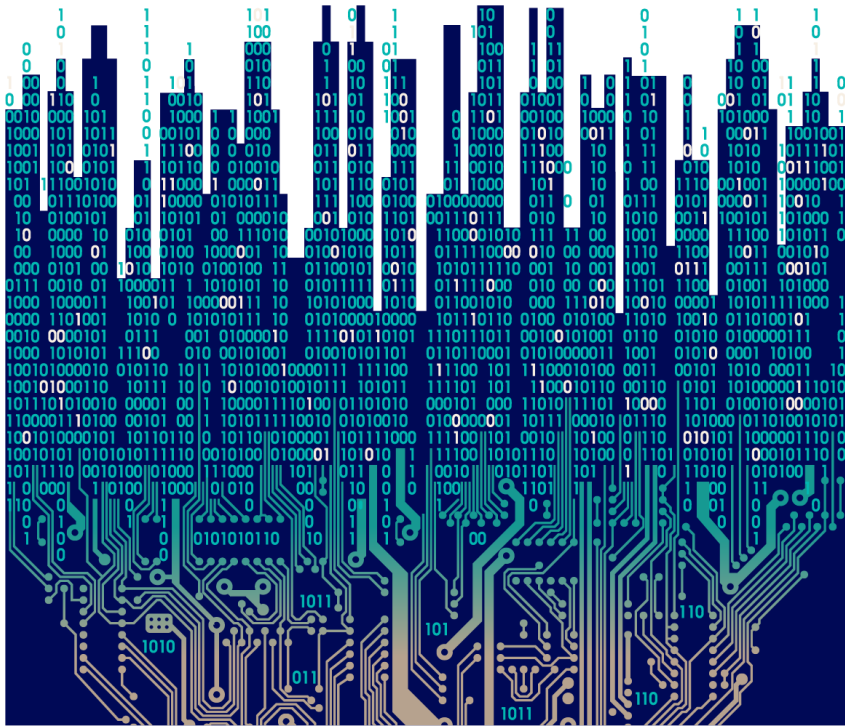
GCSE
Computer Science

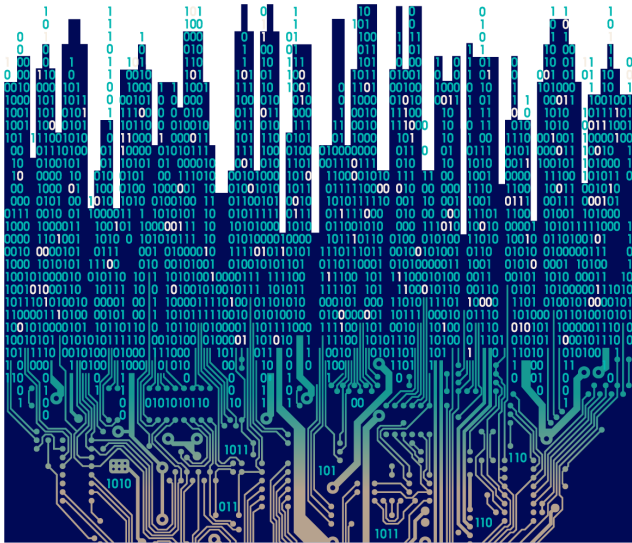
First teaching in September 2020

First assessment 2022



Pearson
Edexcel

Welcome



Pearson
Edexcel

Agenda

Time	Item
16:00	Welcome
16:05	Introduction to the specification
16:10	Paper 1 – Introduction
16:20	Paper 1 – Marking
16:50	Paper 2 – Introduction
17:00	Paper 2 – Marking (Q01–Q03)
17:20	Paper 2 – Marking (Q04–Q06)
17:50	Support
17:55	Final questions
18:00	Finish

Why are you here?

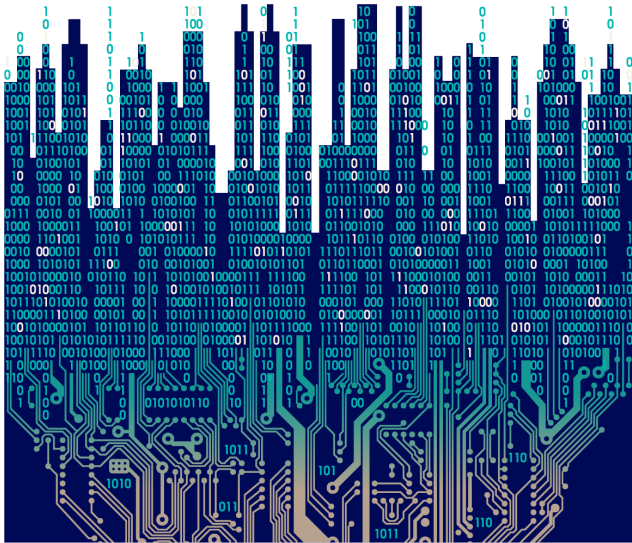
Poll.

Please click the appropriate box to say whether you are:

- an NQT
- moving from another exam board
- new to your school
- an Edexcel teacher who wants a bit of a refresher.

Please use the chat window to indicate what you hope to get out of this course.

Introduction to the specification



Pearson
Edexcel

Pearson Edexcel GCSE Computer Science

- A new course in 2020, with first assessment in 2022
- Has six topics:
  - Topic 1: Computational thinking
  - Topic 2: Data
  - Topic 3: Computers
  - Topic 4: Networks
  - Topic 5: Issues and impact
  - Topic 6: Problem solving with programming.

Assessment breakdown

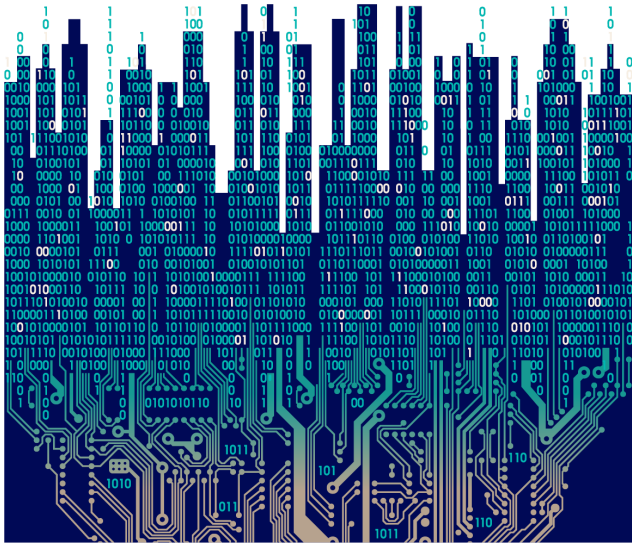
Paper 1: Principles of Computer Science

- Written examination
- 1 hour 30 minutes
- 75 marks (50% of the qualification)

Paper 2: Application of Computational Thinking

- Onscreen examination
- 2 hours
- 75 marks (50% of the qualification)

Paper 1: Principles of Computer Science



Pearson
Edexcel

Paper 1: Principles of Computer Science

- Topics 1–5, only
- One question for each topic, with subparts
- Order of topics will vary each year
- Coverage of individual specification points will vary each year.

Assessment items

- Multiple-choice and multiple-response
- Short-open
- Medium-open
- Tabular
- Diagrammatic
- Extended-open

Ramping in questions

- Paper 1 is ramped so that there is an increase in demand through the paper.
- Subparts of questions are also ramped.
- Candidates should attempt all questions as instructed on the front cover.

Command words

- To help students understand the answer our examiners want to see, we use command words in questions.
- Paper 1 has a range of command words to suit the different question types.

