

**Cambridge Assessment International Education**

Cambridge International General Certificate of Secondary Education

**COMPUTER SCIENCE 0478/0984/2210/21**

Paper 2 Problem-solving and Programming **May/June 2021**

**MARK SCHEME**

Pre-Standardisation

**MAXIMUM MARK: 50**

**Generic Marking Principles**

These general marking principles must be applied by all examiners when marking candidate answers. They should be applied alongside the specific content of the mark scheme or generic level descriptors for a question. Each question paper and mark scheme will also comply with these marking principles.

|  |
| --- |
| GENERIC MARKING PRINCIPLE 1:  Marks must be awarded in line with:   * the specific content of the mark scheme or the generic level descriptors for the question * the specific skills defined in the mark scheme or in the generic level descriptors for the question * the standard of response required by a candidate as exemplified by the standardisation scripts. |
| GENERIC MARKING PRINCIPLE 2:  Marks awarded are always **whole marks** (not half marks, or other fractions). |
| GENERIC MARKING PRINCIPLE 3:  Marks must be awarded **positively**:   * marks are awarded for correct/valid answers, as defined in the mark scheme. However, credit is given for valid answers which go beyond the scope of the syllabus and mark scheme, referring to your Team Leader as appropriate * marks are awarded when candidates clearly demonstrate what they know and can do * marks are not deducted for errors * marks are not deducted for omissions * answers should only be judged on the quality of spelling, punctuation and grammar when these features are specifically assessed by the question as indicated by the mark scheme. The meaning, however, should be unambiguous. |
| GENERIC MARKING PRINCIPLE 4:  Rules must be applied consistently e.g. in situations where candidates have not followed instructions or in the application of generic level descriptors. |
| GENERIC MARKING PRINCIPLE 5:  Marks should be awarded using the full range of marks defined in the mark scheme for the question (however; the use of the full mark range may be limited according to the quality of the candidate responses seen). |
| GENERIC MARKING PRINCIPLE 6:  Marks awarded are based solely on the requirements as defined in the mark scheme. Marks should not be awarded with grade thresholds or grade descriptors in mind. |

**Please note the following further points:**

Please also read the additional guidance in the mark scheme, it provides further information about how to mark the question.

Please annotate **every** question on your scripts. The number of ticks given **must** match the number of marks given. If you award a benefit of doubt (BOD) mark, this **must** also have a **tick**.

The sections in brackets in the mark scheme are not necessary in the candidates answer.

The words in **bold** in the mark scheme are important text that needs to be present, or some notion of it needs to be present. It doesn’t have to be the exact word, but something close to the meaning.

If a word is underlined, this **exact** word must be present.

A single forward slash means this is an alternative word. A double forward slash means that this is an alternative mark point.

Red text in the mark scheme is a response that we think it just about okay, but that will not be published as a response.

Ellipsis (…) on the end of one mark point and the start of the next means that the candidate **cannot** get the second mark point without being awarded the first one. If a MP has ellipsis at the beginning, but there are no ellipsis on the MP before it, then this is just a follow-on sentence and **can** be awarded **without** the previous mark point.

Please inform your team leader when you have submitted your standardisation scripts. They may have many examiners that they are monitoring, so may not see your submission immediately.

The mark scheme has been agreed at standardisation, so it is the AE’s responsibility to apply it. You may have differing opinions on how the paper should be marked, but the mark scheme is what has been agreed by a panel and AE’s are required to apply it.

Please mark your allocation at a steady rate. If you are not able to mark your allocation for a number of days, inform your team leader, do not wait for them to have to chase you. Please ensure that you meet the 40% deadline. If you are not close to or meeting this at the 40% deadline, some of your allocation may be reallocated. If you are going to struggle with this deadline, you **must** inform your team leader ASAP.

There is a blank page(s) in the exam paper. This **must** be annotated with a **SEEN** annotation, to indicate it has been checked for any further responses. Also, any blank responses must have a SEEN annotation, as every question must be annotated. If you do not annotate correctly, you may be stopped by your TL.

If a candidate writes outside the zoned area for the question, this must be linked to the response, even if it is not awarded a mark. This demonstrates at grade review that you did read this part of the response.

