Please write clearly in block capitals.

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

Surname _____

Forename(s) _____

Candidate signature _____

GCSE
COMPUTER SCIENCE

Paper 1 - Computational thinking and programming skills

Specimen Assessment MaterialsTime allowed: 2 hours

Materials
- There are no additional materials required for this paper.
- You must **not** use a calculator.

Instructions
- Use black ink or black ball-point pen. Use pencil only for drawing.
- Answer **all** questions.
- You must answer the questions in the spaces provided.
- Do all rough work in this book.
- Cross through any work you do not want to be marked.
- Questions that require a coded solution must be answered in C#

Information
- The total number of marks available for this paper is 90.

Advice

For the multiple-choice questions, completely fill in the lozenge alongside the appropriate answer.

CORRECT METHOD  WRONG METHODS 

If you want to change your answer you must cross out your original answer as shown. 

If you wish to return to an answer previously crossed out, ring the answer you now wish to select as shown. 

Do not write outside the box

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 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	

0 2 The pseudo-code in **Figure 1** assigns two string values to two variables.

Figure 1

```
title ← 'computer science'
level ← 'gcse'
```

0 2 . 1 Shade **one** lozenge that shows the length of the contents of the variable `level` in **Figure 1**.

[1 mark]

A 1 ⬭

B 2 ⬭

C 3 ⬭

D 4 ⬭

0 2 . 2 Shade **one** lozenge that shows the result of concatenating the variable `title` with the variable `level` in **Figure 1**.

[1 mark]

A `'computer science gcse'` ⬭

B `'Computer Science GCSE'` ⬭

C `'computersciencegcse'` ⬭

D `'computer sciencegcse'` ⬭

Turn over for the next question

Do not write outside the box

0 3 The algorithm in **Figure 2** has been developed to automate the quantity of dog biscuits to put in a dog bowl at certain times of the day.

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

Figure 2

```
1     time ← USERINPUT
2     IF time = 'breakfast' THEN
3        q ← 1
4     ELSE IF time = 'lunch' THEN
5        q ← 4
6     ELSE IF time = 'dinner' THEN
7        q ← 2
8     ELSE
9        OUTPUT 'time not recognised'
10    ENDIF
11    FOR n ← 1 TO q
12       IF n < 3 THEN
13          DISPENSE_BISCUIT('chewies')
14       ELSE
15          DISPENSE_BISCUIT('crunchy')
16       ENDIF
17    ENDFOR
```

0 3 . 1 Shade **one** lozenge which shows the line number where selection is **first** used in the algorithm shown in **Figure 2**.

[1 mark]

A Line number 2 ⬭

B Line number 4 ⬭

C Line number 9 ⬭

D Line number 12 ⬭

0 3 . 2 Shade **one** lozenge which shows the line number where iteration is **first** used in the algorithm shown in **Figure 2**.

[1 mark]

A Line number 1 ⬭

B Line number 8 ⬭

C Line number 11 ⬭

D Line number 13 ⬭

0 3 . 3 Shade **one** lozenge which shows how many times the subroutine `DISPENSE_BISCUIT` would be called if the user input is `'breakfast'` in **Figure 2**.

[1 mark]

A 1 subroutine call ⬭

B 2 subroutine calls ⬭

C 3 subroutine calls ⬭

D 4 subroutine calls ⬭

0 3 . 4 Shade **one** lozenge which shows the data type of the variable `time` in the algorithm shown in **Figure 2**.

[1 mark]

A Date/Time ⬭

B String ⬭

C Integer ⬭

D Real ⬭

0 3 . 5 State how many times the subroutine `DISPENSE_BISCUIT` will be called with the parameter `'chewies'` if the user input is `'lunch'` in **Figure 2.**

[1 mark]

Turn over for the next question

0 4 A programmer has written a C# program that asks the user to input two integers and then output which of the two integers is the largest.
Complete the program by filling in the gaps using the information in **Figure 3**.
Each item in **Figure 3** should only be used once.

[5 marks]

Figure 3

Console.Write	num1	num2	output
else	<	>	else if
string	double	int	

```
int num1;

__________ num2;

Console.WriteLine("Enter a number: ");

num1 = int.Parse(Console.ReadLine());

Console.WriteLine("Enter another number: ");

num2 = int.Parse(Console.ReadLine());

if (num1 > num2)

{

    Console.WriteLine("    _________ is bigger.");

}

else

if (num1 _____________ num2)

{

    Console.WriteLine("    _________ is bigger.");

}

_____________________

{

    Console.WriteLine("The numbers are equal.");

}
```

0 5 Write a C# program that allows a taxi company to calculate how much a taxi fare should be.