Amend	Define	Give, state, name
Calculate	Describe	Identify
Complete	Discuss	Write
Construct	Draw	
Convert	Explain	

Interpreting command words

Paper 1 uses two command words that have well-defined requirements.

Describe

- This command word requires that an account of something be given.
- Statements need to be developed for clarity.
- This command word is sometimes used to ask for a comparison.

Explain

- This command word requires a justification of a point.
- The parts of the response must be linked.
- Use joining words, such as because, so that, or in order to.

Extended open-response questions

Paper 1 will have two extended open-response questions.

Question 4 – final subpart

- Uses the command word Discuss
- Requires demonstration of a sustained line of reasoning
- Requires a written textual response
- There are no marks for spelling, punctuation, and grammar.

Question 5 – final subpart

- May use any command word
- Must not be essay-based
- May be drawing, labelling, tabular, or diagrammatic.

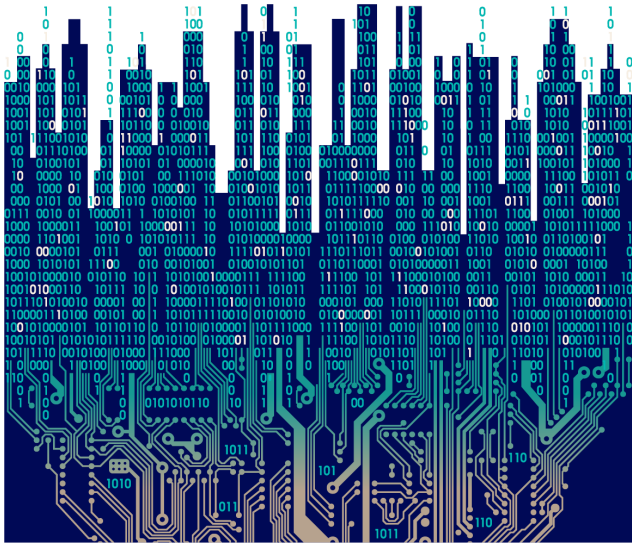
Levels-based mark scheme (LBMS)

Paper 1 uses an LBMS for the final item in Question 4. All other questions are points-based.

Level	Mark	Descriptor
	0	No rewardable content.
Level 1	1–2	Basic, independent points are made, showing elements of understanding of key concepts/principles of computer science. (AO1) The discussion will contain basic information with little linkage between points made or application to the context. (AO2)
Level 2	3–4	Demonstrates adequate understanding of key concepts/principles of computer science. (AO1) The discussion shows some linkages and lines of reasoning with some structure and application to the context. (AO2)
Level 3	5–6	Demonstrates comprehensive understanding of key concepts/principles of computer science to support the discussion being presented. (AO1) The discussion is well developed, with sustained lines of reasoning that are coherent and logically structured, and which clearly apply to the context. (AO2)

Paper 1: Principles of Computer Science

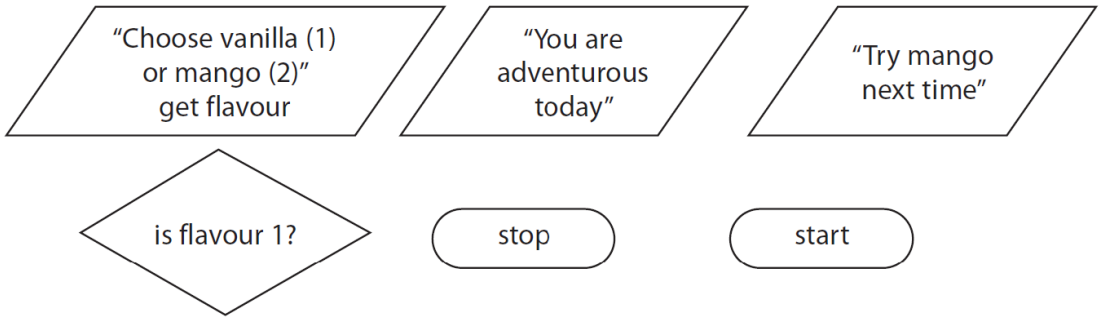
Marked examples



Pearson
Edexcel

Q01d

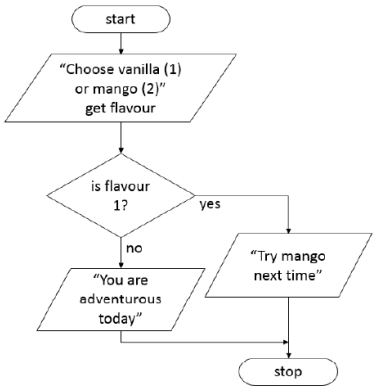
(d) Here is a set of flowchart symbols for an algorithm about flavours.



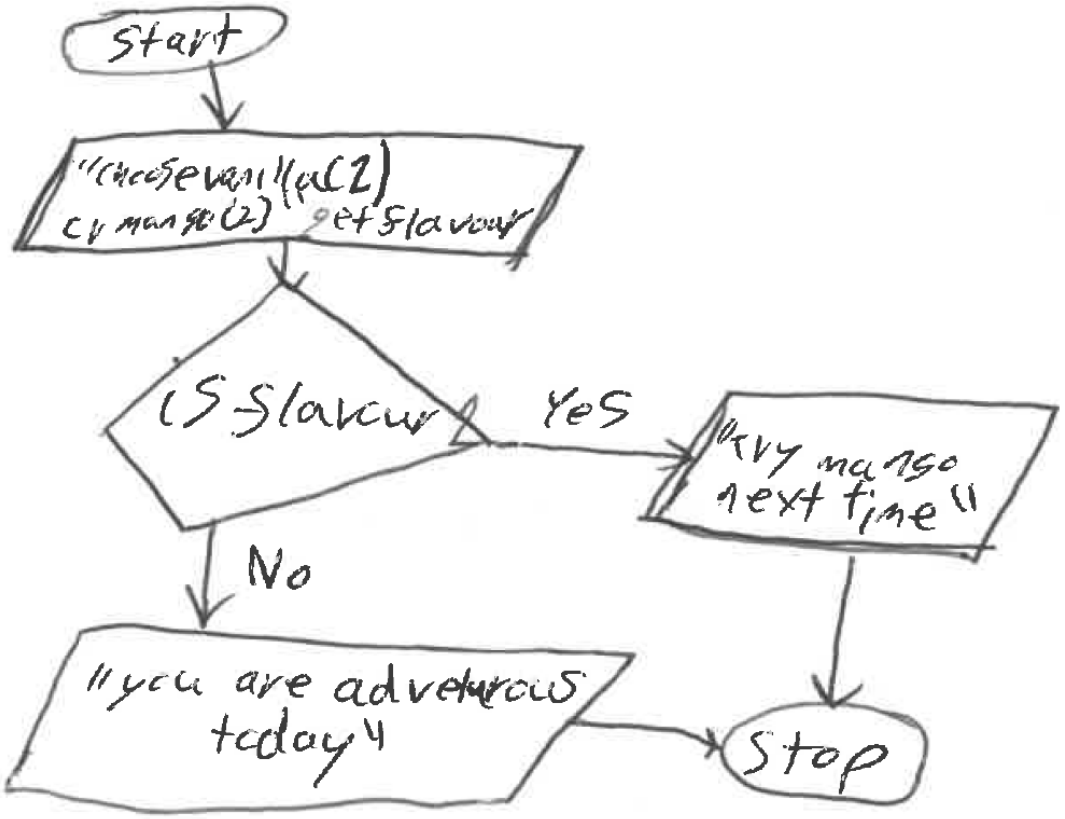
Draw a flowchart to show the completed algorithm.

