Mark Scheme (Results)

Summer 2022

Pearson Edexcel International GCSE
In Computer Science (4CP0/2A)
Paper 02: 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

Summer 2022
Question Paper Log Number P72406A
Publications Code 4CP0_2A_2206_MS

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.

Theory

Question	mp	Answer	Additional Guidance	Mark
1 (a)	A1	**The only correct answer is D** A is not correct because the value can be used more than once B is not correct because the value does not always have to be input C is not correct because the value does not always have to be used in a calculation		(1)

Question	mp	Answer	Additional Guidance	Mark
1 (b)	B1	**The only correct answer is B** A is not correct because this would not aid readability of the code C is not correct because this would not aid readability of the code D is not correct because this would not aid readability of the code		(1)

Question	mp	Answer	Additional Guidance	Mark
1 (c)	C1 C2	**Award 1 mark for each of:** **Data type** / **Example value** char / An example of any single letter, number or symbol real / An example of a number with a decimal point	Accept quotes around the character Real number does not need a number after the decimal point to be awarded the mark	(2)

Question	mp	Answer	Additional Guidance	Mark
1 (d)	D1	**Award up to 2 marks for a description such as:** • Boundary testing uses the most extreme valid data / minimum and maximum valid values (1) whereas erroneous testing uses invalid data (1)		(2)

Question	mp	Answer	Additional Guidance	Mark
1 (e)	E1	Logic/logical		(1)

Question	mp		Additional Guidance	Mark
2 (a)(i)	A1	3/8/20		(1)
2 (a)(ii)	A2	7		(1)
2 (a) (iii)	A3	21/23		(1)
2 (a)(iv)	A4	check/count/pNumber		(1)
2 (a)(v)	A5	pNumber		(1)

Question	mp	Answer	Additional Guidance	Mark
4 (a)	A1	**The only correct answer is B** A is not correct because this is decryption C is not correct because this is not an encryption D is not correct because this is not an encryption		(1)

Question	mp	Answer	Additional Guidance	Mark
4 (b)	B1 B2 B3	**Award 1 mark for each correct symbol** **Word** M A Y **Symbol** 		(3)

Question	mp	Answer	Additional Guidance	Mark
4 (c)(i)	C1	5		(1)
4 (c)(ii)	C2	TIHYSEMNEEERNA		(1)
4 (c)(iii)	C3	Rail (Fence cipher)		(1)

Question	mp	Answer	Additional Guidance	Mark
5 (b)(i)	B1 B2 B3	**Award one mark for each:** • Pass 1 correct in first row of the response (1) • Pass 2 **OR** 3 correct in any row of the response (1) • All passes correct and no extra passes (1) 2 3 8 1 9 10 2 3 1 8 9 10 2 1 3 8 9 10 1 2 3 8 9 10	If the final row (1, 2, 3, 8, 9, 10) is repeated treat as the same row.	(3)
5 (b)(ii)	B4 B5	**Award up to 2 marks for a linked explanation such as:** • The sorting is done in the same place as the original data (1), which means only one additional variable is needed to store the value being swapped (when the values are out of order (1) • All sort and swapping operations are done on the original data (1) which means the amount of memory needed is constant (1) • The list does not need to be split (1), which would use more memory (1)		(2)

Question	mp	Answer	Additional Guidance	Mark
5 (c)	C1 C2 C3	**Award up to 3 marks for a linked description that includes:** • Start at the beginning of the list (1) • Compare each value in the list with the search item (1) • Repeat until a match is found / the end of the list is reached (1)		(3)

Code - Python

Question	mp	Answer	Additional Guidance	Mark
1 (f)	**Award 1 mark for each of:**			
	F1	Bracket added and no other amendments made to the line (1)		
	F2	== becomes = (1)		
	F3	" moved to after **is** and before **,** OR , area changed to +str(area) (1) Must still include the print statement		(3)
Code examples				

Python		`length = float(input("Enter the first number "))`
		
		`area = length * length`
		
		`print("The area of the square is" , area)`		

Question	mp	Answer	Additional Guidance	Mark
1 (g)	**Award 1 mark for each of:**			
	G1	At least 1 condition correct (1)		
	G2	At least 1 > or < condition and message match (1)		
	G3	At least 1 output includes letter and stored letter with an appropriate message (1)		
	G4	All conditions and outputs correct (1)		(4)
Code examples				

Python		`# Amend the code by completing the if statement`
		`if letter > storedLetter:`
		`    print(letter + " is later in the alphabet than " + storedLetter)`
		`elif letter == storedLetter:`
		`    print(letter + " is the same as " + storedLetter)`
		`else:`
		`    print(letter + " is earlier in the alphabet than " + storedLetter)`		

Question	mp		Additional Guidance	Mark
2 (b)	**Award 1 mark for each of:**		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. Do not penalise spelling mistakes in the input message.	(11)
	Evidence should be found in the Function			
	B1	Function interface is correct (1)		
	B2	Condition correct (for the number 1) (1)		
	B3	Loop initialised correctly (from 2 to the input number) (1)		
	B4	If condition correct (modulus) (1)		
	B5	Check returned (1)		
	Evidence should be found in the Main program			
	B6	Input of number stored as an integer variable (1)		
	B7	Input includes a space/colon / new line before the input value		
	B8	Function called correctly (1)		
	B9	Display includes number and message for either a prime number **OR** not (1)		
	Overall			
	B10	Compiling without syntax errors (1)		
	B11	Executing and producing the correct output (1)		
Code examples				
Python				

```python
# Write the function here
def checkPrime(pNumber):
    if pNumber == 1:
        check = False
    else:
        check = True
        for count in range(2,pNumber):
            if pNumber % count == 0 :
                check = False
    return check

# Write the main program here
number = int(input("Enter a number: "))
result = checkPrime(number)
if result == True:
    print(number,"is a prime number")
else:
    print(number,"is not a prime number")
```

