A

AQA

Surname _____

Other Names _____

Centre Number _____

Candidate Number _____

Candidate Signature _____

I declare this is my own work.

GCSE

COMPUTER SCIENCE

Paper 1 Computational thinking and programming skills – C#

8525/1A

Time allowed: 2 hours

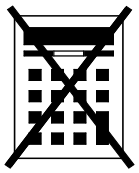
At the top of the page, write your surname and other names, your centre number, your candidate number and add your signature.

[Turn over]

J U N 2 2 8 5 2 5 1 A 0 1

For this paper you must have:
- **an insert.**

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.**
- **If you need extra space for your answer(s), use the lined pages at the end of this book. Write the question number against your answer(s).**
- **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#.**
- **You should assume that all indexing in code starts at 0 unless stated otherwise.**

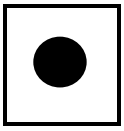
INFORMATION

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

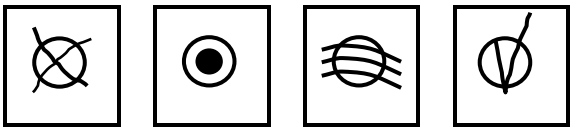
ADVICE

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

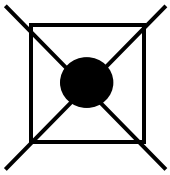
CORRECT METHOD



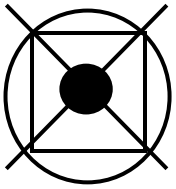
WRONG METHODS



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 TURN OVER UNTIL TOLD TO DO SO

Answer ALL questions.

0 1

An algorithm, that uses the modulus operator, has been represented using pseudo-code in FIGURE 1, provided on page 2 of the insert.

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

The modulus operator is used to calculate the remainder after dividing one integer by another.

For example:

- `14 MOD 3` **evaluates to** `2`
- `24 MOD 5` **evaluates to** `4`

0 1 . 1

Shade ONE lozenge that shows the line number where selection is FIRST used in the algorithm in FIGURE 1. [1 mark]

○ **A Line number 1**

○ **B Line number 2**

○ **C Line number 3**

○ **D Line number 4**

[Turn over]

0 1 . 2

Shade ONE lozenge that shows the output from the algorithm in FIGURE 1 when the user input is 4
[1 mark]

○ **A** 0

○ **B** 2

○ **C** 4

○ **D** 8

○ **E** 16

0 1 . 3

Shade ONE lozenge that shows the line number where assignment is FIRST used in the algorithm in FIGURE 1. [1 mark]

○ **A Line number 1**

○ **B Line number 2**

○ **C Line number 3**

○ **D Line number 4**

[Turn over]

0 1 . 4

Shade ONE lozenge that shows the line number that contains a relational operator in the algorithm in FIGURE 1. [1 mark]

○ **A Line number 1**

○ **B Line number 2**

○ **C Line number 3**

○ **D Line number 4**

0 1 . 5

Shade ONE lozenge to show which of the following is a TRUE statement about the algorithm in FIGURE 1, provided on page 2 of the insert. [1 mark]

○ **A This algorithm uses a Boolean operator.**

○ **B This algorithm uses a named constant.**

○ **C This algorithm uses iteration.**

○ **D This algorithm uses the multiplication operator.**

[Turn over]

0 1 . 6

FIGURE 2, provided on page 3 of the insert, shows an implementation of the algorithm in FIGURE 1, provided on page 2 of the insert, using the C# programming language.

- **Line numbers are included but are not part of the program.**

The program in FIGURE 2 needs to be changed so that it repeats five times using DEFINITE (count controlled) iteration.

