GradeMax

CAMBRIDGE
International Education

Cambridge IGCSE™

COMPUTER SCIENCE **0478/23**

Paper 2 Algorithms, Programming and Logic **May/June 2026**

MARK SCHEME

Maximum Mark: 75

Published

This mark scheme contains the instructions and marking guidance that our examiners used to mark candidate responses for this component.

Before they start marking, examiners complete a standardisation process. They review a range of candidate responses and discuss the mark scheme in detail to agree which answers can and cannot be accepted. Examiners award marks where it is clear that candidate responses have met the requirements of the mark scheme. These requirements may vary between different components or different versions of the same component, depending on the exact requirements of the question and the candidate responses seen during the standardisation process.

Sometimes, our mark schemes may allow marks to be awarded to responses which contain elements that are factually incorrect or for content not covered by the syllabus. We do this in response to the demands of a specific question to support candidates and make sure that our assessments are fair. However, these allowances should not be used as a guide for teaching and learning.

We cannot discuss the mark scheme with schools, and we recommend that schools read the mark scheme together with the assessment material and the Principal Examiner Report for Teachers.

This document consists of **22** printed pages.

© Cambridge University Press & Assessment 2026

[Turn over

GradeMax

General marking principles

All examiners must apply these marking principles when marking candidate responses.

1	Examiners must award marks for each response in line with: • the specific **content** defined in the mark scheme • the specific **skills** defined in the mark scheme or in the level descriptions • the **standard of response** required as exemplified by the standardisation scripts.
2	Examiners must award **whole marks** (not half marks, or other fractions).
3	Examiners must award marks **positively**. This means that: • marks are not deducted for errors or omissions • marks are awarded for correct/valid answers as defined in the mark scheme and also for **other valid answers** which go beyond the scope of the syllabus and mark scheme referring to your team leader as appropriate • the quality of spelling, punctuation and grammar are judged **only** when the mark scheme indicates that these features are **specifically assessed** in the question. However, the meaning of all responses must be unambiguous.
4	Examiners must **consistently** apply the requirements of the mark scheme and any rules for what to do when candidates have not followed instructions correctly.
5	Examiners must use the **full range of marks** defined in the mark scheme, unless limited by the quality of the candidate responses seen.
6	Examiners must award marks according to the mark scheme and **must not** award marks with grade thresholds or grade descriptions in mind.

© Cambridge University Press & Assessment 2026

GradeMax

Annotations guidance for centres

Examiners use a system of annotations as a shorthand for communicating their marking decisions to one another. Examiners are trained during the standardisation process on how and when to use annotations. The purpose of annotations is to inform the standardisation and monitoring processes and guide the supervising examiners when they are checking the work of examiners within their team. The meaning of annotations and how they are used is specific to each component and is understood by all examiners who mark the component.

We publish annotations in our mark schemes to help centres understand the annotations they may see on copies of scripts. Note that there may not be a direct correlation between the number of annotations on a script and the mark awarded. Similarly, the use of an annotation may not be an indication of the quality of the response.

The annotations listed below were available to examiners marking this component in this series.

Annotations

Annotation	Meaning
	Correct point
	Incorrect point
	Follow through
	Repetition
	Ignore
	Benefit of doubt given
	Content of response too vague
	Not answered question
	Omission
	Section not relevant

© Cambridge University Press & Assessment 2026

Annotation	Meaning
~~	Section incorrect
Highlighter	Highlights part of the answer or shows structure of complex answers
SEEN	Page or response seen by examiner
A2	AO2 mark
A3	AO3 mark
NE	Not enough
R1	Required item one
R2	Required item two
R3	Required item three
✓ 1	Correct awarding one mark
✓ 2	Correct awarding two marks
✓ 3	Correct awarding three marks
✓ 4	Correct awarding four marks
✓ 5	Correct awarding five marks
✓ 6	Correct awarding six marks
✓ 7	Correct awarding seven marks
✓ 8	Correct awarding eight marks
✓ 9	Correct awarding nine marks

