



Mark Scheme (Results)

November 2020

Pearson Edexcel International GCSE
In Computer Science (4CP0/2C)
Paper 2: Application of Computational
Thinking

Edexcel and BTEC Qualifications

Edexcel and BTEC qualifications are awarded by Pearson, the UK's largest awarding body. We provide a wide range of qualifications including academic, vocational, occupational and specific programmes for employers. For further information visit our qualifications websites at www.edexcel.com or www.btec.co.uk. Alternatively, you can get in touch with us using the details on our contact us page at www.edexcel.com/contactus.

Pearson: helping people progress, everywhere

Pearson aspires to be the world's leading learning company. Our aim is to help everyone progress in their lives through education. We believe in every kind of learning, for all kinds of people, wherever they are in the world. We've been involved in education for over 150 years, and by working across 70 countries, in 100 languages, we have built an international reputation for our commitment to high standards and raising achievement through innovation in education. Find out more about how we can help you and your students at: www.pearson.com/uk

Autumn 2020

Publications Code 4CP0_2C_2011_MS

All the material in this publication is copyright

© Pearson Education Ltd 2020

General Marking Guidance

- All candidates must receive the same treatment. Examiners must mark the first candidate in exactly the same way as they mark the last.
- Mark schemes should be applied positively. Candidates must be rewarded for what they have shown they can do rather than penalised for omissions.
- Examiners should mark according to the mark scheme not according to their perception of where the grade boundaries may lie.
- There is no ceiling on achievement. All marks on the mark scheme should be used appropriately.
- All the marks on the mark scheme are designed to be awarded. Examiners should always award full marks if deserved, i.e. if the answer matches the mark scheme. Examiners should also be prepared to award zero marks if the candidate's response is not worthy of credit according to the mark scheme.
- Where some judgement is required, mark schemes will provide the principles by which marks will be awarded and exemplification may be limited.
- When examiners are in doubt regarding the application of the mark scheme to a candidate's response, the team leader must be consulted.
- Crossed out work should be marked UNLESS the candidate has replaced it with an alternative response.

Mark Scheme - Theory

Question	mp	Answer	Additional Guidance	Mark
1 (a)	A1	C (equals) (1)		(1)

Question	mp	Answer	Additional Guidance	Mark
1 (b)	B1	Award one mark for any of: <ul style="list-style-type: none"> • TRUE and FALSE (1) • Yes and No (1) • T and F (1) • 0 and 1 (1) 	Ignore capitalisation	(1)

Question	mp	Answer	Additional Guidance	Mark
1 (d)	D1	Award one mark for any of: <ul style="list-style-type: none"> • (Symbolic name associated with) a value that may be changed (1) • A container used to store data (1) • A data store whose contents can change whilst a program is executing (1) 		(1)

Question	mp	Answer	Additional Guidance	Mark
1 (e) (i)	E1	Python 6 C# 12 Java 8		(1)
1 (e) (ii)	E2	Python 3 C# 7 Java 3		(1)
1 (e) (iii)	E3	Award one mark for any of: <ul style="list-style-type: none"> • amount (1) • x (1) • total (1) • count (1) 	Accept listNumbers or numArray	(1)
1 (e) (iv)	E4	message		(1)

Question	mp	Answer	Additional Guidance	Mark
3 (a)	A1	<p>Award one mark for any of:</p> <ul style="list-style-type: none"> • Conversion of plain text into cipher (1) • Converting information/data into a code (1) • Converting information/data into an unreadable format (1) 	Accept alternative wording	(1)

Question	mp	Answer	Additional Guidance	Mark
3 (b)	B1	<p>Award one mark for any of:</p> <ul style="list-style-type: none"> • To ensure that the data can only be read by an authorised person / can't be read by an unauthorised person. (1) 	Accept alternative similar wording. Do not accept hacking.	(1)

Question	mp	Answer	Additional Guidance	Mark
3 (c) (i)	C1 C2 C3 C4	<p>Award 4 marks for a correct response:</p> <ul style="list-style-type: none"> • CAL OTTA MUIN PO • PO MUIN OTTA CAL <p>Award 1 mark each up to a maximum of 3 for:</p> <ul style="list-style-type: none"> • 4 lines of text (1) • Zigzag arrangement of letters (1) • Letters from each line reproduced / consistent use of the key 4: GSQTYXEXMSREP/YKILQPWPEKJWH (1) 		(4)

Question	mp	Answer	Additional Guidance	Mark
3 (c) (ii)	C5 C6	<p>Award 2 marks for a linked explanation such as:</p> <ul style="list-style-type: none"> • Limited number of usable keys (1) to allow for sufficient movement of characters / so can easily be decoded by trial 		(2)

		and error / easy to use brute force to crack (1)		
--	--	---	--	--

Question	mp	ref	Answer	Additional Guidance	Mark								
5 (a)	A1 A2 A3 A4	2.1.4	<p>Award 1 mark for each appropriate validation test (up to 2) and 1 mark for suitable example of erroneous data that matches the test e.g.</p> <table border="1"> <tr> <td>First 3 characters are upper case letters</td> <td>Fff123456o</td> </tr> <tr> <td>Characters 4 to 9 are non-zero numbers</td> <td>FFF101111o</td> </tr> <tr> <td>Consists of only 10 characters</td> <td>FFF111111ee</td> </tr> <tr> <td>Final character correct for sum of numbers</td> <td>FFF112233o</td> </tr> </table>	First 3 characters are upper case letters	Fff123456o	Characters 4 to 9 are non-zero numbers	FFF101111o	Consists of only 10 characters	FFF111111ee	Final character correct for sum of numbers	FFF112233o		(4)
First 3 characters are upper case letters	Fff123456o												
Characters 4 to 9 are non-zero numbers	FFF101111o												
Consists of only 10 characters	FFF111111ee												
Final character correct for sum of numbers	FFF112233o												

Question	mp	ref	Answer	Additional Guidance	Mark
5 (b) (i)	B1 B2	1.1.9	<p>Award 1 mark each for any of:</p> <ul style="list-style-type: none"> • Simple implementation (1) • Can be used for sorted or unsorted lists (1) • If the target is at the beginning of the data structure the search will be faster than a binary search 	Accept disadvantages of a binary compared with a linear search.	(2)
5 (b) (ii)	B3 B4	1.1.9	<p>Award 1 mark each for any of:</p> <ul style="list-style-type: none"> • May need to compare with all items in list before search complete (1) • May need longer time for searching a large list (1) 	Accept advantages of a binary compared with a linear search.	(2)

Question	mp	ref	Answer	Additional Guidance	Mark
5 (c) (i)	C1 C2	2.6.2	<p>Award 1 mark each up to a maximum of 2 for:</p> <ul style="list-style-type: none"> • Both can use arguments / parameter passing (1) • Both can use local variables (1) • Both can be called from anywhere within the program (1) • Code can be reused without being rewritten (1) • Both can be library files (1) • The code can be independently tested for both (1) 		(2)
5 (c) (ii)	C3 C4	2.6.2	<p>Award two marks for a linked explanation such as:</p> <ul style="list-style-type: none"> • A function must always return a result (1) whereas a procedure does not (1) • A function interface must have a data type (1) to signify the type of data that will be returned / a procedure does not need this (1) • The result of a function must always be used (either assigned to a variable or as part of a condition) (1) whereas a procedure does not explicitly return a result to be used (1) • A function produces information (1) whereas a procedure performs a task (1) 		(2)

Mark Scheme – Java Coding

Question	mp	ref	Answer	Additional Guidance	Mark
1 (c)	C1	2.1.5	Delete space in variable name (1)		(3)
	C2	2.1.5	Capitalise O in variable name (1)		
	C3	2.1.5	Correction to print statement (1)		
Code example					
Java	<pre>4 { 5 int numOne = 25; 6 int numTwo = 36; 7 int numThree = numOne + numTwo; 8 System.out.println(numThree); 9 }</pre>				

Question	mp	ref	Answer	Additional Guidance	Mark
2 (a)	A1	2.3.4	Set variable base to 50 or heightChk to true	Logic of algorithm must be followed as set out. Alternatives must address each point. Do not penalise candidates who attempt more than the stated requirements. Don't penalise spelling mistakes and alternative wording of the output.	(10)
	A2	2.2.2	Create WHILE DO loop		
	A3	2.4.1	Request input of height		
	A4	2.2.2	if statement checks if height ≥ 1 or ≤ 100		
	A5	2.2.2	if statement checks if height is ≥ 1 and ≤ 100		
	A6	2.2.2	heightCheck set to false if condition is met		
	A7	2.5.1	Set area using given formula ($0.5 \times \text{base} \times \text{height}$)		
	A8	2.4.1	Display one of base, height and area with appropriate label or displays two of base, height and area without labels		
	A9	2.4.1	Display all of base, height and area with appropriate labels		
	A10	1.1.6	Executing and producing correct output		

Code example

Java

```
1  import java.util.Scanner;
2
3  public class q02a
4  {
5      public static void main(String[] args)
6      {
7          // Initialise variables
8          int base = 50;
9          int height = 0;
10
11         // Display prompt and take input from user
12         Boolean heightChk= true;
13         Scanner input = new Scanner(System.in);
14         while (heightChk)
15         {
16             System.out.print("Enter height of triangle : ");
17             height = Integer.parseInt(input.next());
18             if (height >= 1 && height <= 100)
19             {
20                 heightChk = false;
21             }
22         }
23
24         // Calculate and print out values
25
26         double area = 0.5 * base * height;
27
28         System.out.println("The length of the triangle is : " + base);
29         System.out.println("Height of the triangle is : " + height);
30         System.out.println("Area of the triangle is : " + area);
31     }
32 }
```

Question	mp	ref	Answer	Additional Guidance	Mark
2 (b)	B1	2.2.1	Meaningful variable names used (1)	The code example shown is one way of responding to the task. Other methods should be credited accordingly.	(7)
	B2	2.4.1	Meaningful prompts for input (1)		
	B3	2.4.1	Input of length and width as whole numbers (1)		
	B4	2.5.1	Perimeter calculated (= 2 x length + 2 x width) (1)		
	B5	2.5.1	Gap deducted from perimeter (1)		
	B6	2.4.1	Meaningful output (1)		
	B7	2.1.1	Code is fit for purpose. Must have mark points 1-6 and be easy to read (1)		

Code example

Java

```

1  import java.util.Scanner;
2
3  public class q02b
4  {
5      public static void main(String[] args)
6      {
7          // Set initial values of variables
8          Scanner input = new Scanner(System.in);
9          int length;
10         int width;
11
12         // Request input
13
14         System.out.print("Enter the length: ");
15         length = Integer.parseInt(input.next());
16         System.out.print("Enter the width: ");
17         width = Integer.parseInt(input.next());
18
19         // Calculate number of panels
20
21         double panels = length * 2 + width * 2 - 4;
22
23         // Print out number of panels needed
24
25         System.out.print("Number of panels required = " + panels);
26     }
27 }

```

Question	mp	ref	Answer	Additional Guidance	Mark
4 (a)	A1	2.5.2	english < 40, maths < 50 (1)		
	A2	2.5.3	Use of and operator in IF (1)		
	A3	2.5.2	english < 40, maths < 50 (1)		
	A4	2.5.3	Use of or operator in 'else if 1' (1)		
	A5	2.5.2	english >= 80 AND maths >=85 (1)		
	A6	2.5.3	Correct completion of final else with messageIndex = 3 (1)		

(6)

Code example

Java

```

41 System.out.print(firstName + " " + lastName + ": ");
42 if (english < 40 && maths < 50) //student failed both
43 {
44     messageIndex = 0;
45 }
46
47 else if (english < 40 || maths < 50) // student failed
48 {
49     messageIndex = 1;
50 }
51
52 else if (english >= 80 && maths >= 85) // student pass
53 {
54     messageIndex = 2;
55 }
56
57 else
58 {
59     messageIndex = 3; // student passed both tests
60 }

```

Question	mp	ref	Answer	Additional Guidance	Mark
4 (b)	B1	2.2.2	Loop checks each pupil for low attendance		
	B2	2.1.6	Display of name of low attendances		
	B3	2.3.1	Counter for high attenders initialised		
	B4	2.2.2	Loop uses correct comparison (\geq)		
	B5	2.6.2	Number of high attenders displayed correctly		
	B6	2.6.3	Subprogram for option 1 called correctly		
	B7	2.6.3	Subprogram for option 2 called correctly		
					(7)

Code example

Java

```
// write subprogram for option 1 here
private static void lowAttendees(String [][] myArray)
{
    for (String[] student : myArray)
    {
        int x = Integer.parseInt(student[2]);
        if (x < 75)
        {
            System.out.println(student[0] + " " + student[1]);
        }
    }
}

// write subprogram for option 2 here

private static int highAttendees(String [][] myArray)
{
    int count = 0;
    for (String[] student : myArray)
    {
        int x = Integer.parseInt(student[2]);
        if (x >= 90)
        {
            count++;
        }
    }
    return count;
}
```