Shade ONE lozenge, on pages 10 to 13, next to the program that does this correctly. [1 mark]

○ **A**

```
for (int x = 0; x < 5; x++) {
    Console.Write("Enter a number: ");
    int i = Convert.ToInt32(Console.ReadLine());
    if (i % 2 == 0) {
        Console.WriteLine(i * i);
    }
    else {
        Console.WriteLine(i);
    }
}
```

○ **B**

```
for (int x = 0; x < 6; x++) {
    Console.Write("Enter a number: ");
    int i = Convert.ToInt32(Console.ReadLine());
    if (i % 2 == 0) {
        Console.WriteLine(i * i);
    }
    else {
        Console.WriteLine(i);
    }
}
```

○

C

```
int x = 1;
while (x != 6) {
    Console.Write("Enter a number: ");
    int i = Convert.ToInt32(Console.ReadLine());
    if (i % 2 == 0) {
        Console.WriteLine(i * i);
    }
    else {
        Console.WriteLine(i);
    }
    x = x + 1;
}
```

D

```
int x = 6;
while (x != 0) {
    Console.Write("Enter a number: ");
    int i = Convert.ToInt32(Console.ReadLine());
    if (i % 2 == 0) {
        Console.WriteLine(i * i);
    }
    else {
        Console.WriteLine(i);
    }
    x = x - 1;
}
```

0 2

FIGURE 3, provided on pages 4 and 5 of the insert, shows an algorithm, represented using pseudo-code, that calculates the delivery cost for an order from a takeaway company.

0 2 . 1

Using FIGURE 3, complete the table. [2 marks]

Input value of `orderTotal`	**Input value of** `deliveryDistance`	**Output**
`55.5`	`2`	
`35.0`	`5`	

0 2 . 2

State how many possible values the result of the comparison `deliveryDistance ≤ 5` **could have in the algorithm shown in FIGURE 3. [1 mark]**

0 2 . 3

State the most suitable data type for the following variables used in FIGURE 3. [2 marks]

Variable identifier	Data type
`deliveryCost`	
`messageOne`	

0 2 . 4

State ONE other common data type that you have NOT given in your answer to Question 02.3. [1 mark]

_____

_____

[Turn over]

0 3

FIGURE 4, provided on pages 6 and 7 of the insert, shows a C# program that calculates car park charges.

The user inputs their car registration (eg `MA19 GHJ`**) and the length of the stay.**

The program then outputs the charge.

- **Line numbers are included but are not part of the program.**

0 3 . 1

Rewrite LINE 5 in FIGURE 4 to CONCATENATE the car registration with the string `" is not valid"`**, and store the result in the variable** `displayMessage`**.**

Your answer must be written in C#. [1 mark]

0 3 . 2

The charge for parking for two or more hours is changed to include an additional £2 fee.

Rewrite LINE 15 in FIGURE 4 to show this change.

Your answer must be written in C#. [1 mark]

_____

_____

[Turn over]

8

0 4

The two C# programs in FIGURE 5, provided on pages 8 and 9 of the insert, output the value that is equivalent to adding together the integers between 1 and an integer entered by the user.

For example, if the user entered the integer `5`**, both programs would output** `15`

0 4 . 1

Shade ONE lozenge to indicate which of the statements is true about the programs in FIGURE 5. [1 mark]

○ **A Both programs are equally efficient.**

○ **B Program A is more efficient than Program B.**

○ **C Program B is more efficient than Program A.**

0 4 . 2

Justify your answer for Question 04.1. [2 marks]

[Turn over]

0 5

A programmer has started to write a program using C#. Their program is shown in FIGURE 6, provided on page 10 of the insert.

The program should generate and output 10 numbers, each of which is randomly selected from the numbers in a data structure called `numbers`**.**

The program uses the `Random` **class.**

For example, `r.Next(0, 8)` **would generate a random integer between 0 and 7 inclusive.**

One possible output from the finished program would be 11, 14, 14, 42, 2, 56, 56, 14, 4, 2

- **Line numbers are included but are not part of the program.**

0 5 . 1

The program shown in FIGURE 6 contains a syntax error.

Shade TWO lozenges to indicate the statements that are true about syntax errors. [2 marks]

- ○ **A A syntax error can be found by testing boundary values in a program.**
- ○ **B A syntax error is a mistake in the grammar of the code.**
- ○ **C A syntax error is generally harder to spot than a logic error.**
- ○ **D A syntax error will stop a program from running.**
- ○ **E An example of a syntax error is trying to access the fifth character in a string which only contains four characters.**

[Turn over]

0 5 . 2

The program shown in FIGURE 6 also contains a logic error.

Identify the line number that contains the logic error, and correct this line of the program.

Your corrected line must be written in C#. [2 marks]

Line number _____

Corrected line _____

_____

_____

0 5 . 3

What type of data structure is the variable `numbers`**? [1 mark]**

_____

_____

8

0 6

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.

0 6 . 1

Define the term ABSTRACTION. [1 mark]

0 6 . 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, provided on page 12 of the insert, 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]

A _____

_____

B _____

_____

BLANK PAGE

[Turn over]

0 7

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, on pages 27 to 31, contains vertical lines to help you indent your code. [5 marks]

[Turn over]


3 0

[Turn over]

8

0 8

Write a C# program that calculates the value of a bonus payment for an employee based on how many items they have sold and the number of years they have been employed.

The program should:

- **get the user to input the number of items sold**
- **get the user to input the number of years employed**
- **output the value of the bonus payment:**
  - **if the years of employment is less than or equal to 2 AND the number of items sold is greater than 100, then the bonus will be the number of items sold multiplied by 2**
  - **if the years of employment is greater than 2, then the bonus will be the number of items sold multiplied by 10**
  - **otherwise, the bonus is 0**

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

The answer grid, on pages 33 to 37, contains vertical lines to help you indent your code. [7 marks]

[Turn over]


3 4


3 7

0 9

FIGURE 8, provided on page 13 of the insert, shows an algorithm represented using pseudo-code.

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

`SUBSTRING` **returns part of a string.**

For example, `SUBSTRING(3, 5, 'programming')` **will return the string** `'gra'`**.**

0 9 . 1

Shade ONE lozenge which shows the output of LINE 4 from the algorithm shown in FIGURE 8. [1 mark]

○ **A** F

○ **B** Freddie

○ **C** Lily

○ **D** S

○ **E** Sarah

[Turn over]

0 9 . 2

Shade ONE lozenge which shows the output of LINE 5 from the algorithm shown in FIGURE 8. [1 mark]

○ **A** 1

○ **B** 2

○ **C** 4

○ **D** 5

○ **E** 10

0 9 . 3

State the output of LINE 7 from the algorithm shown in FIGURE 8. [1 mark]

0 9 . 4

Two extra lines are being added to the end of the algorithm in FIGURE 8.

Fill in the gaps so the output from the new final line will be the string `'Thomasrah'`. [2 marks]

```
var ← SUBSTRING( _____ , _____ , name1)

OUTPUT names[ _______ ] + var
```

[Turn over]

12

1 0

FIGURE 9, provided on page 14 of the insert, shows a subroutine represented using pseudo-code.

The `DIV` **operator is used for integer division.**

1 0 . 1

On the opposite page, complete the trace table for the subroutine call `calculate(50)`

You may not need to use all the rows in the table.
[4 marks]

n	a	b	OUTPUT
50			

[Turn over]

1 0 . 2

State the value that will be output for the subroutine call `calculate(1)` **[1 mark]**

1 0 . 3

The identifier for the variable `b` **in FIGURE 9 was not a good choice.**

State a better identifier for this variable that makes the algorithm easier to read and understand.
[1 mark]

BLANK PAGE

[Turn over]

1 0 . 4

A `REPEAT…UNTIL` **iteration structure was used in FIGURE 9.**

FIGURE 10, provided on page 15 of the insert, shows another subroutine called `calculate` **that uses a** `WHILE…ENDWHILE` **iteration structure.**

One difference in the way the subroutines in FIGURE 9 and FIGURE 10 work is:

- **the** `REPEAT...UNTIL` **iteration structure in FIGURE 9 loops until the condition is true**
- **the** `WHILE...ENDWHILE` **iteration structure in FIGURE 10 loops until the condition is false.**

Describe TWO other differences in the way the subroutines in FIGURE 9 and FIGURE 10 work.
[2 marks]

1 _____

_____

_____

_____

2 _____

_____

_____

_____

[Turn over]

8

1 1 . 1

The size of a sound file is calculated using the following formula:

size (in bits) = sampling rate * sample resolution * seconds

To calculate the size IN BYTES, the number is divided by 8

The algorithm in FIGURE 12, represented using pseudo-code, should output the size of a sound file in BYTES that has been sampled 100 times per second, with a sample resolution of 16 bits and a recording length of 60 seconds.

A subroutine called `getSize` has been developed as part of the algorithm.

Complete FIGURE 12, on the opposite page, by filling in the gaps using the items in FIGURE 11, provided on page 16 of the insert.

You will not need to use all the items in FIGURE 11. [6 marks]

FIGURE 12

```
SUBROUTINE getSize(_____,_____, seconds)

    _____ ← sampRate * res * seconds

    size ← _____

    _____ size

ENDSUBROUTINE


OUTPUT _____(100, 16, 60)
```

[Turn over]

1 1 . 2

A local variable called `size` **has been used in** `getSize`**.**

Explain what is meant by a local variable in a subroutine. [1 mark]

1 1 . 3

State THREE advantages of using subroutines.
[3 marks]

1 _____

_____

_____

_____

2 _____

_____

_____

_____

3 _____

_____

_____

_____

[Turn over]

10

1 2

FIGURE 13, provided on page 17 of the insert, shows an algorithm represented in pseudo-code. A developer wants to check the algorithm works correctly.

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

1 2 . 1

On the opposite page, complete the trace table for the algorithm shown in FIGURE 13.

Some values have already been entered. You may not need to use all the rows in the table. [6 marks]

arr			i	j	temp
[0]	[1]	[2]			
c	b	a			

[Turn over]

1 2 . 2

State the purpose of the algorithm. [1 mark]

1 2 . 3

An earlier attempt at writing the algorithm in FIGURE 13 had different code for LINES 4 and 5.

LINES 4 and 5 of the pseudo-code were:

```
FOR i ← 0 TO 2
    FOR j ← 0 TO 2
```

Explain why the algorithm did not work when the value `2` was used instead of the value `1` on these two lines.
[1 mark]

8

1 3

A program is being developed in C# to simulate a card game.

Throughout the game each player always has 100 cards. Each card displays a number.

Players take it in turns to swap one of their cards with another random card from a set of cards until a player has a run of five numbers in sequence within their 100 cards.

1 3 . 1

FIGURE 14 shows part of the program that will get a player to enter the position of a card to swap.

FIGURE 14

```
Console.Write("Enter card position: ");

int position= Convert.ToInt32(Console.ReadLine());
```

Extend the program in FIGURE 14. Your answer must be written in C#.

The program should keep getting the user to enter the card position until they enter a card position that is between 1 and 100 inclusive.

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. [4 marks]

1 3 . 2

There are 500 cards within the game in total. Each card is numbered from 1 to 250 and each number appears twice in the whole set of cards.

The player's 100 cards are always stored in numerical order.

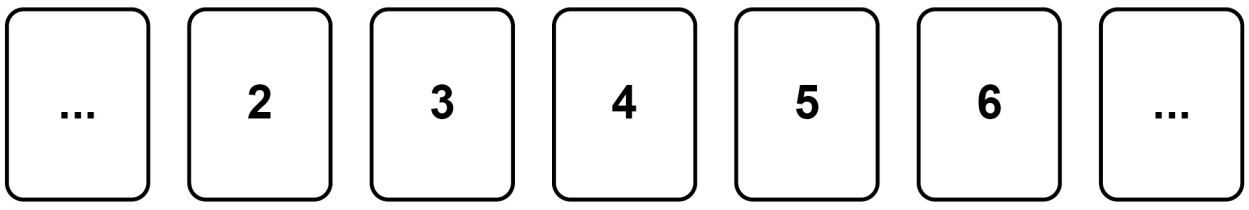
When a player has a valid run of five cards within their 100 cards they have won the game.