© Cambridge University Press & Assessment 2026

GradeMax

Mark scheme abbreviations

/	separates alternative words / phrases within a marking point
//	separates alternative answers within a marking point
underline	actual word given must be used by candidate (grammatical variants accepted)
Max	indicates the maximum number of marks that can be awarded
( )	the word / phrase in brackets is not required, but sets the context
MP	mark point

Note: No marks are awarded for using brand names of software packages or hardware

PapaCambridge papacambridge.com

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance
1	D	**1**	

Question	Answer	Marks	Guidance
2	C	**1**	

Question	Answer	Marks	Guidance
3(a)(i)	Normal (test data)	**1**	
3(a)(ii)	Extreme (test data)	**1**	accept Boundary
3(b)	**One** mark per mark point: • A value to instantly verify the accuracy of data • that is calculated from the input • An example, for example the last digit of an ISBN code on a book / bar code / credit card / parity bit	**3**	

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance

3(c)	One mark per mark point, **max 6**:
	
	• First input for new password
	• Checking the length of the new password
	• Means to re-renter the new password if the length check fails, until it passes
	• Re-input of new password …
	• **and** check if it matches the first input
	• Means to re-start, indefinitely, the input process from the beginning if the verification fails
	• At least **two** outputs seen, from input prompt / error messages / acceptance message.
	
	For example:
	
	`Accepted ← FALSE`
	`REPEAT`
	`  REPEAT`
	`  OUTPUT "Input your new password with at least 10 characters"`
	`  INPUT Password`
	`  IF LENGTH(Password) < 10`
	`  THEN`
	`  OUTPUT "Your password must contain at least 10 characters, please re-input"`
	`  ENDIF`
	`  UNTIL LENGTH(Password)>=10`
	`  OUTPUT "Please re-enter your new password for verification"`
	`  INPUT Password2`
	`  IF Password2 = Password`
	`  THEN`
	`  Accepted ← TRUE`
	`  OUTPUT "Your new password is accepted"`
	`  ELSE`
	`  OUTPUT "Your password has not been accepted, please try again"`
	`  ENDIF`
	`UNTIL Accepted`
		6	No array or variable declarations required for this algorithm
			
			Ignore case

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance
4(a)	**One** mark per mark point: • Line 05 / `DECLARE RunTime : STRING` should be `DECLARE RunTime : INTEGER` • Line 11 / `OUTPUT RunTime` should be `INPUT RunTime` • Line 13 / `RunSecs ← DIV(RunTime, 60)` should be `RunSecs ← MOD(RunTime, 60)` // `RunTime – RunMins * 60` • Line 16 / `Runners[Index] ← RunMins` should be `Runners [Index] ← RunName`	4	Ignore case

	`01 DECLARE Runners : ARRAY[1:250] OF STRING`
	`02 DECLARE Times : ARRAY[1:250] OF INTEGER`
	`03 DECLARE Index : INTEGER`
	`04 DECLARE RunName : STRING`
	`05 DECLARE RunTime : INTEGER`
	`06 DECLARE RunMins : INTEGER`
	`07 DECLARE RunSecs : INTEGER`
	`08 FOR Index ← 1 TO 250`
	`09  OUTPUT "Enter a runner's name and then time(in seconds) "`
	`10  INPUT RunName`
	`11  INPUT RunTime`
	`12  RunMins ← DIV(RunTime, 60)`
	`13  RunSecs ← MOD(RunTime, 60)`
	`14  OUTPUT "Runner's Name: ", RunName`
	`15  OUTPUT "Time: ", RunMins, " Minutes ", RunSecs, " Seconds"`
	`16  Runners[Index] ← RunName`
	`17  Times[Index] ← RunTime`
	`18 NEXT Index`		

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance
4(b)(i)	Any **five** from: • Use a condition-controlled loop inside the existing loop before the `RunTime` input • `REPEAT` // `WHILE RunTime <1 OR RunTime > 3600` (`DO`) • After line `11` / `INPUT RunTime`, … • insert a selection statement with a range check (and error message) • `IF RunTime < 1 OR RunTime > 3600` (not required for `WHILE` loop version) • `OUTPUT "The time must be positive and no greater than 3600 seconds"` • Finish the repeat loop before the calculations start / before line `12` • `UNTIL RunTime >0 AND RunTime <= 3600` // `ENDWHILE`	**5**	Ignore case BOD use of 0 as a valid input for MP 2, MP5 and MP8

