Cambridge Assessment International Education

Cambridge O Level

CANDIDATE NAME

CENTRE NUMBER | | | | | |

CANDIDATE NUMBER | | | | |

8250423928

COMPUTER SCIENCE **2210/21**

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

1 hour 45 minutes

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- **Do not attempt Tasks 1, 2 and 3** in the copy of the pre-release material on page 2; these are for information only.
- Use a black or dark blue pen. You may use an HB pencil for any diagrams or graphs.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- Calculators must **not** be used in this paper.

INFORMATION

- The total mark for this paper is 50.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

© UCLES 2021

Section A

You are advised to spend no longer than 40 minutes answering this section.

Here is a copy of the pre-release material.

DO NOT attempt Tasks 1, 2 and 3 now.

Use the pre-release material and your experience from attempting the tasks before the examination to answer Question 1.

Pre-release Material

A system is required to record and count votes for candidates in school council elections. The voting system will allow for one representative to be elected from a tutor group. The school has between 28 and 35 students in each tutor group, five year groups named Year 7 to Year 11, and there are six tutor groups in each year group. Tutor group names are their year group followed by a single letter e.g. 7A, 7B, etc.

All students are allowed to vote in the system. Each student may only vote once for a representative from their tutor group in the election.

Write and test a program or programs for the voting system.

- Your program or programs must include appropriate prompts for the entry of data; data must be validated on entry.
- Error messages and other output need to be set out clearly and understandably.
- All variables, constants and other identifiers must have meaningful names.

You will need to complete these **three** tasks. Each task must be fully tested.

Task 1 – Setting up the voting system to allow a tutor group to elect a representative.
Write a program to:

- allow the tutor to enter the name of the tutor group
- allow the tutor to enter the number of students in the tutor group
- allow the tutor to enter the number of candidates in the election; maximum of four candidates
- allow the tutor to enter the names of the candidates and store them in a suitable data structure
- allow each student to input their vote or to abstain
- count the votes for each candidate and student abstentions.

When all students have voted, display the name of the tutor group, the votes for each candidate and the name of the candidate who has won the election. If there is a tie for first place, display all candidates with the equal highest number of votes.

Task 2 – Checking that students only vote once.
Each student is given a unique voter number by their teacher.
Extend **Task 1** to achieve the following:

- Allow students to enter their unique voter number before casting their vote.
- Check whether the student has already voted:
  - if so, supply a suitable message and do **not** allow them to vote.
  - if not, store the unique voter number, but **not** their vote, in a suitable data structure, and add their vote to the relevant candidate count or abstention.

Task 3 – Showing statistics and dealing with a tie.
Extend **Task 2** to achieve the following:

- Calculate the percentage of the votes that each candidate received from the number of votes cast, excluding abstentions.
- Display the name of each candidate, the number of votes and the percentage of votes they received from the number of votes cast, excluding abstentions.
- Display the total number of votes cast in the election and the number of abstentions.
- In the event of a tie, allow the election to be immediately run again, with only the tied candidates as candidates, and all the students from the tutor group voting again.

© UCLES 2021

1 **(a)** All variables, constants and other identifiers must have meaningful names.

(i) Identify **one** constant you could have used for **Task 1**, give the value that would be assigned to it and its use.

Constant .....

Value .....

Use .....

.....

.....

[3]

(ii) Identify **one** variable and **one** array you could have used for **Task 1**.
Explain the use of each one.

Variable .....

Use .....

.....

.....

Array .....

Use .....

.....

.....

[4]

(b) Explain how you should change your program in **Task 1** to allow a tutor to enter up to eight candidates for the election.

.....

.....

.....

.....

.....

.....

.....

..... [4]

(c) Write an algorithm using pseudocode, programming statements **or** a flowchart to show how your program completes these parts of **Task 2**:

- Allows students to enter their unique voter number before casting their vote.
- Checks whether the student has already voted:
  - if so, supplies a suitable message and does **not** allow them to vote.
  - if not, stores the unique voter number, but **not** their vote, in a suitable data structure.

It is **not** necessary to show parts completed in **Task 1**, including counting of votes for each candidate.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

[5]

(d) Explain how your program completes these parts of **Task 3**:

- Calculate the percentage of the votes that each candidate received from the number of votes cast, excluding abstentions.
- Display the name of each candidate, the number of votes and the percentage of votes they received from the number of votes cast, excluding abstentions.
- Display the total number of votes cast in the election and the number of abstentions.