The program should:

- allow the user to enter the journey distance in kilometres (no validation is required)
- allow 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.

You **should** use meaningful variable name(s), correct syntax and indentation in your answer.

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

[7 marks]

0 6 Write a C# program that inputs a password and checks if it is correct.

Your program should work as follows:

- input a password and store it in a suitable variable
- if the password entered is equal to `secret` display the message `Welcome`
- if the password entered is not equal to `secret` display the message `Not welcome`.

You **should** use meaningful variable name(s), correct syntax and indentation in your answer.

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

[5 marks]

0 7 The algorithm in **Figure 4** is a sorting algorithm.

- Array indexing starts at 0.
- Line numbers are included but are not part of the algorithm.

Figure 4

```
1    arr ← [4, 1, 6]
2    swapsMade ← false
3    WHILE swapsMade = false
4       swapsMade ← true
5       i ← 0
6       WHILE i < 2
7          IF arr[i+1] < arr[i] THEN
8             t ← arr[i]
9             arr[i] ← arr[i+1]
10            arr[i+1] ← t
11            swapsMade ← false
12         ENDIF
13         i ← i + 1
14      ENDWHILE
15   ENDWHILE
```

0 7 . 1 State the data type of the variable `swapsMade` in the algorithm shown in **Figure 4**.

[1 mark]

0 7 . 2 The identifier `swapsMade` is used in the algorithm shown in **Figure 4**.

Explain why this is a better choice than using the identifier `s`.

[2 marks]

0 7 . 3 Shade **one** lozenge to show which of the following contains the **false** statement about the algorithm in **Figure 4**.

[1 mark]

A The algorithm uses a named constant. ⬭

B The algorithm uses indefinite iteration. ⬭

C The algorithm uses nested iteration. ⬭

0 7 . 4 Complete the trace table for the algorithm shown in **Figure 4**. Some values have already been entered.

[6 marks]

arr			swapsMade	i	t
[0]	[1]	[2]			
4	1	6	false		

Do not write outside the box

0 8 Write a C# program that inputs a character and checks to see if it is lowercase or not.

Your program should work as follows:

- gets the user to enter a character and store it in a suitable variable
- determines if the entered character is a lowercase character
- outputs `LOWER` if the user has entered a lowercase character
- outputs `NOT LOWER` if the user has entered any other character.

You **should** use meaningful variable name(s), correct syntax and indentation in your answer.

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

[7 marks]

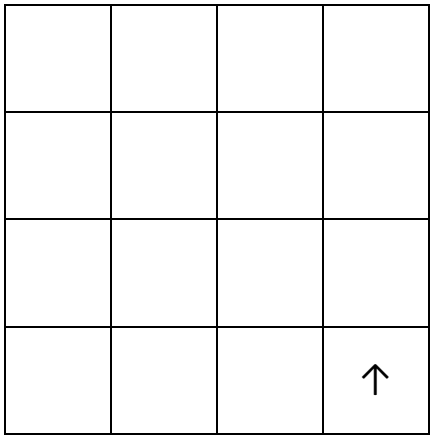
0 9 Four separate subroutines have been written to control a robot.

- `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.