A valid run:

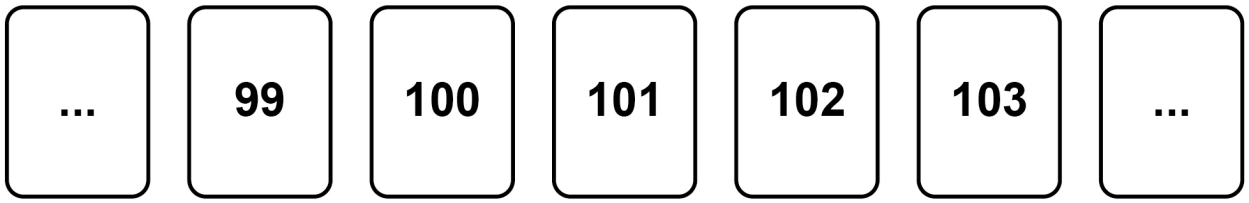
- **consists of five cards**
- **can start from any position in the player's 100 cards**
- **the second card's value is one more than the first card's value, the third card's value is one more than the second card's value, the fourth card's value is one more than the third card's value, and the fifth card's value is one more than the fourth card's value.**

Below are examples of valid runs which means a player has won.

VALID RUN EXAMPLE 1

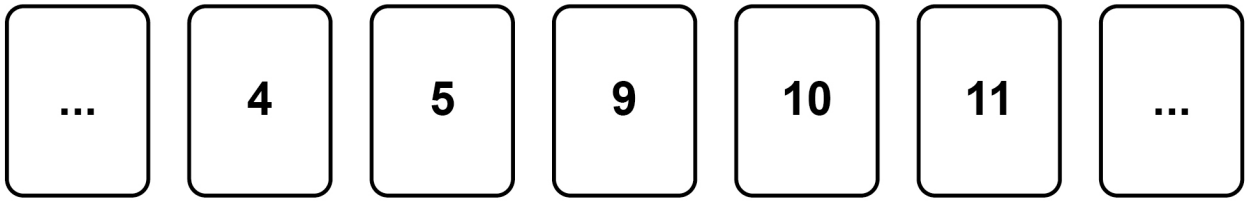


VALID RUN EXAMPLE 2

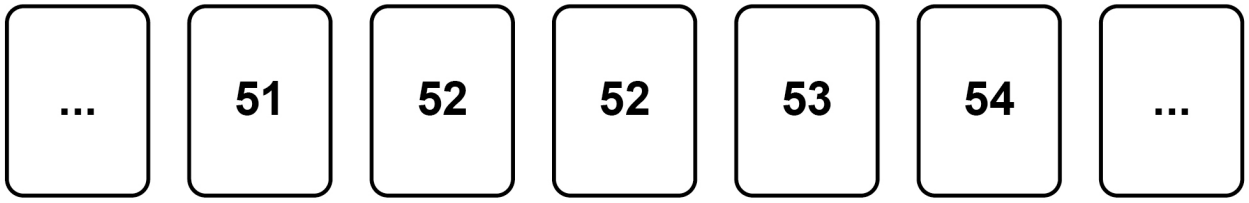


Below are examples of invalid runs.