Any programming statements used in your answer must be fully explained.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [4]

Section B

2 Tick (✓) **one** box in each row to identify if the statement is about **validation**, **verification** or **both**.

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.			

[3]

3 Name and describe the most appropriate programming data type for each of the examples of data given. Each data type must be different.

Data: `37`

Data type name .....

Data type description .....

.....

.....

Data: `Cambridge2021`

Data type name .....

Data type description .....

.....

.....

Data: `47.86`

Data type name .....

Data type description .....

.....

.....

[6]

4 The pseudocode algorithm shown has been written by a teacher to enter marks for the students in her class and then to apply some simple processing.

```
Count ← 0
REPEAT
  INPUT Score[Count]
  IF Score[Count] >= 70
    THEN
      Grade[Count] ← "A"
    ELSE
      IF Score[Count] >= 60
        THEN
          Grade[Count] ← "B"
        ELSE
          IF Score[Count] >= 50
            THEN
              Grade[Count] ← "C"
            ELSE
              IF Score[Count] >= 40
                THEN
                  Grade[Count] ← "D"
                ELSE
                  IF Score[Count] >= 30
                    THEN
                      Grade[Count] ← "E"
                    ELSE
                      Grade[Count] ← "F"
                  ENDIF
              ENDIF
          ENDIF
      ENDIF
  ENDIF
  Count ← Count + 1
UNTIL Count = 30
```

(a) Describe what happens in this algorithm.

.....

.....

.....

.....

.....

.....

.....

..... [3]

(b) Write the pseudocode to output the contents of the arrays `Score[]` and `Grade[]` along with suitable messages.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [3]

(c) Describe how you could change the algorithm to allow teachers to use it with any size of class.

.....

.....

.....

.....

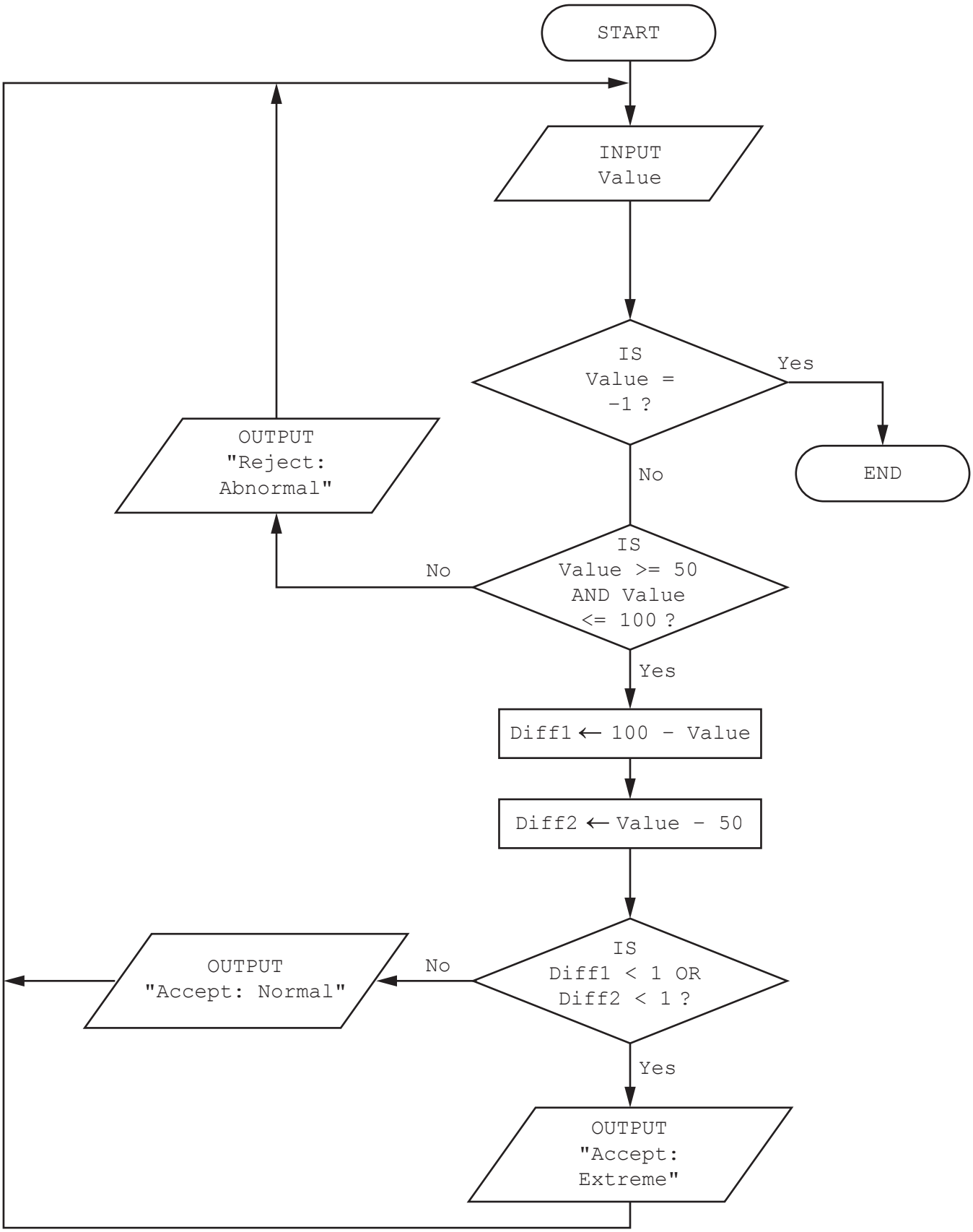
.....

