OCR
Oxford Cambridge and RSA

GCE

Computing

Unit **F452:** Programming Techniques and Logical Methods

Advanced Subsidiary GCE

Mark Scheme for June 2015

This mark scheme is published as an aid to teachers and students, to indicate the requirements of the examination. It shows the basis on which marks were awarded by examiners. It does not indicate the details of the discussions which took place at an examiners' meeting before marking commenced.

All examiners are instructed that alternative correct answers and unexpected approaches in candidates' scripts must be given marks that fairly reflect the relevant knowledge and skills demonstrated.

Mark schemes should be read in conjunction with the published question papers and the report on the examination.

OCR will not enter into any discussion or correspondence in connection with this mark scheme.

These are the annotations, (including abbreviations), including those used in scoris, which are used when marking

Annotation	Meaning
^	Omission mark
BOD	Benefit of the doubt
[	Subordinate clause / consequential error
✗	Incorrect point
E	Expansion of a point
FT	Follow through
NAQ	Not answered question
NBOD	No benefit of doubt given
P	Point being made
REP	Repeat
/	Slash / half-mark
✓	Correct point
TV	Too vague
0	Zero (big)
BP	Blank Page – this annotation **must** be used on all blank pages within an answer booklet (structured or unstructured) and on each page of an additional object where there is no candidate response.

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Mark</th><th>Guidance</th></tr>
<tr><td>1</td><td>a</td><td></td><td><i>Example:</i>
<table>
<tr><td>B</td><td>R</td><td>A</td><td>Z</td><td>I</td><td>L</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>1</td></tr>
<tr><td>C</td><td>Z</td><td>E</td><td>C</td><td>H</td><td></td><td>R</td><td>E</td><td>P</td><td>U</td><td>B</td><td>L</td><td></td><td>0</td><td>0</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td>5</td><td>2</td><td>:</td><td>2</td><td>1</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

<i>Design</i>
• Two team names/space for team names
• The number/space of goals scored clearly associated to each team in numeric format
• The current time displayed in data or mask

<i>Award up to 3 marks for any relevant assumptions eg</i>
• Limited number of characters for team name (depending on how it is displayed)
• Goals scored allow I or 2 digits
• Time in appropriate format (example mm:ss or 29m 35s)
• When centering text with an even number of characters, place extra space on the right/left</td><td>6</td><td>Accept indication of team names being positioned
Accept only first 3 assumptions.</td></tr>
</table>

Question			Answer	Mark	Guidance
	b		*Description, eg* • Program divided into separate self-contained/specific modules or tasks • Which can be written / tested individually • Modules can be subdivided further into smaller modules *Advantages, eg* • Modules can be shared between programmers • Reduced development time • Easier to maintain/debug • Modules allocated according to programmers expertise • Modules can be reused *Award up to two marks for the description and 1 mark for an advantage, wherever they appear in the response.*	3	Allow small enough to be programmed Do not accept easier to test
	c		• Identifier: Board/board • Number of dimensions: 2 • Most appropriate data type: Character/String	3	Allow suitable name Allow 2D Do not except alphanumeric
	d		*In order* • 15 • Column • Row	3	cao
	e	i	• The text will be displayed one position to the right from where it should be	1	
		ii	*eg* • In line 03, use Column + i - 1	1	
		iii	• Logic error	1	

Question			Answer	Mark	Guidance
	f		• The text to be displayed will not fit (within the array) • array out of bounds • …when it gets to column 16 • Type of error: runtime error	3	Accept logic error

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Mark</th><th>Guidance</th></tr>
<tr><td></td><td>g</td><td></td><td>

Example:

```
PROCEDURE DisplayGoal(Name, Number, Team)

   DisplayString("GOAL!", 1, 6)

   IF LENGTH(Name) > 12 THEN
      Name = LEFT(Name, 12)
   END IF
   DisplayString(Name, 2, 1)

   IF Number < 10 THEN
      DisplayString(Number, 2, 15)
   ELSE
      DisplayString(Number, 2, 14)
   END IF

   TeamLength = LENGTH(Team)
   IF  TeamLength > 15 THEN
      Team = LEFT(Team, 15)
      Pos = 1
   ELSE
      Pos = (15 – TeamLength) DIV 2
   END IF
   DisplayString(Team, 3, Pos)

END PROCEDURE
```

Award marks for:

- Requires name, number and team of scorer (as parameters or inputs)
- Correct use of DisplayString()
- Displays Name at position (2,1)
- Name truncated to 12/13 characters if necessary
- Displays Number at position (2,14)/(2,15)…
- ...Dealing correctly with 1 and 2 digit numbers
- Truncates team name if necessary
- Displays team name in centre

</td><td>8</td><td>Award marks for placing strings in the right position regardless of how the candidate has done this as long as the logic for doing so is clear. There is a separate mark for using DisplayString() to do this correctly (bullet 2).</td></tr>
<tr><td><b>2</b></td><td>a</td><td></td><td>

- 100011

</td><td>1</td><td></td></tr>
</table>

Question			Answer	Mark	Guidance
	b	i	• A name/symbol given to a variable/subroutine/etc. • d/ x	2	
		ii	• They don't have meaningful names / no indication of what the value contained represents	1	
	c		• Initialise the value of d (to the empty string) • Before it gets used in line 04 / as it may already contain a value/ to give it a starting value	2	
	d		• Joins the strings d and x into one string • Concatenation operator	2	Allow append Do not accept 'add'

Question			Answer	Mark	Guidance

	e		*Example (in pseudocode):*
			
			`// get the message`
			` INPUT Message`
			
			`// initialise output string`
			`Result = ""`
			
			`// loop through each letter`
			`FOR pos = 1 to LENGTH(Message)`
			
			   `// put the code of the letter in result`
			   `SELECT CASE MID(Message, pos, 1)`
			      `CASE "A" :`
			          `Result = Result & "0"`
			      `CASE "B" :`
			          `Result = Result & "11"`
			      `CASE "C" :`
			          `Result = Result & "100"`
			      `CASE "D" :`
			          `Result = Result & "101"`
			    `END SELECT`
			
			`END FOR`
			
			`//return the result`
			`OUTPUT Result`
			Award up to 5 marks for the algorithm:
			• INPUT message
			• Uses a loop
			• ... to correctly visit each character
			• Replaces “A”, “B”, “C” and “D” with correct code
			• Outputs the result
			
			Award up to 2 marks for style
			• Meaningful identifiers
			• Commenting
			• Indenting
				7	Cannot access bullet points 2, 3 & 5 without a loop
					
					Allow for python’s use of “str.replace”

Question			Answer	Mark	Guidance
	f		*eg* - Translator Diagnostics - reports when syntax errors are made and suggests solutions / example from code - Breakpoints - Allows the code to stop at chosen point - To check variables /example from code *Award one mark for each correctly named facility, and up to two marks for the description.*	6	Not dry running/trace tables - Steping - Executes each line in turn - To allow checking of path(s)/values - (Variable) watch - To monitor the status of variables (and objects)… - … as you step through code/as they change
3	a		- The data in the file is sorted - …according to a key field	2	
	b		Card identification number: - Data type: String/text/alphanumeric - Size: 6 Amount of credit: - Data type: Real/decimal/float/double - Size: 4 / 8 Free meals: - Data type: Boolean - Size: 1	6	Currency size 4/8 bytes
	c		- Adds up sizes eg 6 + 4+ 1 = 11 - Multiplies by 100 eg 11 * 100 = 1100 - Adds 10% for overhead eg 1100 * 1.1 = 1210 bytes	3	Allow follow through from b and between steps If 8 bytes used for Amount the corresponding answers are 15, 1500, 1650
	d		- In line 06 = is a relational/comparison operator - To check whether two items are the same - In line 07 = is an assignment operator - To change the value of a variable/NewAmount	4	Do not accept conditional operator for line 06

Question			Answer	Mark	Guidance
	e		*In order:* • FreeMeals (= TRUE) • Amount + 3.50 • FreeMeals	3	Accept a reasonable test for free meals. For this question case is irrelevant for variable names Allow correct cast i.e float(3.50, real(3.50) Do not accept £3.50

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Marks</th><th colspan="2">Guidance</th></tr>
<tr><td>3</td><td>f</td><td></td><td><em>Example:</em>
<pre>
// Input data and open files
INPUT NewID, NewAmt, NewFree

OPEN CardFile in READ MODE
OPEN NewFile in WRITE MODE

Inserted = FALSE

//loop through file
REPEAT
  READ Id, Amt, Free from CardFile

  //find insertion point
  IF Id > NewId
           AND Inserted = FALSE THEN
     WRITE NewID, NewAmt, NewFree
           to NewFile
     Inserted = TRUE
  END IF

  WRITE Id, Amt, Free to NewFile

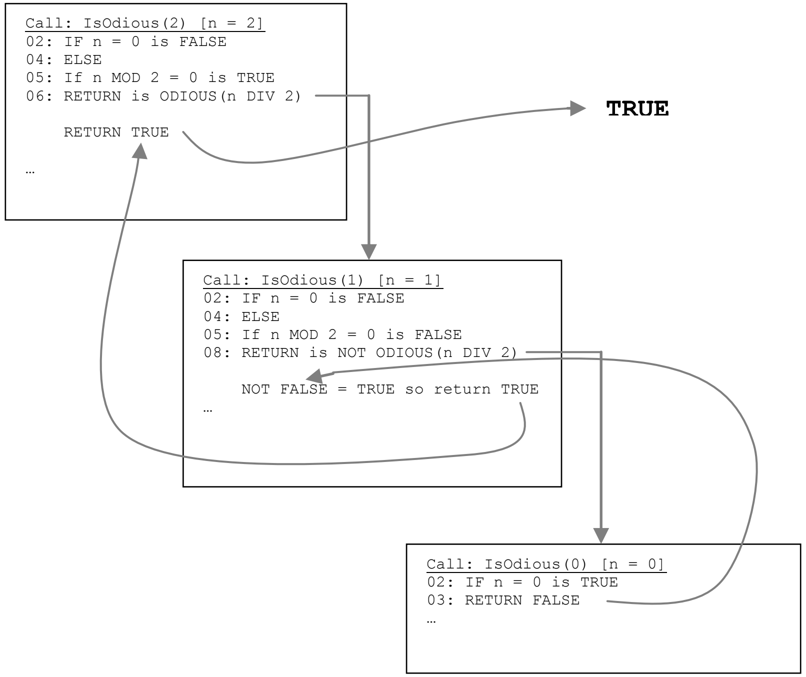
UNTIL CardFileAtEndOfFile

//if not yet inserted, append
IF Inserted = FALSE THEN
  WRITE NewID, NewAmt, NewFree
           to NewFile
END IF

//close files and replace
CLOSE CardFile
CLOSE NewFile

Replace CardFile with NewFile
</pre></td><td>8</td><td><strong>Content</strong></td><td><strong>Levels of response</strong>
<strong>High level response (6–8 marks)</strong>
Candidate offers a complete, working algorithm which both shows clearly how insertion point is determined and how the new file is produced. The algorithm is in correctly structured pseudocode with indentation, suitable identifiers or comments as appropriate. Technical terms and spelling will be used appropriately and correctly.
<strong>Medium level response (3–5 marks)</strong>
Candidate has an algorithm which is not fully explained or contains some errors for example in determining the insertion point. There is an attempt to structure the code correctly but may contain some errors, however the overall structure of the code can still be understood. Technical terms and spelling are mostly correct.
<strong>Low level response (0–2 marks)</strong>
Candidate's algorithm neither shows fully how the insertion point is determined nor explains how a record is inserted. The code is poorly structured or not structured at all, and errors with spelling and technical terms make the algorithm difficult to understand.</td></tr>
</table>

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Mark</th><th>Guidance</th></tr>
<tr><td>4</td><td>a</td><td></td><td>• 9 DIV <b>4</b> = 2
• 7 MOD 2 = <b>1</b>
• 1 MOD 3 = <b>1</b></td><td>3</td><td></td></tr>
<tr><td></td><td>b</td><td></td><td>• (WHILE) loop (on line 03) will repeat lines 04 to 08 or 03 to 09
• as long as n > 0 / until n is not > 0</td><td>2</td><td></td></tr>
<tr><td></td><td>c</td><td></td><td>• (line 02) Temp is set to TRUE
• Condition in line 3 is false (so it skips the loop)
• in line 10, it returns TEMP (which was set to TRUE)</td><td>3</td><td>Do not accept code line i.e. Temp=TRUE</td></tr>
<tr><td></td><td>d</td><td></td><td>
<table>
<tr><th>Line Number</th><th>n</th><th>temp</th><th>Comment</th></tr>
<tr><td>01</td><td>2</td><td></td><td></td></tr>
<tr><td>02</td><td></td><td>TRUE</td><td></td></tr>
<tr><td>03</td><td></td><td></td><td></td></tr>
<tr><td>04</td><td></td><td></td><td></td></tr>
<tr><td>07</td><td></td><td></td><td></td></tr>
<tr><td>08</td><td>1</td><td></td><td></td></tr>
<tr><td>09</td><td></td><td></td><td></td></tr>
<tr><td>03</td><td></td><td></td><td></td></tr>
<tr><td>04</td><td></td><td></td><td></td></tr>
<tr><td>05</td><td></td><td>FALSE</td><td></td></tr>
<tr><td>06</td><td>0</td><td></td><td></td></tr>
<tr><td>07</td><td></td><td></td><td></td></tr>
<tr><td>08</td><td>0</td><td></td><td></td></tr>
<tr><td>09</td><td></td><td></td><td></td></tr>
<tr><td>03</td><td></td><td></td><td></td></tr>
<tr><td>10</td><td></td><td></td><td></td></tr>
<tr><td>11</td><td></td><td></td><td></td></tr>
</table>
<i>One mark for each bulleted group of rows</i></td><td>6</td><td>If the value of n & temp does not change allow copy down.
Ignore comments
Start marking from top until wrong, then start marking from bottom until wrong</td></tr>
<tr><td></td><td>e</td><td></td><td>• The function calls itself on line 06
• The function <u>calls itself</u> on line <u>08</u></td><td>2</td><td></td></tr>
<tr><td></td><td>f</td><td>i</td><td><i>eg</i>
• Code is generally shorter
• (can be) closer to natural language description</td><td>1</td><td>Allow humans think recursively</td></tr>
</table>

Question			Answer	Mark	Guidance
		ii	*eg* • Uses more memory / resources • Difficult to trace /debug	1	Difficult to understand is TV Allow difficult to follow
	g		*Award one mark for each of the sets of steps below* (1st call: IsOdious(2) / n is 2) • (line 2) FALSE so go to line 04 (line 5) TRUE so go to line 06 • 2nd call : IsOdious(1) / n = 1 • (line 2) FALSE so go to line 04 (line5) FALSE so go to line 08 • 3rd call : IsOdious(0)/n = 0 Line(2) TRUE so go to line 03 Return **FALSE** • Return to 2nd call and complete line 08 so Return NOT(FALSE) = **TRUE** • Return to 1st call and complete line 06 so Return **TRUE**	6	Response may be as a diagram or as text but must fully address each point to get the marks. Recursive calls must be clearly indicated as such and returned values. Do not accept simply “goes back to line 2” if it is not clear that this is a new call. Example:  Call: IsOdious(2) [n = 2] 02: IF n = 0 is FALSE 04: ELSE 05: If n MOD 2 = 0 is TRUE 06: RETURN is ODIOUS(n DIV 2) RETURN TRUE … TRUE Call: IsOdious(1) [n = 1] 02: IF n = 0 is FALSE 04: ELSE 05: If n MOD 2 = 0 is FALSE 08: RETURN is NOT ODIOUS(n DIV 2) NOT FALSE = TRUE so return TRUE … Call: IsOdious(0) [n = 0] 02: IF n = 0 is TRUE 03: RETURN FALSE …

OCR (Oxford Cambridge and RSA Examinations)
1 Hills Road
Cambridge
CB1 2EU

OCR Customer Contact Centre

Education and Learning
Telephone: 01223 553998
Facsimile: 01223 552627
Email: general.qualifications@ocr.org.uk

www.ocr.org.uk

For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored

Oxford Cambridge and RSA Examinations
is a Company Limited by Guarantee
Registered in England
Registered Office; 1 Hills Road, Cambridge, CB1 2EU
Registered Company Number: 3484466
OCR is an exempt Charity

OCR (Oxford Cambridge and RSA Examinations)
Head office
Telephone: 01223 552552
Facsimile: 01223 552553

PART OF THE
CAMBRIDGE ASSESSMENT
GROUP

LLOYD'S REGISTER QUALITY ASSURANCE · ISO9001

UKAS
MANAGEMENT
SYSTEMS
001