



## **General Certificate of Education**

# **Computing 6510**

## **CPT4      Processing and Programming Techniques**

# **Mark Scheme**

*2008 examination - January series*

Mark schemes are prepared by the Principal Examiner and considered, together with the relevant questions, by a panel of subject teachers. This mark scheme includes any amendments made at the standardisation meeting attended by all examiners and is the scheme which was used by them in this examination. The standardisation meeting ensures that the mark scheme covers the candidates' responses to questions and that every examiner understands and applies it in the same correct way. As preparation for the standardisation meeting each examiner analyses a number of candidates' scripts: alternative answers not already covered by the mark scheme are discussed at the meeting and legislated for. If, after this meeting, examiners encounter unusual answers which have not been discussed at the meeting they are required to refer these to the Principal Examiner.

It must be stressed that a mark scheme is a working document, in many cases further developed and expanded on the basis of candidates' reactions to a particular paper. Assumptions about future mark schemes on the basis of one year's document should be avoided; whilst the guiding principles of assessment remain constant, details will change, depending on the content of a particular examination paper.

Further copies of this Mark Scheme are available to download from the AQA Website: [www.aqa.org.uk](http://www.aqa.org.uk)

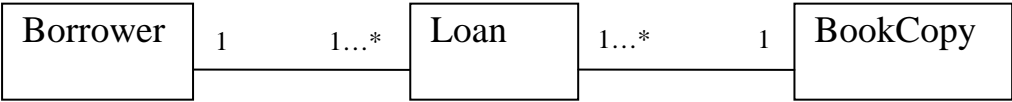
Copyright © 2008 AQA and its licensors. All rights reserved.

#### COPYRIGHT

AQA retains the copyright on all its publications. However, registered centres for AQA are permitted to copy material from this booklet for their own internal use, with the following important exception: AQA cannot give permission to centres to photocopy any material that is acknowledged to a third party even for internal use within the centre.

Set and published by the Assessment and Qualifications Alliance.

- 1 (a) BF2; 1
- 1 (b) -;1038; 2 1 mark for sign, 1 mark for value
- 1 (c) +;191;.125; A 1/8 3
- 1 (d) -;2;.03125; A 1/32 3 If incorrect part marks as follows  
 1 mark for complemented mantissa 01000001  
 1 mark for moving binary point 2 places
- 1 (e) To maximise precision in a given number of bits;  
 A to maximise accuracy in a given number of bits  
 To minimise rounding errors;  
 To allow a wider range of values to be stored; Max 2
- 2 (a)
- | Interactive  | Batch  |       |
|--|--|-------|
| User and computer in two way communication;// User communicates directly with the program while it is running; | Processing continues from beginning to end without user interaction/intervention;                        | max 1 |
| Program runs with higher priority  | Program run in background;// Program runs with lower priority  | max 1 |
| Processing carried out as users enter the data;<br>Results are available immediately;                          | Processing delayed until all data have been entered;<br>Results are available when the job is completed; | max 1 |
- Max 2
- 2 (b) Job ID; priority; user name; job delimiters; job completion time; estimated running/processor time; max length of time job can run for; start time of job; main memory required; file size; devices/hardware required (e.g. printer); compiler/assembler/software required; data/source code file required; output destination; what to do on non-successful completion of job; I/O Bound or Processor bound type of batch job; Max 3
- 2 (c) Process requires service from a resource;  
 Process is timed out// Time slice expires;  
 Process is pre-empted; Max 2
- 2 (d) Priority queue; 1
- 3 (a) Surface number; A layer/platter  
 Track/Cylinder Number;  
 Sector/Block Number; A segment/ cluster 3
- 3 (b) (i) Memory used for temporary storage of one or more disk blocks; in transit between disk and main memory; 2
- 3 (b) (ii) To allow for different speed of devices.  
 Logical records & physical records of different size; Max 1

- 3 (c) (i) Transfer Completed;  
Transfer aborted/failed/timed out;  
Handshaking 2
- 3 (c) (ii) Interrupting device/ source supplies;  
an offset/vector; **A** index/indexed address  
added to the base address; **A** base register Max 2
- Gives the start address of interrupt service routine/ ISR//  
Address vector table cell contains start address of ISR/  
**R** Interrupting device supplies start address of ISR 3
- 3 (c) (iii) a different routine can be easily introduced//  
routine can be relocated/ dynamically loaded; *or words to this effect*  
**A** The interrupting device only needs to supply a new offset 1
- 4 (a) 

```

classDiagram
    class Borrower
    class Loan
    class BookCopy
    Borrower "1" -- "1..*" Loan
    Loan "1..*" -- "1" BookCopy
    
```

1 mark for correct boxes  
1 mark for correct lines  
1 mark for correct line endings 3

4 (b) `Loan = class`  
Public  
Procedure CreateLoan  
Procedure DeleteLoan  
Procedure GetLoanDetails;  
Private  
Person: Borrower  
BookLoaned: BookCopy;  
DateOfLoan: Time/Date **A** string  
ReturnDate: Time/Date; **A** string  
End;

1 mark for Loan=Class + Public + Private + End  
1 mark for CreateLoan + DeleteLoan + GetLoanDetails  
1 mark for Person + BookLoaned  
1 mark for DateOfLoan + ReturnDate  
**A** any reasonable names for operations and data items. 4

4 (c) Add a new data item ShortLoan; of type Boolean; **A** loanlength; integer;  
**A** loantype; string;  
Modify the code for the operations; Max 2

5 (a) (i) