I'm owed – fortunately for me all of my friends have different first names so it is really easy to tell them apart. I charge £10 for every 30 minutes of the walk (and I always round this up so 47 minutes would be two half-hour charges or £20).

1 9 . 1 The developer needs to remove all of the unnecessary detail from the client's request. Shade the lozenge next to the name for this process.

[1 mark]

A Abstraction ⬭

B Conversion ⬭

C Decomposition ⬭

D Validation ⬭

1 9 . 2 The developer has decided that the following two points are the only important details from the client's request.

- The charge is based on time and not how many dogs are walked.
- The charge is £10 every 30 minutes.

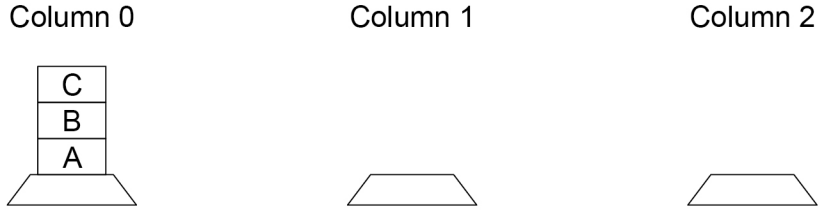
State **two** other relevant details that the developer has missed.

[2 marks]

2 0 The following subroutines control the way that labelled blocks are placed in different columns.

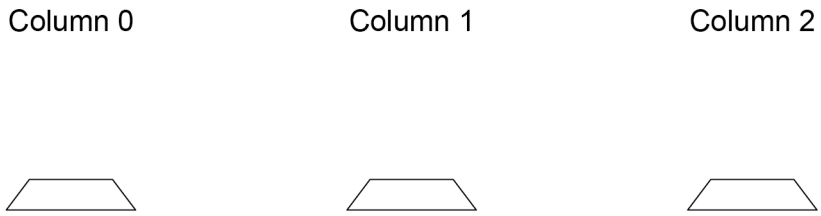
`BLOCK_ON_TOP(column)`	returns the label of the block on top of the column given as a parameter.
`MOVE(source, destination)`	moves the block on top of the `source` column to the top of the `destination` column.
`HEIGHT(column)`	returns the number of blocks in the specified column.

2 0 . 1 This is how the blocks A, B and C are arranged at the start.



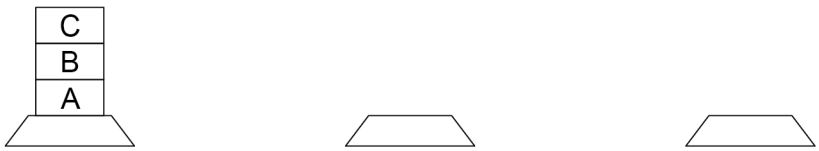
Draw the final arrangement of the blocks after the following algorithm has run.

```
MOVE(0, 1)
MOVE(0, 2)
MOVE(0, 2)
```



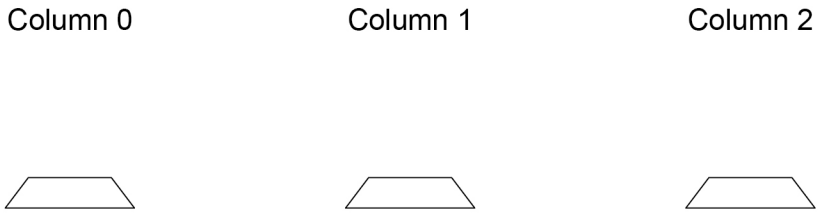
[3 marks]

2 0 . 2 This is how the blocks A, B and C are arranged at the start.



Draw the final arrangement of the blocks after the following algorithm has run.

```
WHILE HEIGHT(0) > 1
   MOVE(0, 1)
ENDWHILE
MOVE(1, 2)
```



[3 marks]

Turn over for the next question

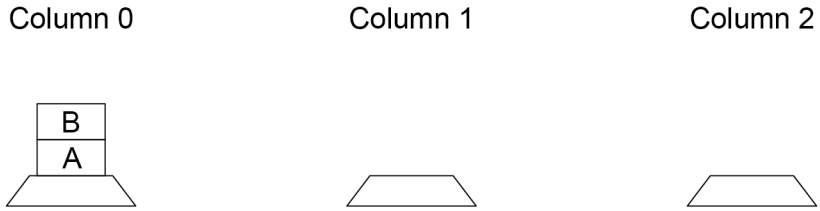
2 0 . 3 Develop an algorithm using either pseudo-code or a flowchart that will move every block from column 0 to column 1.

Your algorithm should work however many blocks start in column 0. You may assume there will always be at least one block in column 0 at the start and that the other columns are empty.

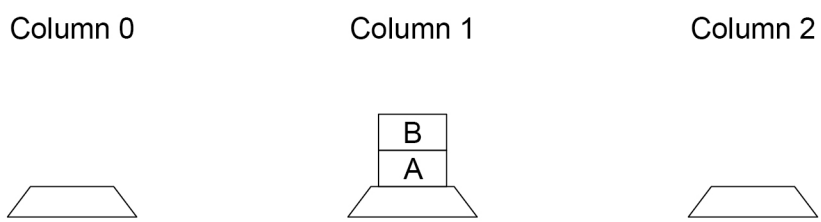
The order of the blocks must be preserved.

The `MOVE` subroutine must be used to move a block from one column to another. You should also use the `HEIGHT` subroutine in your answer.

For example, if the starting arrangement of the blocks is:



Then the final arrangement should have block B above block A:



[4 marks]

2 1 . 1 Define the term **abstraction**.

[1 mark]

2 1 . 2 State the name for the process of breaking a problem down into sub-problems.

[1 mark]

2 2 **Figure 4** shows an algorithm, represented using pseudo-code.

The algorithm calculates the total cost of hiring a hotel for a wedding.

Figure 4

```
numberOfGuests ← USERINPUT
numberOfRooms ← USERINPUT
charge ← 25
IF numberOfGuests > 50 THEN
   totalCost ← numberOfGuests * 2
ELSE
   IF numberOfGuests ≥ 25 THEN
      totalCost ← numberOfGuests * 4
   ELSE
      totalCost ← numberOfGuests * 5
   ENDIF
ENDIF
totalCost ← totalCost + (numberOfRooms * 100)
IF totalCost < 1400 THEN
   totalCost ← totalCost + charge
ENDIF
OUTPUT totalCost
```

Complete the table below using the algorithm in **Figure 4**.

[3 marks]

Input value for numberOfGuests	**Input value for** numberOfRooms	**Output**
50	30	
20	10	
500	5	

2 3 **Figure 6** shows an algorithm, represented using pseudo-code.

Figure 6

```
days ← [10, 15, 4]
sales ← [20, 33, 12]
weeks ← [0, 0, 0]
FOR i ← 0 TO 2
   daysTotal ← days[i] + sales[i]
   weeks[i] ← daysTotal DIV 7
ENDFOR
weeksTotal ← weeks[0] + weeks[1] + weeks[2]
OUTPUT weeksTotal
```

The `DIV` operator is used for integer division.

Complete the trace table for the algorithm in **Figure 6**.

Part of the table has already been filled in.

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

[6 marks]

i	daysTotal	weeks			weeksTotal
		[0]	[1]	[2]	
		0	0	0	