Pearson
Edexcel

Examiners' Report

Principal Examiner Feedback

Summer 2025

Pearson Edexcel International GCSE

In Computer Science (4CP0)

Paper 2

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

Summer 2025
Publications Code 4CP0_02_2506_ER

Introduction

The responses seen from candidates this year were of a generally high standard particularly with the coding element of the paper. There was a further reduction in the number of centres using Java and C#, with almost all candidates producing Python code.

File handling and string manipulation are still areas that many candidates struggle with, however it was encouraging to see that most candidates attempted file handling in Q5, even if they did not complete it accurately, and that the vast majority of candidates made a reasonable attempt to answer Q6. Some candidates also found the concept of post-conditional loops a challenge as seen in Q3(c).

The written parts of the paper were also generally well answered, though often only the more able candidates were able to link responses appropriately for the 2-mark and 3-mark questions. Exam technique remains an area for improvement for many candidates. A common misconception on the operation of a bubble sort was evident and is described in more detail below.

REPORT ON INDIVIDUAL WRITTEN QUESTIONS

- There were two multiple-choice questions included in the question paper. 1(a) and 2(a) were answered well with most candidates knowing that 'if' is the key-word for selection, and also able to match the given definition to validation.

- Question 1(b), where candidates had to match the error type to its description, was also very well answered. The few candidates who did not get full marks seemed to struggle with linking translation failure to syntax errors.

- Single mark questions were also generally well answered.
    - Question 1(c)(i) was well answered; however, some candidates did give the definition of a constant or an example of its use e.g. PI, rather than a reason for using one.
    - Question 1(d); few candidates lost marks as most could identify the required parts of the code accurately. The most common error was 1(d)(iii) where some candidates identified age as a real number.
    - In questions 3(a)(ii), most candidates could accurately identify the number of swaps, though some candidates confused this with the number of comparisons in the first pass. 3(a)(iii) was not as well answered as most candidates did not recognise that an additional pass with no swaps was required to complete the algorithm.
    - Question 4(a) was very well answered as almost all candidates understood the meaning of decomposition.

- Multiple mark questions had mixed outcomes, with candidates often not able to adequately link their responses in order to gain full marks.
    - Question 1(c)(ii) performed quite well as most candidates were able to explain that variables store a single value, while an array could store multiple values. Where candidates did not get both marks, it was often due to confusion about holding different data types.
    - Question 2(b) was also well answered with most candidates getting both marks for 2(b)(ii) however, a minority of candidates were not able to correctly identify a data type and lost both marks as the example had to match the given type. In 2(b)(iii) most candidates scored 2 marks

from the 3 available as they were able to accurately identify input and outputs but were too vague in identifying a process often stating something like 'getting Tax'.

- Question 2(c) was well answered, however quite a few candidates did use a different data type for erroneous even though the questions specifically asked for integer data.
- Question 3(a)(i) responses were very often in the (less common) right-to-left direction. Although the right-to-left sorts required more work by the candidate to get both marks, those who went this direction often did better than the more conventional left-to-right sorts.
  For those candidates completing right-to-left sorts, a common misconception in bubble sorts became apparent. This was seen across a large number of centres rather than isolated instances. Many candidates seemed to think that only one value moves in each pass. Candidates bubbled the '19' up to the second last position in the list. This left a single space between the 18 and 19. Rather than the 19 continuing to bubble up, candidates allowed the 18 to bubble up behind the 19 thus gaining no marks. This common misconception is demonstrated below.

9	**11**	**12**	**19**	**14**	**18**	**15**	**17**
9	11	12	14	19	18	15	17
9	11	12	14	18	19	15	17
9	11	12	14	18	15	19	17
9	11	12	14	15	18	19	17
9	11	12	14	15	18	17	19
9	11	12	14	15	17	18	19

  Another common issue with this question was candidates using each available row to show individual comparisons. This resulted in candidates running out of rows and usually only able to complete the first pass thus gaining only 1 of the 2 available marks.
- Question 3(b); most candidates recognised that a binary search was the most appropriate option, but many failed to justify the choice. Many would simply repeat the question, 'it is already sorted', rather than explaining why being sorted made a binary search more efficient i.e. divide and conquer.
- Question 4(b)(ii); typical responses only scored 1 mark from the available 2. Although most candidates knew both what abstraction means and what libraries are, it was rare that candidates produced an appropriately linked response. Many candidates explained abstraction but gave irrelevant examples such as the ignoring the size or colour of a dice. Others explained abstraction for the first mark and then gave a benefit of using a library without linking this to abstraction.
- Question 5(a); both parts of this question had typically good responses with almost all candidates obtaining some of the available marks. While many candidates could identify that score was being reset in 5(a)(i), many failed to expand their response to gain the 2nd mark by explaining the impact this will have on the program output. Some candidates gave vague or inaccurate descriptions for 5(a)(ii) with a number of candidates describing a simple number guessing game. Most candidates did recognise that this was an addition quiz and often gave detailed descriptions hitting more than the two required descriptors.

REPORT ON INDIVIDUAL CODE QUESTIONS

Q1(b)(ii) This question was generally well answered with most candidates obtaining all three marks. Almost all responses fixed the syntax errors, with the most often lost mark, although rare, from candidates not fixing the logical error to add 1 to the redCount.

Q2(c)(ii) Another well answered question with most candidates obtaining most of the available marks. When marks were lost it was occasionally where candidates used the 'or' logical operator rather than 'and' to link the two relational conditions, but the most often lost mark was candidates not being able to accurately call the getHex function giving the correct argument.

Q3(c) It was anticipated that many candidates would struggle to identify that a post-conditional while loop was being implemented in the flowchart and this proved to be the case. Very few candidates initialised variables with appropriate values to allow the loop to run at least once. Most candidates modified the code to create a pre-conditioned loop. This was not penalised in the mark-scheme but may be in future series where post-conditional loops are specifically requested. Some candidates who used a pre-conditional loop often had an input value being lost, as an age was requested before the loop was entered and then again immediately at the beginning of the loop. As with flow-chart questions in previous series, many candidates failed to identify that the first question (diamond) box was intended to be a while loop, and so created two if statements in their code, resulting in solutions where only one age could be entered. A few candidates were not able to set the average to a whole number before output, or to output multiple variables in a single output statement.

Q4(c) Candidates usually scored well in this comment-first coding question. Most candidates were able to create an appropriate loop for the three dice rolls though many were not able to use the random library accurately to generate a random value between 1 and 6. As in previous series many candidates failed to use either of the given constants and lost a simple mark. A small but significant number of candidates did not use appropriate messages for user outputs as indicated in the question paper.

Q5(b) File handling remains a challenging area for many candidates. While a significant number were able to open a file, many used the incorrect file mode—most commonly 'r' or 'a' —instead of the required 'w'. Furthermore, only a few candidates were able to write meaningful content to the file. It was also common to see the file being opened after score and grade calculations had already been attempted, rather than before initiating the relevant processing loop.

In the identification of the two highest scores from a set of values, it was expected that most candidates would iterate through the inner array while tracking and subtracting the lowest value, or alternatively, that some might sort each inner array and sum the highest two elements. Methods such as sum() and min() were also considered viable approaches. However, only a small number of candidates employed any of these strategies.

Instead, many opted to use overly complex sequences of selection statements in an attempt to identify the lowest value. This approach often led to logical errors, resulting in inaccurate final scores under certain conditions, particularly where there were two equal low values.

Q6 Although column alignment in the final output was not required to achieve full marks, it was encouraging to observe that many candidates demonstrated the use of string formatting techniques, including f-strings, to produce well-formatted output. However, the use of f-strings introduced additional complexity, leading to runtime errors for some candidates. In a few cases, candidates spent unnecessary time calculating string lengths and manually adding spaces to achieve alignment.

Most candidates successfully implemented subprograms to convert temperatures between different scales, even if they were never called. However, a common challenge was the correct use of string manipulation and type conversion to extract the temperature value and scale from the given temperature strings and then pass these to the relevant subprograms. While many candidates were able to accept a city name as input and iterate through the city table to find a match, fewer candidates successfully identified the corresponding index and then use it to update the appropriate element in the temperature array.

In stronger responses, candidates used the array's length in conjunction with count-controlled loops to update temperatures effectively. Higher-level solutions often included optimisations such as exiting the loop early upon finding a match or using the index() function to locate the city and directly update the associated temperature.

A number of candidates developed fully functional solutions but did not achieve full level-based marks due to the inclusion of unnecessary loops or redundant data structures. A common pattern involved using separate loops to create lists for Celsius and Fahrenheit values, another loop to construct the table body, and an additional loop for calculating and displaying the averages row—adding unnecessary complexity to the overall solution.

Pearson Education Limited. Registered company number 872828
with its registered office at 80 Strand, London, WC2R 0RL, United Kingdom