Faroukh Salah
Amara Grzinski
Taz Grimstow
Sadia Bhatti
Fernado Askabat
Siyao Wang

```
64     if (option == 1)
65     {
66         // complete the if statement
67         lowAttendees(pupilAttendance);
68     }
69
70     else if (option == 2)
71     {
72         // complete the else if statement
73         int highAtt = highAttendees(pupilAttendance);
74         System.out.println("Number of high attenders = " + highAtt);
75     }
```

There were 7 high attenders

For Q6, the first 11 marks are for coding that matches requirements of task. The remaining 9 marks should be allocated on a best fit.

Question	mp	ref	Answer	Additional Guidance	Mark
6	A1	2.2.1	Initialise variable for total sales by member of staff (1)	- -- Sales for each member of staff	
	A2	2.5.1	Add succeeding months' sales to total sales (1)	George Taylor 28466	
	A3	2.1.6	Display total sales by member of staff (1)	Fehlix Chayne 35250	
	A4	2.2.2	Loop repeats for each member of staff (1)	Sumren Bergen 30186	
	A5	2.2.1	Running team total sales initialised (1)	Samira Beckle 37430	
	A6	2.2.2	Add succeeding individual staff sales to team total (1)	Nellie Bogart 20448	
	A7	2.1.6	Display final running total (1)	Cheryl Grouth 34383	
	A8	2.1.6	Staff member with highest individual sales identified (1)	Danuta Graunt 26421	
	A9	2.1.6	Staff member with second highest individual sales identified (1)	Jaiden Deckle 21507	
	A10	2.1.6	Information about staff with highest sales displayed (1)	Jimran Caliks 32415	
	A11	2.1.6	Information about staff with second highest sales displayed (1)	Deynar Derran 28590 =====	
				Total staff sales: 295096	
				Highest sales by: Samira Beckle with 37430	
				Second highest sales by: Fehlix Chayne with 35250	

(11)

Band 1 (1-3 marks)	Band 2 (4-6 marks)	Band 3 (7-9 marks)	Mark
Little attempt to decompose the problem into component parts	Some attempt to decompose the problem into component parts	The problem has been decomposed into component parts	
Some parts of the logic are clear and appropriate to the problem	Most parts of the logic are clear and mostly appropriate to the problem	The logic is clear and appropriate to the problem	
Some appropriate use and manipulation of data types, variables, data structures and program constructs	The use and manipulation of data types, variables and data structures and program constructs is mostly appropriate	The use and manipulation of data types, variables and data structures and program constructs is appropriate	
Parts of the code are clear and readable	Code is mostly clear and readable	Code is clear and readable	
Finished program will not be flexible enough with other data sets or input	Finished program will function with some but not all other data sets or input	Finished program could be used with other data sets or input	
The program meets some of the given requirements	The program meets most of the given requirements	The program fully meets the given requirements	
			(9)

Code example

Java

```
23     int allStaffTotal = 0;
24     int highestSales = 0;
25     int secondSales = 0;
26     int high = 0;
27     int second = 0;
28     int staffTotal;
29     int sales;
30     int staffNo = -1;
31
32     System.out.println("Sales for each member of staff");
33     for (String [] staff : staffSales)
34     {
35         staffTotal = 0;
36         sales = 3;
37         staffNo += 1;
38         while (sales < staff.length)
39         {
40             staffTotal = staffTotal + Integer.parseInt(staff[sales]);
41             sales += 1;
42         }
43         System.out.println(staff[1] + " " + staff[2] + " " + staffTotal);
44
45         if (staffTotal > secondSales)
46         {
47             if (staffTotal > highestSales)
48             {
49                 secondSales = highestSales;
50                 highestSales = staffTotal;
51                 second = high;
52                 high = staffNo;
53             }
54             else
55             {
56                 secondSales = staffTotal;
57                 second = staffNo;
58             }
59         }
60         allStaffTotal = allStaffTotal + staffTotal;
61     }
62     System.out.println();
63     System.out.println("Total staff sales = " + allStaffTotal);
64     System.out.println("Highest staff sales by " + staffSales[high][1] + " " + staffSales[high][2] + " with " + highestSales);
65     System.out.println("Second staff sales by " + staffSales[second][1] + " " + staffSales[second][2] + " with " + secondSales);
```