Use every symbol exactly once. Use as many arrows and labels as you require.
Do **not** add any additional symbols.

(4)

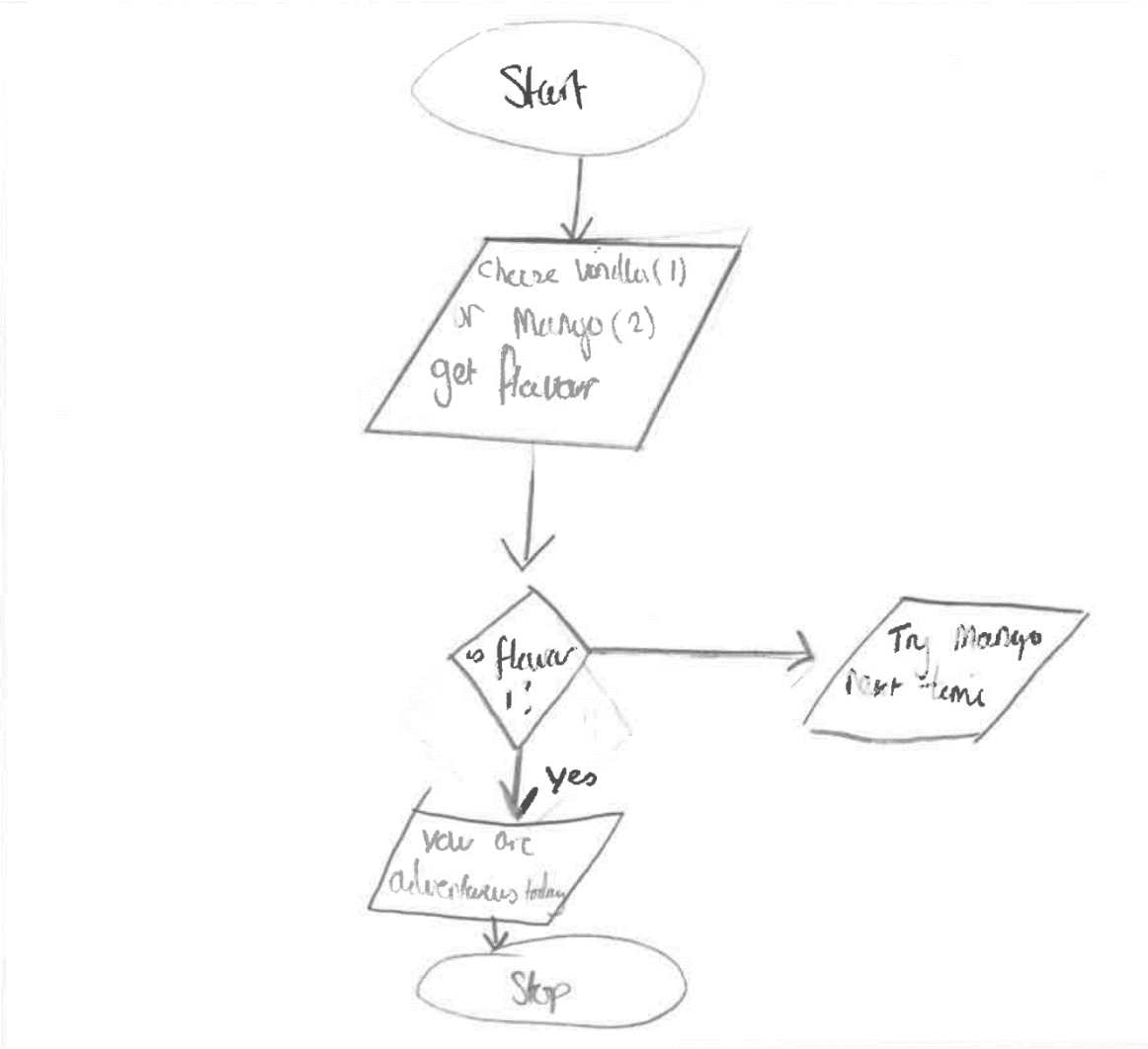
Question number	Answer	Additional guidance	Mark
1(d)	• Start and stop terminators in correct positions (1) • Decision symbol has two outputs only, i.e. the two output messages (1) • Yes/no labels on decision match output messages (1) • Fully connected to function correctly (1) 		**4**

Q01d – Response A



3/4

Q01d – Response B



2/4

Q03g

(g) Describe **two** ways that a high-level language differs from a low-level language.

(4)

1 .....

.....

.....

.....

2 .....

.....

.....

.....

Question number	Answer	Additional guidance	Mark
3(g)	Any **two** from: • High-level languages use instructions that look like English (1), whereas a low-level language uses mnemonics/binary code (1). • A high-level language statement generates many lines of machine code (1), whereas each line of low-level languages is/generates a single machine instruction (1) • High-level languages are general purpose/exist across microprocessors/CPUs/machine-independent (1), whereas a low-level language is microprocessor/CPU/machine specific (1) • High-level languages are abstracted from the hardware (1), whereas low-level languages manipulate the hardware directly (1)	• Responses should be about the languages, not about the output of the language translators (efficiency) or other software based on the language (tools)	**4**

Q03g – Response A

1 High level languages are programmer friendly whereas low level languages are difficult and time consuming to work out.

2 High level languages will run on any computer device with different CPUs but low level languages are not Machine Specific so Change depending on the Computer.

2/4

Q03g – Response B

(g) Describe **two** ways that a high-level language differs from a low-level language.

(4)

1 High-level languages are closer to human languages and easier to understand whereas low-level languages ~~and~~ are difficult and require knowledge of it to understand.

2 ~~Hig~~ High-level languages have tools to debug the code whereas low-level languages do not.

1/4

Q04d

(d) Discuss methods a movie streaming company could use to secure its network.

Your answer should consider:

- physical security
- access control
- firewalls.

(6)

Question Number	Indicative content
4(d)	Discuss methods of securing networks. **Physical security** • The movie streaming company should use physical security to protect its products and data from unauthorised access. • Physical security can be used to only allow authorised people to enter critical areas, such as the room with the file servers where the movies are stored. • The machine storing the movies and the movie streaming company's data should be in locked rooms/cupboards. • Each door could be fitted with an electronic lock system which can record who enters the server room and when. • Electronic lock systems can read cards/fobs/biometrics of employees and check them against details stored on a database to deter **Access control** • The movie streaming c • Each time someone att checked against a data • The credentials also de movie files. Different employees/n

Level	Mark	Descriptor
	0	No rewardable content.
Level 1	1-2	Basic, independent points are made, showing elements of understanding of key concepts/principles of computer science. (AO1) The discussion will contain basic information with little linkage between points made or application to the context. (AO2)
Level 2	3-4	Demonstrates adequate understanding of key concepts/principles of computer science. (AO1) The discussion shows some linkages and lines of reasoning with some structure and application to the context. (AO2)
Level 3	5-6	Demonstrates comprehensive understanding of key concepts/principles of computer science to support the discussion being presented. (AO1) The discussion is well developed, with sustained lines of reasoning that are coherent and logically structured, and which clearly apply to the context. (AO2)

Q04d – Response A

(d)

Access doors ~~should be installed~~ Access doors should be in place, in order to prevent unauthorised access to server rooms (could be fitted with facial recognition software).

Software should only be made accessible to authorised personal, by installing a multifactor authentication. Access control can help decide who logs onto the network of the movie streaming company, and limit who has access to their files and who has permission to alter them.

A firewall could also be installed, in order to prevent communications from entering the ~~network~~ network without permission and also to prevent programs and users from accessing the internet from within the network without permission.