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance

4(b)(i)	**Example:**
	
	01 DECLARE Runners : ARRAY[1:250] OF STRING
	02 DECLARE Times : ARRAY[1:250] OF INTEGER
	03 DECLARE Index : INTEGER
	04 DECLARE RunName : STRING
	05 DECLARE RunTime : INTEGER
	06 DECLARE RunMins : INTEGER
	07 DECLARE RunSecs : INTEGER
	08 FOR Index ← 1 TO 250
	09 OUTPUT "Please enter a runner's name and time in seconds "
	10 INPUT RunName
	REPEAT
	11 INPUT RunTime
	IF RunTime < 1 OR RunTime > 3600
	THEN
	OUTPUT "The time must be positive and not greater than 3600 seconds"
	ENDIF
	UNTIL RunTime >0 AND RunTime <= 3600
	12 RunMins ← DIV(RunTime, 60)
	13 RunSecs ← MOD(RunTime, 60)
	14 OUTPUT "Runner's Name: ", RunName
	15 OUTPUT "Time: ", RunMins, " Minutes ", RunSecs, "Seconds"
	16 Runners[Index] ← RunName
	17 Times[Index] ← RunTime
	18 NEXT Index		

4(b)(ii)	**One** mark from: Example: • The original FOR loop will always iterate 250 times, even if there are not 250 runners. • Only 250 runners are allowed in the array. • Two athletes could have the same name, so would be difficult to identify. • Answers related to validation or verification are acceptable.	**1**	

© Cambridge University Press & Assessment 2026

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>5(a)</td><td>One mark for each correct column.

<table>
<tr><th>PassCount</th><th>FailCount</th><th>Count</th><th>Size</th><th>OUTPUT</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td></td><td></td></tr>
<tr><td>1</td><td></td><td>1</td><td>49.8</td><td></td></tr>
<tr><td>2</td><td></td><td>2</td><td>48.7</td><td></td></tr>
<tr><td></td><td>1</td><td>3</td><td>62.5</td><td></td></tr>
<tr><td>3</td><td></td><td>4</td><td>49.9</td><td></td></tr>
<tr><td>4</td><td></td><td>5</td><td>30.0</td><td></td></tr>
<tr><td></td><td>2</td><td>6</td><td>58.2</td><td></td></tr>
<tr><td>5</td><td></td><td>7</td><td>–12.5</td><td></td></tr>
<tr><td></td><td>3</td><td>8</td><td>50.0</td><td></td></tr>
<tr><td>6</td><td></td><td>9</td><td>26.7</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td>Number passed: 6</td></tr>
<tr><td></td><td></td><td></td><td></td><td>Number failed: 3</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table></td><td>5</td><td>If no other marks awarded, allow 1 mark for correct initialisation of <code>PassCount</code>, <code>FailCount</code> and <code>Count</code> to 0