0 9 . 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)
```

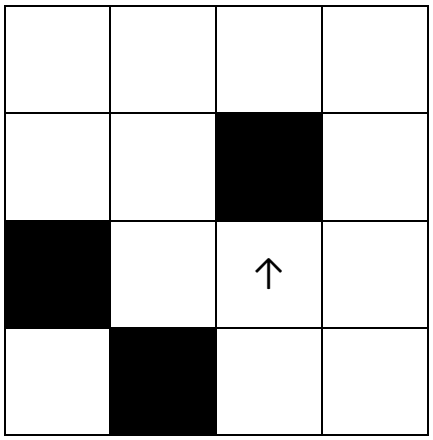
[3 marks]



0 9 . 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 0 State **two** benefits of developing solutions using the structured approach.

[2 marks]

1 1 Fill in the blank arrays to show the steps involved in applying the bubble sort algorithm to the array `[3, 5, 1, 4, 2]`. You only need to show the missing steps where a change is applied to the array.

[5 marks]

3	5	1	4	2
1	2	3	4	5

1 2 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 2 . 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 2 . 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.

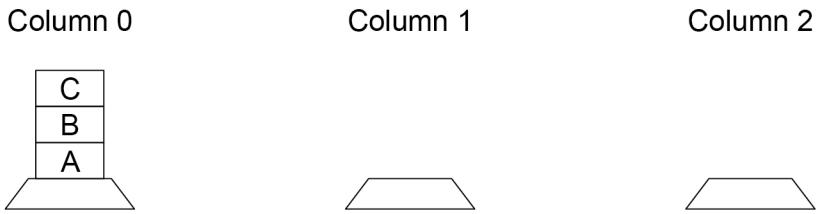
State **two** other relevant details that the developer has missed.

[2 marks]

1 3 The following subroutines control the way that labelled blocks are placed in different columns.

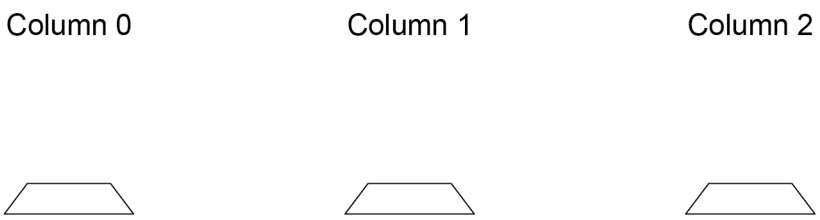
`BLOCK_ON_TOP(column)`	returns the label of the block on top of the column given as a parameter.
`MOVE(source, destination)`	moves the block on top of the `source` column to the top of the `destination` column.
`HEIGHT(column)`	returns the number of blocks in the specified column.

1 3 . 1 This is how the blocks A, B and C are arranged at the start.



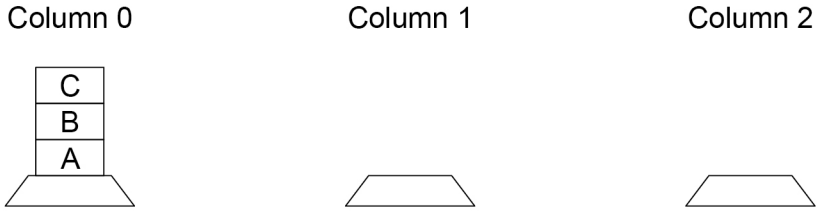
Draw the final arrangement of the blocks after the following algorithm has run.

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



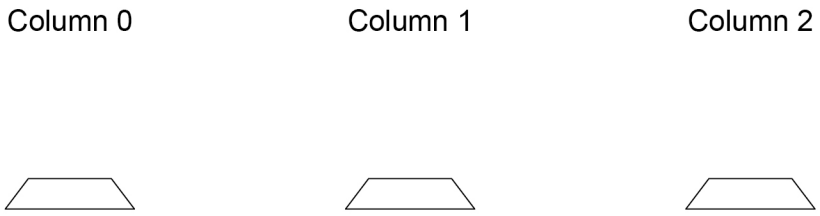
[3 marks]

1 3 . 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

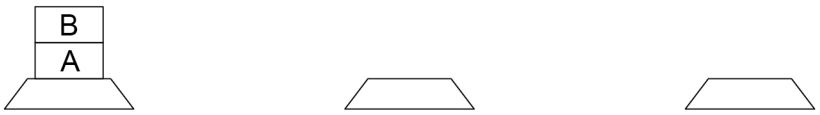
13.3 Develop an algorithm using either pseudo-code or a flowchart that will move every block from column 0 to column 1.

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.

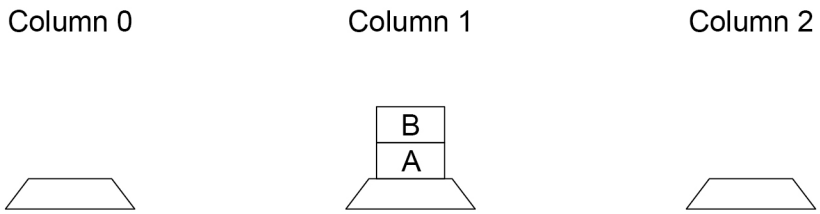
The order of the blocks must be preserved.

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]

Do not write outside the box

Do not write outside the box

1 4 A programmer has written the C# program in **Figure 5** to add up the numbers between one and five.

Figure 5

```
int total = 0;
for (int number = 1; number < 6; number++)
{
  total = total + number;
}
Console.WriteLine(total);
```

The program needs to be changed so that it also multiplies all of the numbers between one and five.

Shade **one** lozenge next to the program that will do what the programmer wants.

[1 mark]

A

```
int total = 0;
int product = 1;
for (int number = 1; number < 6; number++)
{
  total = total + number;
  product = total * number;
}
Console.WriteLine(total);
Console.WriteLine(product);
```

B