4/6

Q04d – Response B

A way a streaming company could protect its network is first by using physical security. They can do this by using keycards to access the server room, locks, one way entrances etc. This is important as it makes sure no one can destroy or steal their networks.

Access control also does this as it stops certain people from the work place entering the server room. This allows for specific people to access these rooms e.g tech support, engineers etc.

Firewalls will for digital protection as it filters anything from outside the internal network to see if it has access.

2/6

Q05a

(a) Describe **one** reason that lossless compression is preferred to lossy compression for transmitting word-processed documents over a network.

(2)

Question number	Answer	Additional guidance	Mark
5(a)	A lossless compression will allow the exact contents of the word-processed document to be reconstructed (1) whereas a lossy compression will permanently remove data from the file (1)		**2**

Q05a – Response A

lossy compression removes unwanted data from the file whereas lossless doesn't meaning no words or letters will be removed from the document.

2/2

Q05a – Response B

5 Data

(a) Describe **one** reason that lossless compression is preferred to lossy compression for transmitting word-processed documents over a network.

(2)

Lossless compression reduces the filesize without any data being deleted whereas lossy compression deletes data when reducing the file size.

1/2

Q05c

(c) Explain the reason why at least nine bits are needed to store 300 different binary patterns.

(2)

Question number	Answer	Additional guidance	Mark
5(c)	One from • 2^9 gives 512 (1) patterns whereas 2^8 gives 256 patterns (1) • 2^8 (1) does not give enough patterns whereas 2^9 (1) gives more than enough patterns	• Do not award responses stating 8 bits is not big enough, as it is implied in the stem.	2

Q05c – Response A

(c) Explain the reason why at least nine bits are needed to store 300 different binary patterns.

(2)

number of patterns stored = 2^{bits}

$2^9 = 512$

$2^8 = 256$ 8 bits is too small

2/2

Q05c – Response B

(c) Explain the reason why at least nine bits are needed to store 300 different binary patterns.

(2)

because the number of bits is the power from two e.g. $2^8 = 256$ so $2^9 = 300$.

1/2

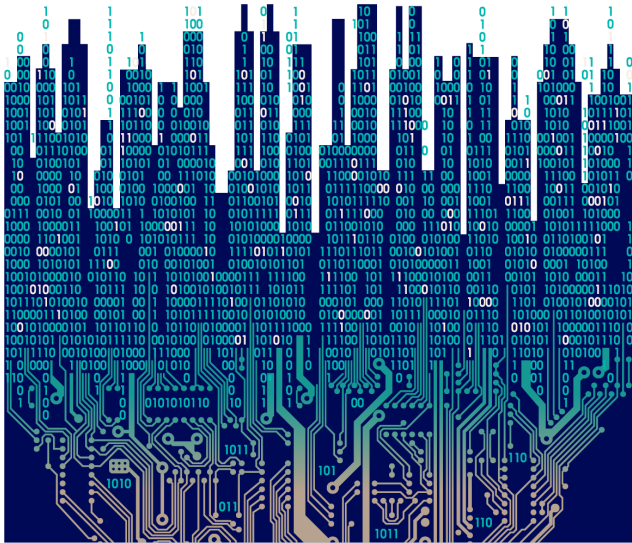
Command words?

Poll.

Please click the appropriate box to say whether you:

- understand the requirements of the explain and describe command words
- want more practice with the command words for Paper 1.

Paper 2: Application of Computational Thinking



Pearson
Edexcel

Paper 2: Application of Computational Thinking

- Topic 6 only
- Order of specification points will vary each year
  - Data files will be provided if required by the specification point assessed
- Coverage of individual specification points will vary each year.

Assessment items – Paper 2

- All questions require interaction with the Python 3 programming language and a development environment on a computer.
- Each question is based on an initial code file, which is provided.
- Students are asked to amend the initial code file to:
  - Identify parts of or components in the code
  - Find and fix errors in the code
  - Select between optional lines of code
  - Arrange lines of code to produce a functional program
  - Complete lines of code
  - Add lines of code

Ramping in questions

- Each paper is ramped so that there is an increase in demand through the paper.
- Candidates should attempt all questions as instructed on the front cover.

Command words

- To help students understand the answer our examiners want to see, we use command words in questions.
- Paper 1 has a range of command words to suit the different question types.
- Paper 2 has two command words only.
- An Appendix in the specification sets out the expectation of each command word.

Levels-based mark scheme

Paper 2 uses points-based mark schemes for some questions and a hybrid mark scheme for others. A combination of up to three LBMS may be used.

One LBMS is for solution design.

Solution design (levels-based mark scheme)

0	1	2	3	Max.
No rewardable material	• There has been little attempt to decompose the problem. • Some of the component parts of the problem can be seen in the solution, although this will not be complete. • Some parts of the logic are clear and appropriate to the problem. • The use of variables and data structures, appropriate to the problem, is limited. • The choice of programming constructs, appropriate to the problem, is limited.	• There has been some attempt to decompose the problem. • Most of the component parts of the problem can be seen in the solution. • Most parts of the logic are clear and appropriate to the problem. • The use of variables and data structures is mostly appropriate. • The choice of programming constructs is mostly appropriate to the problem.	• The problem has been decomposed clearly into component parts. • The component parts of the problem can be seen clearly in the solution. • The logic is clear and appropriate to the problem. • The choice of variables and data structures is appropriate to the problem. • The choice of programming constructs is accurate and appropriate to the problem.	3

Levels-based mark scheme

The second LBMS is for good programming practices.

Good programming practices (levels-based mark scheme)

0	1	2	3	Max.
No rewardable material	• There has been little attempt to lay out the code into identifiable sections to aid readability. • Some use of meaningful variable names. • Limited or excessive commenting. • Parts of the code are clear, with limited use of appropriate spacing and indentation.	• There has been some attempt to lay out the code to aid readability, although sections may still be mixed. • Uses mostly meaningful variable names. • Some use of appropriate commenting, although may be excessive. • Code is mostly clear, with some use of appropriate white space to aid readability.	• Layout of code is effective in separating sections, e.g. putting all variables together, putting all subprograms together as appropriate. • Meaningful variable names and subprogram interfaces are used where appropriate. • Effective commenting is used to explain logic of code blocks. • Code is clear, with good use of white space to aid readability.	3

Levels-based mark scheme

The third LBMS is for functionality.

Functionality (levels-based mark scheme)

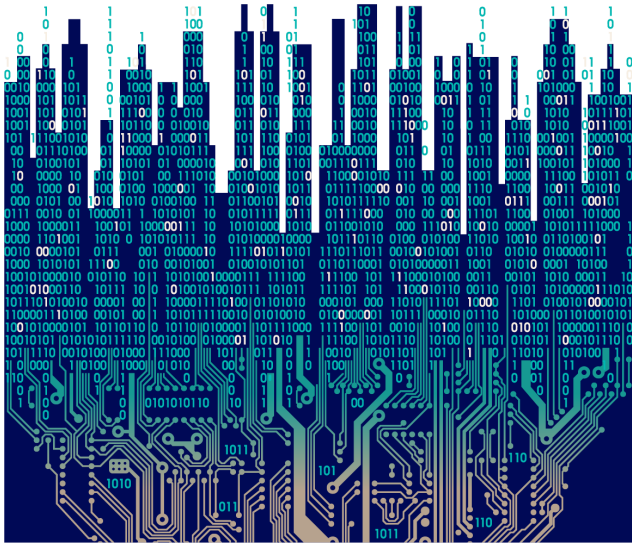
0	1	2	3	Max.
No rewardable material	**Functionality (when the code is run)** • The component parts of the program are incorrect or incomplete, providing a program of limited functionality that meets some of the given requirements. • Program outputs are of limited accuracy and/or provide limited information. • Program responds predictably to some of the anticipated input. • Solution is not robust and may crash on anticipated or provided input.	**Functionality (when the code is run)** • The component parts of the program are complete, providing a functional program that meets most of the stated requirements. • Program outputs are mostly accurate and informative. • Program responds predictably to most of the anticipated input. • Solution may not be robust within the constraints of the problem.	**Functionality (when the code is run)** • The component parts of the program are complete, providing a functional program that fully meets the given requirements. • Program outputs are accurate, informative, and suitable for the user. • Program responds predictably to anticipated input. • Solution is robust within the constraints of the problem.	3

Using mark schemes

- Some questions will be marked using points-based mark schemes.
- Some questions will be marked using a hybrid mark scheme, made up from both points and one or more of the levels-based mark schemes.
- Every trait in a level does not have to be ticked for a response to be awarded that level.
- The level awarded in the different levels-based mark schemes do not have to match.