Repeated errors such as not initialising all columns, penalise once and FT

Ignore case in output, but punctuation must be correct</td></tr>
</table>

© Cambridge University Press & Assessment 2026

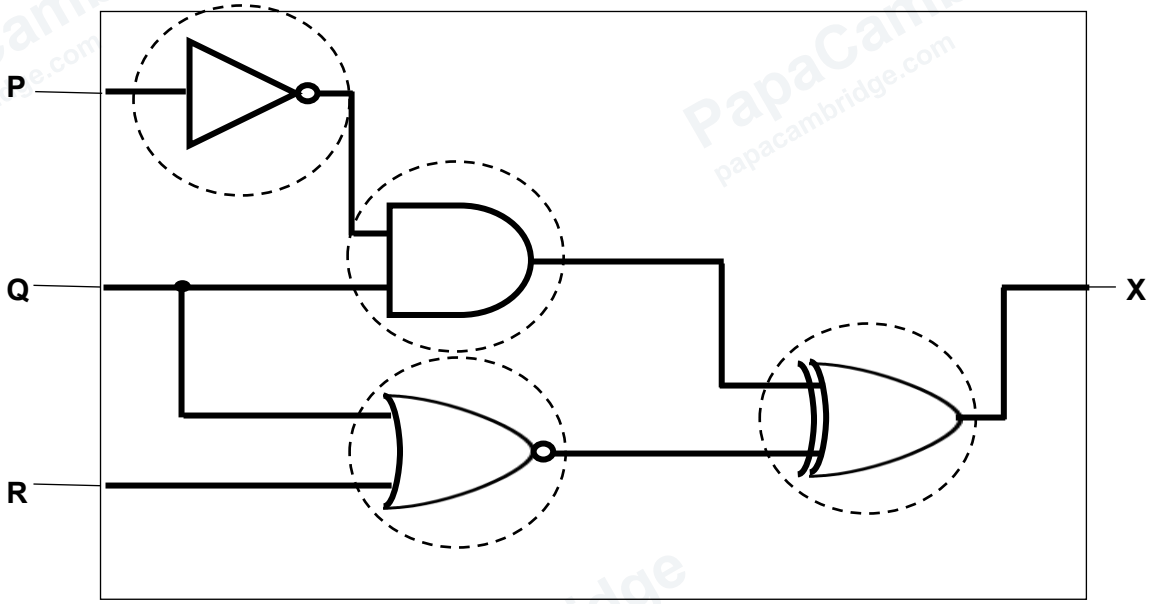
GradeMax

Question	Answer	Marks	Guidance
5(b)	Any **two** from: • Add another decision box after the input box and before the existing decision box. • Check for values <0 inside this decision box • If the condition fails, loop back to the input box for re-input // if the condition passes, go to the next decision box	**2**	This is a flowchart question, so answers must be in the context of a flowchart, or changes to the existing flowchart must be identifiable. Accept answers written as flowchart boxes.

Question	Answer	Marks	Guidance
6(a)	**One** mark per mark point: • Correct use of `LENGTH` function with `Fact` • Correct use of `UCASE` function with `Fact` • Contents of `LENGTH` and `UCASE` both output (can be both in same `OUTPUT` statement) **Example:** `Fact ← "I learnt to program in computer science"` **`OUTPUT LENGTH(Fact)`** **`OUTPUT UCASE(Fact)`**	**3**	Ignore case in keywords as pseudocode Case in any strings must match the question paper.
6(b)	**One** mark per mark point: • Correct use of `SUBSTRING` with `Fact` • and two other parameters (`13` and `7`) as an `OUTPUT` statement, accept 12 and 7 **Example:** `OUTPUT SUBSTRING(Fact, 13, 7,` accept `OUTPUT SUBSTRING(Fact, 12, 7)`	**2**	

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance
7(a)	**One** mark for each correct gate, with the correct input(s) as shown. 	**4**	

© Cambridge University Press & Assessment 2026

0478/23 Mai eme June 2026

GradeMax

Question	Answer	Marks	Guidance
7(b)	**Four** marks for eight correct outputs (X column) **Three** marks for six or seven correct outputs **Two** marks for four or five correct outputs **One** mark for two or three correct outputs **P** **Q** **R** **X** 0 0 0 **1** 0 0 1 **0** 0 1 0 **1** 0 1 1 **1** 1 0 0 **1** 1 0 1 **0** 1 1 0 **0** 1 1 1 **0**	**4**	

Question	Answer	Marks	Guidance
8(a)	**One** mark per mark point: • primary key: `SubjectCode` • Reason: it is unique for that subject and level	**2**	

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance
8(b)	**One** mark per mark point: • `PracticalExams/TheoryExams/MinimumClassSize/MaximumClassSize/ CourseLength/TheoryExams` • must be represented by whole numbers • `CourseWork` • this is a yes/no field, either there is coursework or there is **no** coursework	**4**	
8(c)	**One** mark per mark point: • At least two correct fields in `SELECT` • Remaining correct field in `SELECT` • Correct `FROM OPTION_SUBJECTS` • Correct `WHERE MinimumClassSize < 10` • Correct `ORDER BY SubjectCode;` Correct code: `SELECT` **`SubjectCode, SubjectName, Level`** `FROM` **`OPTION_SUBJECTS`** `WHERE` **`MinimumClassSize < 10`** `ORDER` **`BY SubjectCode (ASC);`**	**5**	Ignore case on field names, but spelling must match names in table. If space in field/table name, penalise once and FT. The default ordering is ascending, so award the mark if ASC is there or not. Penalise if anything else foe example if DESC or ALPHABETICAL is included instead.