.....

..... [3]

5 The flowchart represents an algorithm.

The algorithm will terminate if –1 is entered.



(a) Complete the trace table for the input data:

50, 75, 99, 28, 82, 150, –1, 672, 80

Value	Diff1	Diff2	OUTPUT

[4]

(b) Describe the purpose of the algorithm.

.....

.....

.....

..... [2]

6 A library uses a database table, GENRE, to keep a record of the number of books it has in each genre.

ID	GenreName	Total	Available	Loaned	Overdue
ABI	Autobiography	500	250	250	20
BIO	Biography	650	400	250	0
EDU	Education	20200	10000	10200	1250
FAN	Fantasy	1575	500	1075	13
GFI	General Fiction	35253	23520	11733	0
GNF	General Non-Fiction	25200	12020	13180	0
HFI	Historical Fiction	6300	3500	2800	0
HNF	Historical Non-Fiction	8000	1523	6477	0
HUM	Humour	13500	9580	3920	46
MYS	Mystery	26000	13269	12731	0
PFI	Political Fiction	23561	10523	13038	500
PNF	Political Non-Fiction	1823	750	1073	23
REF	Reference	374	374	0	0
ROM	Romance	18269	16800	1469	0
SAT	Satirical	23567	12500	11067	0
SCF	Science Fiction	36025	25000	11025	0
SPO	Sport	45720	32687	13033	3256
THR	Thriller	86000	46859	39141	0

(a) State the reason ID could be used as a primary key in the table GENRE.

.....

..... [1]

(b) State the number of records in the table GENRE.

.....

..... [1]

(c) Complete the query-by-example grid to display any genres with overdue books. Only display the ID, GenreName and Overdue fields in order of the number of books overdue from largest to smallest.

Field:					
Table:					
Sort:					
Show:	☐	☐	☐	☐	☐
Criteria:					
or:					

[4]

BLANK PAGE

© UCLES 2021

BLANK PAGE

© UCLES 2021

BLANK PAGE

Permission to reproduce items where third-party owned material protected by copyright is included has been sought and cleared where possible. Every reasonable effort has been made by the publisher (UCLES) to trace copyright holders, but if any items requiring clearance have unwittingly been included, the publisher will be pleased to make amends at the earliest possible opportunity.

To avoid the issue of disclosure of answer-related information to candidates, all copyright acknowledgements are reproduced online in the Cambridge Assessment International Education Copyright Acknowledgements Booklet. This is produced for each series of examinations and is freely available to download at www.cambridgeinternational.org after the live examination series.

Cambridge Assessment International Education is part of the Cambridge Assessment Group. Cambridge Assessment is the brand name of the University of Cambridge Local Examinations Syndicate (UCLES), which itself is a department of the University of Cambridge.

© UCLES 2021