Paper 2: Application of Computational Thinking

Marked examples



Pearson
Edexcel

Q01

1 A program must calculate the circumference of a circle. The user enters the radius of the circle. A radius of zero or less is invalid.

The formula to calculate the circumference of a circle is:

$$circumference = 2\pi r$$

- π is the constant Pi
- r is the radius.

Open file **Q01.py**

Amend the code to add or complete lines to:

- import the math library
- create two variables
- take input from the user
- check for an invalid input of zero or less
- display a message to tell the user the input is invalid
- calculate the circumference
- round the circumference to three decimal places using the round() function.

Do **not** add any additional functionality.

Save your amended code file as **Q01FINISHED.py**

(Total for Question 1 = 10 marks)

```
17 # Main program
18 # ----------------------------
19 # =====> Complete the line to a
20 #        the user, to the varia
21 radius =
22
23 # =====> Complete the code in t
24 #        an invalid radius of z
25 if (radius           ):
26     # =====> Add a line to tell
27
28 else:
29     # =====> Complete the calcu
30     circumference =
31
32     # =====> Complete the line
33     #        decimal places usi
34     circumference =
35
```

Answer

Award marks as shown.

- Add a line to import the math library (1)
  Original: <Blank>
  Amended: `import math`
- Create and set an integer variable (1)
  Original: <Blank>
  Amended: `radius = 0`
- Create and set a real variable (1)
  Original: <Blank>
  Amended: `circumference = 0.0`
- Complete the line to take input from the user
  Original: `radius =`
  Amended: `radius = int (input ("Enter the radius of a`
  Call to input(), with a prompt (1)
  Call to int() to convert string to integer (1)
- Complete the line to validate for negative radii (1)
  Original: `radius`
  Amended: `radius <= 0`
- Add a line to print an invalid input message (1)
  Original: <Blank>
  Amended: `print ("Invalid radius")`
- Complete the calculation of the circumference (1)
  Original: `circumference =`
  Amended: `circumference = 2 * math.pi * radius`
- Complete the line to round circumference to three decimal pl
  Original: `circumference =`
  Amended: `circumference = round (circumference, 3)`
  Call to round() (1)
  Correct parameters to round() (1)

Q01 – Response

```
# ---------------------------------------------------------------------
# Import libraries
# ---------------------------------------------------------------------
# =====> Import the math library
import math
# ---------------------------------------------------------------------
# Global variables
# ---------------------------------------------------------------------
# =====> Create an integer variable named radius and set it to 0
radius= 0

# =====> Create a real variable named circumference and set it to 0.0
circumference= 0.0

# ---------------------------------------------------------------------
# Main program
# ---------------------------------------------------------------------
# =====> Complete the line to assign an integer, input by
#        the user, to the variable radius
radius = input("what would you like the radius to be?")

# =====> Complete the code in the brackets to check for
#        an invalid radius of zero or less input by the user
if (radius < circumference):
    print ("invalid entry")
    # =====> Add a line to tell the user the entry is invalid

else: print ("Calculating logical operation")
    # =====> Complete the calculation of the circumference
circumference = (math.pi * (radius*2))##0

    # =====> Complete the line to round circumference to three
    #        decimal places using the round() function
    circumference =

    print ("The circumference is", circumference)
```

6/10

Q02

2 A programmer has started to write a program, but it does not work correctly.

The program should ask users to enter their initials.

The program should report an error when the user enters fewer than two initials or more than three initials or when the characters are not alphabetic.

When the input is valid, the initials should be reported back to the user in uppercase.

The program should allow the user to go again as long as the user enters a 'Y' or 'y' when asked.

Open file **Q02.py**

Amend the code to:

- fix the syntax error on original line 5
  `initials =`
- fix the syntax error on original line 17
  `else if (len (initials) > 3):`
- fix the syntax error on original line 19
  `else`
- change the Boolean operator to fix the logic error on original line 11
  `while ((a == "Y") and (a == "y")):`
- add a Boolean operator to fix the logic error on original line 13
  `if (initials.isalpha ()):`
- change the relational operator to fix the logic error on original line 15
  `elif (len (initials) <= 2):`
- amend the assignment to fix the logic error on original line 23
  `a == input ("Would you like to go again? ")`
- amend original line 20 to convert to uppercase
  `initials = initials.lower ()`
- change the variable a to a more meaningful name
- add a comment to original line 21, to explain what it does.

Do **not** add any additional functionality.

Save your amended code file as **Q02FINISHED.py**

(Total for Question 2 = 10 marks)

```
3  # ----------------------------
4  a = "Y"
5  initials =
6
7  # ----------------------------
8  # Main program
9  # ----------------------------
10
11 while ((a == "Y") and (a == "y"
12     initials = input ("Enter yo
13     if (initials.isalpha ()):
14         print ("Must be alphabe
15     elif (len (initials) <= 2):
16         print ("Not long enough
17     else if (len (initials) > 3
18         print ("Too long")
19     else
20         initials = initials.low
21         print ("Your initials a
```

Answer

Award marks as shown.

- (line 5) (1)
  From: `initials =`
  To: `initials = ""` (any string value)
- (line 17) (1)
  From: `else if (len (initials) > 3):`
  To: `elif (len (initials) > 3):`
- (line 19) (1)
  From: `else`
  To: `else:`
- (line 11) (1)
  From: `while ((a == "Y") and (a == "y"`
  To: `while ((a == "Y") or (a == "y")):`
- (line 15) (1)
  From: `elif (len (initials) <= 2):`
  To: `elif (len (initials) < 2):`
- (line 13) (1)
  From: `if (initials.isalpha ()):`
  To: `if (not initials.isalpha ()):`

Q02 – Response

```
1  # ---------------------------------------------------------------------
2  # Global variables
3  # ---------------------------------------------------------------------
4  a = "Y"
5  nickname = 0
6
7  # ---------------------------------------------------------------------
8  # Main program
9  # ---------------------------------------------------------------------
10
11 while ((a == "Y") or (a == "y")):
12     nickname = input("Enter your initials, without spaces: ")
13     if nickname.isalpha ():
14         print("Must be alphabetic characters")
15     elif (len(nickname) <= 2):
16         print("Not long enough")
17     if (len(nickname) > 3):
18         print("Too long")
19     else:
20         nickname = nickname.upper()
21         print("Your nickname is:", nickname)  # this prints the nickname
22
23     a = input("Would you like to go again? ")
```

6/10

Q03

3 A program is needed to print a horizontal histogram showing the frequency of the vowels (A, E, I, O, U) in a string. The user enters the string.

The string will fit on one line only and will not include a carriage return.