| **Question** | **Answer** | **Marks** | **Notes** |
| --- | --- | --- | --- |
|  | **Section A** |  |  |
| 1(a)(i) | Many correct answers. They must be meaningful and related to **Task 1.** The names are examples only.  **One** mark per bullet point   * Constant MaxCandidates * Value 4 * Use Storing the maximum number of candidates for the election | **3** | Range of possible answers. |
| 1(a)(ii) | Many correct answers. They must be meaningful and related to **Task 1.** The names are examples only.  **One** mark per bullet point   * Variable NumberCandidates * Use Storing the number of candidates in the election (for a tutor group) * Array CandidateNames * Use Storing the names of the candidates for the election | **4** | Range of possible answers, these are examples only. |
| 1(b) | **One** mark per bullet point **(Max 4)**   * Change the input message to state the maximum number of candidates is 8 * Change the value of the MaxCandidates constant/variable to 8 * Change the loop limit to up to 8 * Change the validation to allow input up to 8 * Change the array size to ensure sufficient capacity to store up to 8 names * Change the counters to ensure votes can be counted for up to 8 candidates | **4** | Range of possible answers, accept other correct statements. |
| 1(c) | Any **five** from   1. Input with message to enter unique voter number 2. Validation of unique voter number 3. Use flags to check if current student has voted 4. Use of flags/other means to check if all entries in array for previous voters have been checked 5. Entry of current voter into correct position in array 6. Message to warn voter who has already voted 7. … and not allow them to vote   **Example answer**  REPEAT  OUTPUT "Please enter your unique voter number"  INPUT UniqueVoterNumber  FoundFlag ← False  AllNumbersChecked ← False  Counter ← 0  WHILE FoundFlag = False AND AllNumbersChecked = False  IF StudentNumbers[Counter] = ""  THEN  AllNumbersChecked = True  StudentNumbers[Counter] ← UniqueVoterNumber  ELSE  IF UniqueVoterNumber = StudentNumbers[Counter]  THEN  FoundFlag = True  PRINT “Sorry, you have already voted”  ELSE  Counter = Counter + 1  ENDIF  ENDIF  ENDWHILE  IF FoundFlag = False  THEN  OUTPUT "Please enter the code of your chosen candidate"  INPUT Vote  ENDIF | **5** | Range of possible answers, accept any recognisable program code, pseudocode or flowchart. |
| 1(d) | Explanation of how the program does the following:  Any **four** from   1. For each candidate, find out how many votes were cast. 2. Find out how many votes were cast in the election. 3. Calculate the percentage of votes for each candidate. 4. … excluding abstentions 5. Display the results with appropriate messages. 6. Display the number of votes cast and the number of abstentions with appropriate message. | **4** | Take care that the answer does more than state the task in the pre-release Task 3.  Programming statements can be used but must be explained. |

| **Question** | **Answer** | **Marks** | **Notes** |
| --- | --- | --- | --- |
|  | **Section B** |  |  |
| **2** | |  |  |  |  | | --- | --- | --- | --- | | **Statement** | **Validation** | **Verification** | **Both** | | Entering the data twice to check if both entries are the same. |  | ✓ |  | | Automatically checking that only numeric data has been entered. | ✓ |  |  | | Checking data entered into a computer system before it is stored or processed. |  |  | ✓ | | Visually checking that no errors have been introduced during data entry. |  | ✓ |  |   **One** mark per correct column. | **3** | CAO |
| 3 | **One** mark per bullet point  37   * Data type name Integer * Data type description (Any) whole number   Cambridge2021   * Data type name String * Data type description A group of characters/text   47.86   * Data type name Real * Data type description (Any real) number that could be a whole number or a fraction | **6** | Accept alternative correct answers related to programming data types.  Review at STM States each data type must be different in the question. |
| 4(a) | **One** mark per bullet point **(Max 3)**   * Marks are input into the array Score[] * Marks are checked against a range of boundaries * … and a grade is assigned to each mark that has been input * … then stored in the array Grade[] * The algorithm finishes after 30 marks have been input | **3** | Range of possible answers, accept other correct statements. |
| 4(b) | **One** mark per bullet point **(Max three)**   * Correct loop, including counter if not a FOR loop * Correct output of Score[] * Correct output of Grade[] * Suitable messages/text in output for both arrays   Example answers  Count ← 0  REPEAT  PRINT "Student: ", Count, " Mark: ", Score[Count], " Grade: ",Grade[Count]  Count ← Count + 1  UNTIL Count = 30  Or  FOR Count ← 0 TO 29  PRINT "Student: ", Count, " Mark: ", Score[Count], " Grade: ", Grade[Count]  NEXT | **3** | Allow any correct solution. |
| 4(c) | Any **three** correct statements (max **three**) e.g.   * Add an input facility to allow teachers to enter the class size * Add a variable to store the input class size * Use the class size variable as the terminating condition for the loop * Make sure the arrays are sufficiently large to accommodate the largest possible class size | **3** | Allow alternative solutions. Declaration of arrays not on syllabus, but allow it if candidates make appropriate reference to changing the size of the array. |
| 5(a) | |  |  |  |  | | --- | --- | --- | --- | | **Value** | **Diff1** | **Diff2** | **OUTPUT** | | 50 | 50 | 0 | Accept: Extreme | | 75 | 25 | 25 | Accept: Normal | | 99 | 1 | 49 | Accept: Normal | | 28 |  |  | Reject: Abnormal | | 82 | 18 | 32 | Accept: Normal | | 150 |  |  | Reject: Abnormal | | -1 |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |   **One** mark for each correct column (max **four**) | **4** | STM discussion - what happens if the candidates write their entries consecutively at the top of each column? |
| 5(b) | **One** mark per bullet point **(**max **two)**   * It is a range/validation check * … that checks if numbers entered are between 50 and 100 (inclusive) * … it identifies the type of test data | **2** | Range of possible answers, accept other correct statements. |
| 6(a) | The data in the ID column/field is unique in each row/record | **1** | Accept other correct statements. |
| 6(b) | 18 | **1** | CAO |
| 6(c) | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Field: | ID | GenreName | Overdue |  |  | | Table: | GENRE | GENRE | GENRE |  |  | | Sort: |  |  | Descending |  |  | | Show: | 🗹 | 🗹 | 🗹 | 🞏 | 🞏 | | Criteria: |  |  | >0 |  |  | | or: |  |  |  |  |  |   **One** mark for the correct fields present and correctly named  **One** mark for correct table name and show box in all columns  **One** mark for correct sorting  **One** mark for correct search criterion | **4** | Fieldnames must match those given in database table exactly. Ignore case. |