```
int total = 0;
int product = 1;
for (int number = 1; number < 6; number++)
{
  total = total + number;
  product = product * number;
}
Console.WriteLine(total);
Console.WriteLine(product);
```

C

```
int total = 0;
int product = 1;
for (int number = 1; number < 6; number++)
{
  total = total + number;
  product = product * total;
}
Console.WriteLine(total);
Console.WriteLine(product);
```

D

```
int total = 0;
int product = 1;
for (int number = 1; number < 6; number++)
{
  total = total + number;
  product = (total + product) * number;
}
Console.WriteLine(total);
Console.WriteLine(product);
```

1 5 A program has been written in C# to display all the odd integers between 1 and the largest odd number smaller than an integer entered by the user. The program is shown in **Figure 6**.

Figure 6

```
int odd = 1;
int number;
Console.Write("Enter an integer: ");
number = Convert.ToInt32(Console.ReadLine());
while (odd != Number)
{
  Console.WriteLine(odd);
  odd = odd + 2;
}
Console.WriteLine("Finished!");
```

The program works correctly if the integer entered by the user is an odd, positive integer. For example, if 7 is entered the program correctly displays the values 1, 3 and 5

The program does not work correctly if an odd integer less than 1 is entered by the user. For example, when −7 is entered the program should display the values 1, -1, -3 and -5 but it doesn't do this.

Using C# only, change the program code inside the while loop so that it will work correctly for any odd integer entered by the user.

[4 marks]

Do not write outside the box

1 6 **Figure 7** shows part of a program written in C#.

Figure 7

```
bool validChoice;
int choice;
validChoice = false;
while (validChoice == false)
{
   Console.Write("Enter your choice [1 - 10] ");
   choice = int.Parse(Console.ReadLine());
   if (choice >= 1 & choice <= 10)
   {
      validChoice = true;
   }
   else
   {
      Console.WriteLine("Invalid choice");
   }
}
Console.WriteLine("Valid choice");
```

Complete the following test plan for the code shown in **Figure 7**.

Test type	Test data	Expected result
Normal data	5	`Valid choice` message displayed
Invalid data		
Boundary data		

[2 marks]

1 7 **Figure 8** shows a C# program that is being developed.

It is supposed to calculate and display the highest common factor of two numbers entered by the user.

The highest common factor of two numbers is the largest number that both numbers can be divided by without leaving a remainder.

Examples:

- the highest common factor of the numbers 6 and 9 is 3
- the highest common factor of 2 and 5 is 1

Line numbers are shown but are not part of the program code.

Figure 8

```
1  int num1 = Convert.ToInt32(Console.ReadLine());
2  int num2 = Convert.ToInt32(Console.ReadLine());
3  int hcf = 1;
4  int count = 1;
5  while (count < num1)
6  {
7    if (num1 % count == 0 && num2 % count == 0)
8    {
9      hcf = count;
10   }
11   count = count + 1;
12 }
13 Console.WriteLine(hcf);
```

The highest common factor of two numbers is the largest number that both numbers can be divided by without leaving a remainder.

Examples:

- the highest common factor of the numbers 6 and 9 is 3
- the highest common factor of 2 and 5 is 1

The program in **Figure 8** works correctly sometimes but not always. When the user enters the numbers 4 and 6 it correctly outputs 2, but when the user enters the numbers 4 and 4 it should output 4 but it does not.

Do not write outside the box

1 7 . 1 State the output from the program in **Figure 8** when the user enters the numbers 4 and 4

[1 mark]

1 7 . 2 State the line number from the program in **Figure 8** which contains the error that stops the program from sometimes working correctly.

[1 mark]

1 7 . 3 Describe how the line of code identified in your answer to **17.2** should be changed so that the program in **Figure 8** will work correctly.

[1 mark]

Turn over for the next question

1 8 Write a C# program that calculates an estimate of the braking distance in metres for a new model of go-kart that is travelling between `10` and `50` kilometres per hour (kph).

Your program should:

- keep asking the user to enter a speed for the go-kart until they enter a speed that is between `10` and `50` (inclusive)
- calculate the braking distance in metres by dividing the speed by 5
- ask the user if the ground is wet (expect the user to enter `yes` if it is)
- if the ground is wet, multiply the braking distance by 1.5
- output the final calculated braking distance.

You **should** use meaningful variable name(s), correct syntax and indentation in your answer.

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

[8 marks]

Do not write outside the box

END OF QUESTIONS

Do not write outside the box

DO NOT WRITE ON THIS PAGE
ANSWER IN THE SPACES PROVIDED