For the string 'The quick brown fox jumps over the lazy dog', this histogram shows that 'A' occurs only once and 'E' occurs exactly three times.

The output must be formatted like the histogram. The output will vary according to the user's input.

```
A | A
E | EEE
I | I
O | OOOO
U | UU
```

Open file **Q03.py**

Amend the code to:

- create two one-dimensional data structures, implemented as lists
- complete calls to built-in subprograms
- complete a call to a user-devised subprogram
- use selection to check for each vowel and increment the corresponding count
- ensure the program behaves predictably if the user enters an empty string or a string without any vowels, by displaying the vertical axis only.

Do **not** add any additional functionality.

Save your amended code file as **Q03FINISHED.py**

(Total for Question 3 = 13 marks)

```
8  theVowels =
9
10 # =====> Create a one-dimensional data structure to hold th
11 theCounts =
12
13 # ------------------------------------------------------------
14 # Subprograms
15 # ------------------------------------------------------------
16 def displayHistogram (inSymbols, inNumbers):
17     # =====> Complete the call to range (), using len ()
18     for index in range (                    ):
19         # Repeat the symbol for the number of times require
20         # =====> Complete the print statement to print the
21         #        for the number of times it was counted
22         print (                             )
23
24 # ------------------------------------------------------------
25 # Main program
26 # ------------------------------------------------------------
27 # User types in a string
28 data = input ("Enter a string: ")
29
30 # =====> Complete the line to convert data to uppercase
31 data =
32
33 # Count each vowel in the input
34 for letter in data:
35     # =====> Use selection to check for each vowel and incr
```

Answer

Award marks as shown.

- Complete theVowels with a list of strings (1)
  `theVowels = ["A", "E", "I", "O", "U"]`
- Complete theCounts with a list of integers (1)
  `theCounts = [0, 0, 0, 0, 0]`
- Call to range() must use 'len (inSymbols)' (1)
  `for index in range (len (inSymbols)):`
- First item in print statement should be the vowel/symbol using indexi
  `inSymbols[index]`
- Histogram line should be the vowel/symbol repeated using string mult
  (symbol*count) using indexing (1)
  `inSymbols[index] * inNumbers[index]`
- Contents of data converted using <string>.upper() (1)
  `data = data.upper()`

1	2
Functionality (when the code is run) • The component parts of the program are incorrect or incomplete, providing a program of limited functionality that meets some of the given requirements. • Program outputs are of limited accuracy and/or provide limited information. • Program responds predictably to some of the anticipated input. • Solution is not robust and may crash on anticipated or provided input.	**Functionality (when the code is run)** • The component parts of the program are complete, providing a functional program that meets most of the stated requirements. • Program outputs are mostly accurate and informative. • Program responds predictably to most of the anticipated input. • Solution may not be robust within the constraints of the problem.

Q03 – Response

```
# ---------------------------------------------------------------------
# Global variables
# ---------------------------------------------------------------------
data = ""

# =====> Create a one-dimensional data structure holding the
#        five vowels in upper case

theVowels = ["A", "E", "I", "O", "U"]

# =====> Create a one-dimensional data structu

theCounts = [0,0,0,0,0]

# -----------------------------------------
# Subprograms
# -----------------------------------------

def displayHistogram (inSymbols, inNumbers):

    # =====> Complete the call to range (), us

    for index in range (len(inSymbols[0])):

        # Repeat the symbol for the number of
        # =====> Complete the print statement
        #        for the number of times it wa

        print (inNumbers, inSymbols)

```

```
# ---------------------------------------------------------------------
# Main program
# ---------------------------------------------------------------------
# User types in a string

data = input ("Enter a string: ")

# =====> Complete the line to convert data to uppercase

data = data.upper()

# Count each vowel in the input
for letter in data:
    if letter == theVowels[0]:#if the letter A is found it adds 1 to A
        theCounts[0] + 1

    elif letter == theVowels[0]:#if the letter E is found it adds 1 to E
        theCounts[1] + 1

    elif letter == theVowels[0]:#if the letter I is found it adds 1 to I
        theCounts[2] + 1

    elif letter == theVowels[0]:#if the letter O is found it adds 1 to O
        theCounts[3] + 1

    elif letter == theVowels[0]:#if the letter U is found it adds 1 to U
        theCounts[4] + 1

    else:
        print("Not a vowel")
    # =====> Use selection to check for each vowel and increment th                nt


# Print a horizontal histogram
# =====> Complete the call to the user-devised subprogram

displayHistogram (theVowels,theCounts)

print (data)
```

9/13

Q04

4 An International Standard Book Number (ISBN) is a unique identifier for commercially published books. Here is an example of an ISBN barcode.

© PAL

A program has been written to calculate the check digit for a 10-digit ISBN.

The user enters the first nine digits of the ISBN. A check digit is calculated and appended to the original digits. The numeric value of the check digit is calculated by:

- multiply ten times the first digit from the left, nine times the second digit from the left, and onwards until two times the furthest digit from the left
- add all the products together
- apply a modulus 11 to the sum of all the products
- calculate the numeric value of the check digit by subtracting the result of the modulus operation from 11
- when the numeric value of the check digit is 11, the string '0' is appended to the ISBN
- when the numeric value of the check digit is 10, the string 'X' is appended to the ISBN
- in all other cases, the string value of the check digit is appended to the ISBN.

This table shows inputs and required outputs for a fully functional program.

No validation of inputs is required.

Input	Output
071954400	Check digit = 9 ISBN = 0719544009
061826941	Check digit = X ISBN = 061826941X
047119047	Check digit = 0 ISBN = 0471190470

Open file **Q04.py**

Amend the code to mak
need to rearrange the lir

Use comments, whi
read and understan

The lines of code in the program are mixed up. The comments are in the right order to match the logic of the solution.

Do **not** change the functionality of the given lines of code.

Do **not** add any additional functionality.

Save your amended code file as **Q04FINISHED.py**

(Total for Question 4 = 15 marks)

```
13
14 elif (checkDigit == 10):
15 isbn = input ("Enter a 9-digit ISBN nur
16 if (checkDigit == 11):
17 checkDigit = 11 - modulus
18 modulus = total % 11
19 print ("Check digit = " + strCheckDigit
20 while (index < len (isbn)):
21 else:
22
23     total = product + total
24     strCheckDigit = "0"
25     product = int (isbn[index]) * multi
26     strCheckDigit = "X"
27     strCheckDigit = str (checkDigit)
28     multiplier = multiplier - 1
29     index = index + 1
30
31 product = 0
32 checkDigit = 0
33 strCheckDigit = ""
```

Answer

Award marks as shown.

- White space used to aid readability (1)
- User input accepted as first operation, after initialisat
- Modulus calculated before checkDigit calculation (1)
- Repetition (while index < len (isbn)) around product
- product calculated before incrementing index (1)
- product calculated before decrementing multiplier (1)
- total set after product calculated (1)
- Initialisation of variables before calculations:
  - multiplier (1)
  - index (1)
  - total (1)
- Correct outputs for each set of test data:
  - 071954400 – 9 (1)
  - 047119047 - 0 (1)
  - 061826941 – X (1)
- One or more comments match blocks of code (1)
- Printing of result is the last line in the program (1)

Q04 – Response

```
# ------------------------------------------------------------------
# Global variables
# ------------------------------------------------------------------
isbn = ""
index = 0
product = 0
total = 0
multiplier = 10
modulus = 0
checkDigit = 0
strCheckDigit = ""

# ------------------------------------------------------------------
# Main program
# ------------------------------------------------------------------
# Get the user's input
Box_ID = int(input("Enter a 6-digit Box ID number: "))

# Multiply each digit by its position in the number
while (index < len (Box_ID)):
    product = Box_ID[index] * multiplier
    total = product + total
    index = index + 1
    multiplier = multiplier - 1

# Find the remainder from integer division by 11
modulus = total % 11

# Calculate the check digit
checkDigit = 11 - modulus

# Test for boundaries
if (checkDigit == 11):
    strCheckDigit = "0"
elif (checkDigit == 10):
    strCheckDigit = "X"
else:
    strCheckDigit = str (checkDigit)

print ("Check digit = " + strCheckDigit + " ISBN = " + isbn + strCheckDigit)
```

11/15

Q05

5 A set of data is stored in a comma-separated value text file. Each line in the file is made up of ten integers. All numbers are in the range of zero to 100, inclusive. The file has multiple lines.