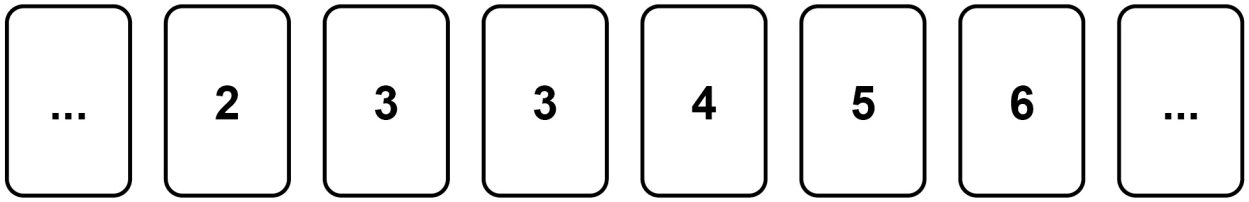
INVALID RUN EXAMPLE 1



INVALID RUN EXAMPLE 2



INVALID RUN EXAMPLE 3



[Turn over]

Write a C# program to check if a player has a valid run of five cards within their 100 cards.

When writing your program you should assume:

- **there is an array called** `cards` **that contains the values of the player's 100 cards**
- `cards[0]` **will contain the value of the first card and** `cards[99]` **will contain the value of the last card**
- **the values in** `cards` **are already stored in numerical order**
- **there is a Boolean variable called** `gameWon` **that has a value of** `False`.

Your program should set `gameWon` **to** `True` **if there is a valid run.**

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

The answer grid, on pages 61 to 64, contains vertical lines to help you indent your code. [6 marks]

[Turn over]


6 2

[Turn over]

10

BLANK PAGE

[Turn over]

1 4

A program is being written to simulate a computer science revision game in the style of bingo.

At the beginning of the game a bingo ticket is generated with nine different key terms from computer science in a 3 x 3 grid. An example bingo ticket is provided in FIGURE 15, provided on page 17 of the insert.

The player will then be prompted to answer a series of questions.

If an answer matches a key term on the player's bingo ticket, then the key term will be marked off automatically.

14.1

FIGURE 16, on pages 68 and 69, shows an incomplete C# program to create a bingo ticket for a player.

The programmer has used a two-dimensional array called `ticket` to represent a bingo ticket.

The program uses a subroutine called `generateKeyTerm`. When called, the subroutine will return a random key term, eg `"CPU"`, `"ALU"`, `"NOT gate"` etc.

Complete the C# program in FIGURE 16, on pages 68 and 69, by filling in the five gaps.

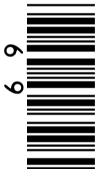
- **Line numbers are included but are not part of the program.**

[4 marks]

[Turn over]

FIGURE 16

```
1 string[,] ticket = new string[,] {{"","",""},
                                    {"","",""},
                                    {"","",""}};

2 int i = 0;

3 while (i < 3) {

4     int j = _____ ;

5     while (j < 3) {

6       ticket[_____ ,_____] = generateKeyTerm();

7       _____ ;

8     }
```



```
9           _____;
10  }
```

[Turn over]

1 4 . 2

Each time a player answers a question correctly the `ticket` **array is updated; if their answer is in the** `ticket` **array then it is replaced with an asterisk (*).**

An example of the `ticket` **array containing key terms and asterisks is shown in FIGURE 17, on page 18 of the insert.**

Write a subroutine in C# called `checkWinner` **that will count the number of asterisks.**

The subroutine should:

- **take the** `ticket` **array as a parameter**
- **count the number of asterisks in the** `ticket` **array**
- **output the word** `Bingo` **if there are nine asterisks in the array**
- **output the total number of asterisks if there are fewer than nine asterisks in the array.**

You MUST write your own count routine and not use any built-in count function that might be available in C#.

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

The answer grid, on pages 71 to 73, contains vertical lines to help you indent your code.
[8 marks]

[Turn over]


7 2

END OF QUESTIONS

12

Additional page, if required.
Write the question numbers in the left-hand margin.

Additional page, if required.
Write the question numbers in the left-hand margin.

BLANK PAGE

For Examiner's Use	
Question	Mark
1	
2–3	
4–5	
6–7	
8–9	
10	
11	
12	
13	
14	
TOTAL	