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance
9	Max **six** marks from any **two** categories: **One** mark per mark point • Use of commenting • to describe sections of code and their function/purpose • to make the code easier for other programmers to understand // to make specific parts of the code easier to find **One** mark per mark point **(Max 3)** • Use of meaningful identifier (names) • for variables / constants / procedures / arrays • so that the purpose is suggested by the identifier / name • so that specific program sections / parts are easier to find / identify **One** mark per mark point **(Max 3)** • Use of procedures/functions/subprograms • so that the overall program is modular • so that repeated code is avoided • the program is easier to read • the program is shorter // fewer lines of code • the subprogram can be called anywhere it is needed (with different parameters) **One** mark per mark point • Use of whitespace/indentation • so that the program is laid out clearly/ is more readable • so that the beginning and end of groups of code, such as selection statements/iteration are clear	**6**	Max two per technique if the technique is not named.

© Cambridge University Press & Assessment 2026

GradeMax

Question	Answer	Marks	Guidance
10	Use the tables for AO2 and AO3 below to award a mark in a suitable band using a best fit approach, then add up the total. Marks are available for: • AO2 (maximum 9 marks) • AO3 (maximum 6 marks) **Data structures required** with names as given in the scenario: Arrays or lists `RiverNames[]`, `RiverStats[]`, `DaysCount[]` **Requirements (techniques)** **R1** Initialise `RiverStats` array, input and store the names of the rivers in the array, input daily BOD for 30 days and store it in the array (initialisation, iteration, input, output). **R2** Calculate the average daily reading for each river rounded to 2 d.p.. Count the number of moderately polluted days and the number of highly polluted days for each river (totalling, counting, iteration, rounding, selection). **R3** Output the average daily BOD readings, the number of moderately polluted days and the number of highly polluted days for each river. Identify and output the most polluted river (output, selection, finding max).	**15**	

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance
10	**Example 15-mark answer in pseudocode**		

```
// note that array and variable declarations are not required in candidate responses
DECLARE RiverStats : ARRAY[1:30, 1:5] OF REAL
DECLARE DaysCount : ARRAY[1:5, 1:2] OF INTEGER
DECLARE RiverNames : Array[1:5] OF STRING
DECLARE Index1, Index2 : Integer
DECLARE TotalBOD : REAL
DECLARE AverageBOD : REAL
DECLARE MostPollution : INTEGER
DECLARE MostPollRiver : String

// initialising array
FOR Index1 ← 1 TO 30
    FOR Index2 ← 1 TO 5
        RiverStats[Index1, Index2] ← 0
    NEXT Index2
NEXT Index1

// input river names
FOR Index1 ← 1 TO 5
    OUTPUT "Input the name of river ", Index1
    INPUT RiverNames[Index1]
NEXT Index1

// input of daily BOD readings
FOR Index1 ← 1 TO 30
    OUTPUT "Day ", Index1
    FOR Index2 ← 1 TO 5
        OUTPUT "Input the reading for ", RiverNames[Index2]
        INPUT RiverStats[Index1, Index2]
    NEXT Index2
NEXT Index1

// finding the average daily BOD and counting the number of moderately and highly polluted days
FOR Index1 ← 1 TO 5
```

© Cambridge University Press & Assessment 2026

Question	Answer	Marks	Guidance

10	`TotalBOD ← 0`
	`DaysCount[Index1, 1] ← 0`
	`DaysCount[Index1, 2] ← 0`
	`FOR Index2 ← 1 TO 30`
	`TotalBOD ← TotalBOD + RiverStats[Index2, Index1]`
	`IF RiverStats[Index2, Index1] >= 2 AND RiverStats[Index2, Index1] <= 8`
	`THEN`
	`DaysCount[Index1, 1] ← DaysCount[Index1, 1] + 1`
	`ENDIF`
	`IF RiverStats[Index2, Index1] > 8`
	`THEN`
	`DaysCount[Index1, 2] ← DaysCount[Index1, 2] + 1`
	`ENDIF`
	`NEXT Index2`
	
	`// calculating and rounding the average BOD to 2 dp`
	`AverageBOD ← ROUND(TotalBOD / 30, 2)`
	`OUTPUT RiverNames[Index1], " has an average daily pollution rate of ", AverageBOD`
	`OUTPUT "with ", DaysCount[Index1, 1], " moderately polluted day(s)"`
	`OUTPUT "and ", DaysCount[Index1, 2], " highly polluted day(s)"`
	`OUTPUT ""`
	`NEXT Index1`
	
	`// finding and outputting the most polluted river`
	`MostPollRiver ← RiverNames[1]`
	`MostPollution ← DaysCount[1, 2]`		

GradeMax

Question	Answer	Marks	Guidance

10	`FOR Index1 ← 2 TO 5`
	`  IF DaysCount[Index1, 2] > MostPollution`
	`  THEN`
	`  MostPollution ← DaysCount[Index1, 2]`
	`  MostPollRiver ← RiverNames[Index1]`
	`  ENDIF`
	`NEXT Index1`
	`OUTPUT "The most polluted river is ", MostPollRiver`
	`OUTPUT "It had ", MostPollution, " day(s) in the last 30 days with a high pollution reading"`		

© Cambridge University Press & Assessment 2026

Marking Instructions in italics			
AO2: Apply knowledge and understanding of the principles and concepts of computer science to a given context, including the analysis and design of computational or programming problems			
0	**1–3**	**4–6**	**7–9**
No creditable response.	At least one programming technique has been used. *Any use of selection, iteration, counting, totalling, input and output.*	Some programming techniques used are appropriate to the problem. *More than one technique seen applied to the scenario, check the list of techniques needed.*	The range of programming techniques used is appropriate to the problem. *All criteria stated for the scenario have been covered by the use of appropriate programming techniques, check the list of techniques needed.*
	Some data has been stored but not appropriately. *Any **use** of variables or arrays or other language dependent data structures e.g. Python lists.*	Some of the data structures chosen are appropriate and store some of the data required. *More than one data structure **used** to store data required by the scenario.*	The data structures chosen are appropriate and store all the data required. *The data structures **used** store all the data required by the scenario.*

© Cambridge University Press & Assessment 2026

GradeMax

Marking Instructions in italics			
AO3: Provide solutions to problems by: • **evaluating computer systems** • **making reasoned judgements** • **presenting conclusions**			
0	**1–2**	**3–4**	**5–6**
No creditable response.	Program seen without relevant comments.	Program seen with some relevant comment(s).	The program has been fully commented
	Some identifier names used are appropriate. *Some of the data structures used have meaningful names.*	The majority of identifiers used are appropriately named. *Most of the data structures used have meaningful names.*	Suitable identifiers with names meaningful to their purpose have been used throughout. *All the data structures used have meaningful names.*
	The solution is illogical.	The solution contains parts that may be illogical.	The program is in a logical order.
	The solution is inaccurate in many places. *Solution contains few lines of code with errors that attempt to perform a task given in the scenario.*	The solution contains parts that are inaccurate. *Solution contains lines of code with some errors that logically perform tasks given in the scenario. Ignore minor syntax errors.*	The solution is accurate. *Solution logically performs all the tasks given in the scenario. Ignore minor syntax errors.*
	The solution attempts at least one of the requirements. *Solution contains lines of code that attempt at least one task given in the scenario.*	The solution attempts to meet most of the requirements. *Solution contains lines of code that attempt most tasks given in the scenario.*	The solution meets all the requirements given in the question. *Solution performs all the tasks given in the scenario.*

© Cambridge University Press & Assessment 2026

0478/23 Mark Scheme June 2026