A program is required to calculate and report the mean of each line.

A program and subprograms have been started to carry out the processing.

This is what the output must look like. Column widths may vary, but alignment and number formats should match the table.

Row	Mean
1	33.90
2	55.30
3	41.50
4	48.10
5	54.50

Open file **Q05.py**

Amend the program and subprograms to meet the following requirements:

- all data in Q05_Data.txt must be processed
- only local variables must be created, i.e. do not create any global variables
- results must be displayed in columnar format, using <string>.format().

Do **not** add any additional functionality.

Save your amended code as **Q05FINISHED.py**

(Total for Question 5 = 12 marks)

```
9  # ----------------------------------------------
10 def processLines (inFile):
11     # =====> Write your code here
12
13
14     # Get the line of numbers from the file
15     # =====> Write your code here
16
17
18         # Calculate the mean for the items and ac
19         # =====> Write your code here
20
21
```

Answer	Addi
Award marks as shown. • Calculate the mean of each row (1) • Strip carriage return (1) • Split line by comma (1) • File opened for reading only (1) • File closed before exiting program (1) • No global variables created (1)	Cons sche • [e • [a (• [fi 5 • [

Solution design (levels-based mark scheme)

0	1	2
No rewardable material	• There has been little attempt to decompose the problem. • Some of the component parts of the problem can be seen in the solution, although this will not be complete. • Some parts of the logic are clear and appropriate to the problem. • The use of variables and data structures, appropriate to the problem, is limited.	• There has been some attempt decompose the problem. • Most of the component parts problem can be seen in the solution. • Most parts of the logic are cle and appropriate to the proble • The use of variables and data structures is mostly appropria • The choice of programming constructs is mostly appropria

Functionality (levels-based mark scheme)

0	1	2
No rewardable material	**Functionality (when the code is run)** • The component parts of the program are incorrect or incomplete, providing a program of limited functionality that meets some of the given requirements. • Program outputs are of limited accuracy and/or provide limited	**Functionality (when the code is run)** • The component parts of the program are complete, providi functional program that meets most of the stated requiremer • Program outputs are mostly accurate and informative. • Program responds predictably

Q05 – Response

```
# ---------------------------------------------------------------------
# Constants
# ---------------------------------------------------------------------
FILE_NAME = "Q05_Data.txt"       # The input data file
NUMS_PER_LINE = 10               # Number of items per line in the fil

# ---------------------------------------------------------------------
# Subprograms
# ---------------------------------------------------------------------
def processLines (inFile):
    # =====> Write your code here
    open_file = open("Q05_Data.txt","r")
    meancalc=0
    meancalc2=0
    row=1
    layout = "{:<5},{:>6}"#sets layout for table
    # Get the line of numbers from the file
    # =====> Write your code here
    for line in open_file:

        for index in line:
            if index != "," and index != "\n":
                index=int(index)

                # Calculate the mean for the items and adjust the row
                # =====> Write your code here
                meancalc=index+meancalc#adds everything together for
                # Display the information in columnar format
                # =====> Write your code here
                index=index+1
                index=str(index)
            else:
                i=1+1
        meancalc2=meancalc/10
        print(layout.format(row,meancalc2)) #pri
        row=row+1
    open_file.close()


    # =====> Write your code here
    # Close the file


def displayTableHeaders ():
    # =====> Write your code here
    layout = "{:<5},{:>6}"#sets format
    print(layout.format("Row","Mean"))
    print("_ "*10)

# ---------------------------------------
# Main program
# ---------------------------------------
# Do the processing and the display
displayTableHeaders ()
processLines (FILE_NAME)
```

8/12

Q06

```
                ["Henri", "Matisse", 1869],
                ["Frida", "Kahlo", 1907],
                ["Georgia", "O'Keeffe", 188
                ["Kara", "Walker", 1969],
                ["Yayoi", "Kusama", 1929]]

theLabels = []    # Put the new user label
# ====> Write your code here
```

6 A program is needed to produce codes for labels that identify the artist of works in an exhibition.

Records for each artist are stored in a two-dimensional data structure, implemented as a list. The fields for each record are first name, last name, and year of birth.

The code for each artist is constructed by joining the first letter of the last name, the first letter of the first name, and the year of birth.

For example, a coded label for ("Andy", "Warhol", 1928) is 'WA1928'.

Open file **Q06.py**

Write a program to meet the following requirements:

- create a code for each artist in 'theArtists'
- store all codes in the data structure named 'theLabels'
- display the labels for all the artists, one label per line
- find and display the name and year of birth of the youngest artist
- ensure the code works for any number of artists.

Do **not** add any additional functionality.

Use comments, white space and layout to make the program easier to read and understand.

Save your amended code as **Q06FINISHED.py**

(Total for Question 6 = 15 marks)

Question number	Answer	Additional guidanc
	Award marks as shown. Points-based mark scheme: • Use of two-dimensional indexing to get single letters for constructing code (1) • Conversion of date (integer) to string for constructing code (1) • Use of string concatenation to construct code (1) • Appending the new label to the data structure (1) • Initial value of date set to appropriate value (0 or negative) for use with relational operator (1) • A method for tracking the youngest artist (index, whole record) (1)	Considerations for scheme: • [6.1.1] Use dec problem and cr • [6.2.2] Use of Y data structure, • [6.3.1] Convers required by pro • [6.1.2] Write in • [6.1.4] Main pr clear sections; show different solution/functio meaningful; co

Solution design (levels-based mark scheme)

0	1	2
	• There has been little attempt to decompose the problem.	• There has been some attempt decompose the problem.

Good programming practices (levels-based mark scheme)

0	1	2
material	• There has been little attempt to lay out the code into identifiable sections to aid readability. • Some use of meaningful variable names.	• There has been some attempt to lay out the code to aid readability, although sections may still be mixed. • Uses mostly meaningful variable

Functionality (levels-based mark scheme)

0	1	2
rdable material	**Functionality (when the code is run)** • The component parts of the program are incorrect or incomplete, providing a program of limited functionality that meets some of the given requirements. • Program outputs are of limited accuracy and/or provide limited	**Functionality (when the code is run)** • The component parts of the program are complete, providi functional program that meets most of the stated requiremer • Program outputs are mostly accurate and informative. • Program responds predictably