Question	mp		Additional Guidance	Mark
3	**Award 1 mark for each of:**			
	A1	At least one variable with a meaningful name (1)		
	A2	At least one condition and price correct (ignore order) (1)		
	A3	All conditions and prices correct (1)		
	A4	Calculation for Total cost is correct (1)		
	A5	Price per textbook and total cost displayed (1)		
	A6	Calculation of cost and display of output are not included in the if statement (1)		(6)
Code examples				
Python				

```python
# Q03

#  Initialise variables
quantity = 0
price = 0
cost = 0

#  Print prompt and take number of textbooks required
quantity = int(input("Please enter the quantity of books required: "))

#  Generate and display the price per textbook and the total cost
if quantity >= 1 and quantity <= 5:
    price = 20
elif quantity < 10:
    price = 15
else:
    price = 12

cost = price * quantity
print("The price per book is",price)
print("The total cost is",cost)
```

Question	mp	Answe	Additional Guidance	Mark
5 (a)	**Award 1 mark for each of:**			
	A1	Variable to store a number of incorrect passwords (1)		
	A2	Loop used (1)		
	A3	Each password checked to see if the first character is an uppercase letter (1)		
	A4	Each password checked to see if it contains a digit (1)		
	A5	Only check for digits if first character is uppercase OR Only checks if first character is uppercase if at least one digit is present (1)		
	A6	Number of incorrect passwords incremented correctly (1) (allow follow through)		
	A7	All incorrect passwords displayed (1) (allow follow through but must try to display those without uppercase and those without digits)		
	A8	Text file closed (1)		
	A9	Number of incorrect passwords displayed (1) (allow follow through)		(9)
Code examples				
Python				

```python
# Add your code here
incorrectPasswords = 0

for password in passwordFile:
    valid = False

    if password[0] in alphabet:
        index = 1
        while (valid == False) and (index in range (1, len(password))):
            if password[index]in digit:
                valid = True
            else:
                index = index + 1

    if not valid:
        incorrectPasswords = incorrectPasswords + 1
        print(password)

print(incorrectPasswords,"passwords are incorrect.")

passwordFile.close()
```

Question	mp	Answer	Additional Guidance	Mark
6	A1	At least two appropriate subprograms used		(1)
	A2	Repeat guess until the word has been guessed **OR** no attempts left		(1)
	A3	Generate and display the number of attempts left	Must see working correctly	(1)
	A4	Request input of the word		(1)
	A5	Validate the input of the word to ensure it is the same length as the random word		(1)
	A6	Check to see if the guess matches the random word		(1)
	A7	If the word has been guessed, display message including the random word and number of attempts		(1)
	A8	Keep track and display letters that are in the word	Must see working	(1)
	A9	Keep track and display letters that are not in the word	Must see working	(1)
	A10	Letters appear only once in each list	Must see working	(1)
	A11	Display lose message including the random word		(1)

Band 1 (1-3 marks)	Band 2 (4-6 marks)	Band 3 (7-9 marks)	Mark
Little attempt to decompose into component parts	Some attempt to decompose 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 examples

Python

Subprograms

```
# Add your subprograms here
def checkInput(pWordLength):
    # Validate input to make sure it is the same length as the secret word
    check = ""
    while len(check) != pWordLength:
        check = input("Enter your guess. The secret word has " +str(pWordLength)+ " letters: ")
    return(check)

def checkLettersInWord(pGuess, pWordToGuess, pLetters, type):
    for letter in pGuess: # Check each letter in the guess
        if type == 0: # The check to carry out is for correct letters
            # If the letter is in the secret word and not already in the correctLetters, add it
            if letter in pWordToGuess and letter not in pLetters:
                pLetters = pLetters + letter
        else: # The check to carry out is for wrong letters
            # If the letter is not in the secret word and not already in the wrongLetters, add
            if letter not in pWordToGuess and letter not in pLetters:
                pLetters = pLetters + letter
    return pLetters

def display(pCorrectLetters, pWrongLetters):
    # Display the correct and wrong letters
    if len(pCorrectLetters) > 0:
        print("Letter(s) in the secret word:",pCorrectLetters)

    if len(pWrongLetters) > 0:
        print("Letter(s) not in the secret word:",pWrongLetters)
```

Main program

```
# Add your main program code here
wordLength = len(wordToGuess)
maxAttempts = wordLength + 3
numAttempts = 0
correctLetters = ""
wrongLetters = ""
guessed = False

# Loop until the no attempts are left or the secret word has been guessed
while numAttempts < maxAttempts and guessed == False:
    print("You have",maxAttempts - numAttempts,"attempts to guess the secret word.")
    numAttempts = numAttempts + 1

    guess = checkInput(wordLength)  # Validate the input

    if guess == wordToGuess: # Check to see if the secret word has been guessed
        guessed = True
        print("Well done. You guessed the secret word was",wordToGuess,"in",numAttempts,"attempts.")

    else: # If it hasn't been guessed check for correct and wrong letters
        correctLetters = checkLettersInWord(guess,wordToGuess, correctLetters, 0)
        wrongLetters = checkLettersInWord(guess,wordToGuess, wrongLetters, 1)
        display(correctLetters, wrongLetters)

# Display game over message if attempts have run out and the secret word has not been guessed
if numAttempts == maxAttempts and guessed == False:
    print("Game over. You did not guess that the secret word was",wordToGuess)
```