Q06 – Response

```
# ------------------------------------------------------------------
# Global variables
# ------------------------------------------------------------------
theArtists = [["Andy", "Warhol", 1928],
              ["Pablo", "Picasso", 1881],
              ["Salvador", "Dali", 1904],
              ["Lavinia", "Fontana", 1552],
              ["Jackson", "Pollock", 1912],
              ["Henri", "Matisse", 1869],
              ["Frida", "Kahlo", 1907],
              ["Georgia", "O'Keeffe", 1887],
              ["Kara", "Walker", 1969],
              ["Yayoi", "Kusama", 1929]]

theLabels = []   # Put the new user labels into this structure
# ====> Write your code here

youngYear = 0

# ------------------------------------------------------------------
# Main program
# ------------------------------------------------------------------
# ====> Write your code here

for artist in theArtists:      # Iterates each artist's record in the array th
    first = artist[1][0]       # Finds the first letter of the first name, sec
    second = artist[0][0]
    year = artist[2]

    if year > youngYear:       # Compares the year of the current record to th
        youngYear = year
        youngName = artist[0] + ' ' + artist[1]

    label = str(first) + str(second) + str(year)    # Formats the label

    theLabels.append(label)    # Adds the label to the array theLabels

for label in theLabels:        # Iterates through the labels in theLabels and
    print(label)

print("The youngest artist is:", youngName, "with year of birth", youngYear)
```

15/15

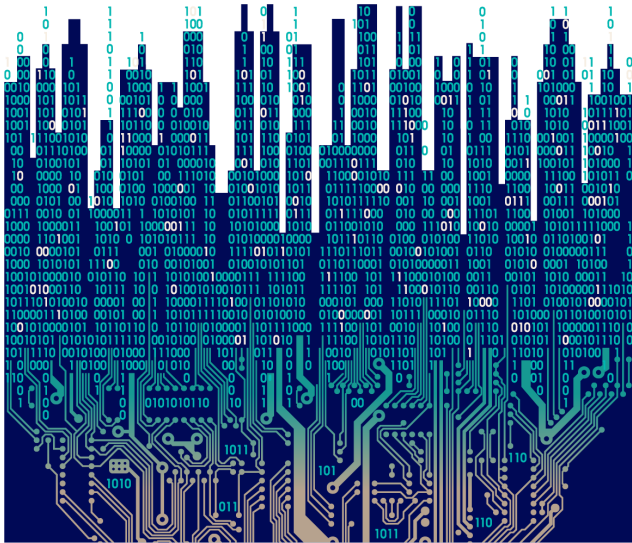
Question types?

Poll.

Please click the appropriate box to say which type of question you feel least prepared for:

- fix errors and add/amend code
- add or amend lines of code
- put lines of code in order
- add blocks of code
- questions with no scaffolding.

Support



Pearson
Edexcel

Tracking progress

- Detailed analysis available of your students' exam performance is available on ResultsPlus for Paper 1.
- It can help you identify topics and skills where students could benefit from further learning.
- You can see actual scores for each exam question for a student, class or group.
- You can understand how your students' performance compares with class and Pearson Edexcel national averages.
- You can acquire data that may support effective learning and teaching approaches.

Support materials

Planning

- **Interactive scheme of work** – supports the new practical approach and assessment requirements for GCSE Computer Science, and brings the content alive by showing Computer Science in action. Download and customise our sample Scheme of Work to help you plan your course.
- **Getting started guide** – provides detailed guidance on the specification and assessment
- **Programming Language subset (PLS)** – scopes precisely what Python constructs are required
- **Good Programming Practice Guide** – clarifies how computer science principles are supported by the PLS

Understanding the assessment requirements

- The sample assessment and specimen materials will help you familiarise yourself with the format and level of demand of the two exams and understand how your students' papers will be marked.

Marked exemplars

- We have developed a range of teaching and learning materials to help you prepare for the onscreen assessment, including a series of programming exercises, similar to those that students will tackle in the exam, with both commentaries and solutions.

Training events

- We host a series of online training events each year to help teachers deepen their understanding of our qualifications.

Published resources

- **Textbooks –** New and updated versions of the Pearson Edexcel Student Book, Revision Guide and Revision Workbook to support the new specification have been published.
- **Industry partners** – We have mapped the specification to specific content from our industry partners, to help you explore how to get the most impact from these resources.

How familiar are you with the key documents?

Poll.

Please click the appropriate box to say which documents you are familiar with:

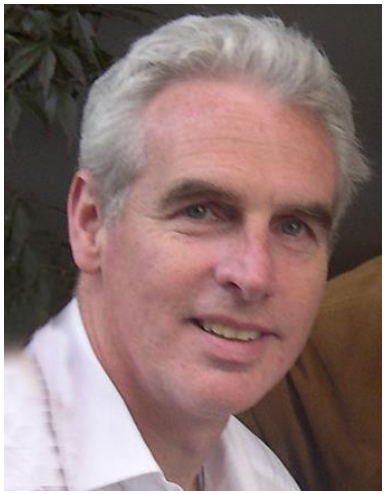
- Specification
- Sample assessment materials
- Practical Programming Subset (PPS)
- Getting Started guide (GSG)
- Good Programming Practice Guide.

Please use the chat window to leave any feedback on any additional support you require.

Personal support

Subject advisory service

led by Tim Brady, is a direct and personal source of help and support.



Email: TeacingComputerScience@pearson.com

Phone : (+44) 333 016 4160 (Mon-Fri, 9am-5pm GMT)

How confident are you?

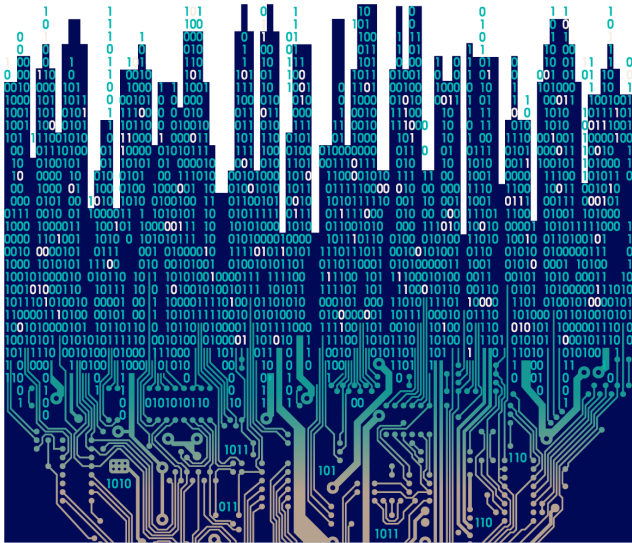
Poll.

Please click the appropriate box to say whether you are:

- confident about marking Paper 1
- confident about marking Paper 2
- confident about marking both Paper 1 and Paper 2
- so confident you are going to register to be an examiner next year.

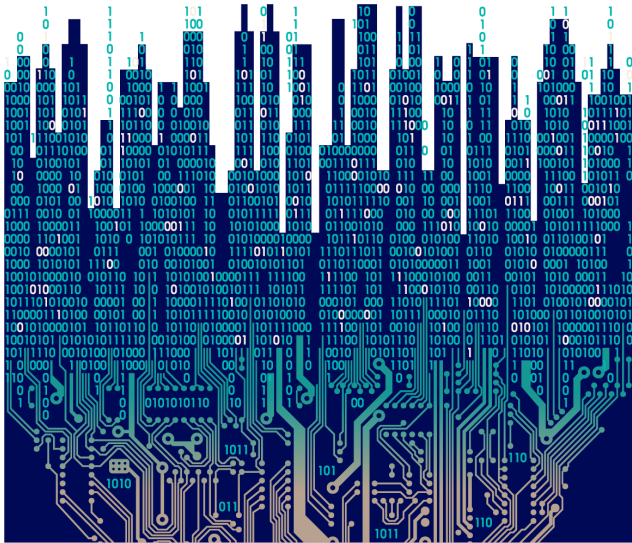
Please use the chat window to leave any feedback you'd like us to know about.

Final questions



Pearson
Edexcel

Thank you for attending.



Pearson
Edexcel

Pearson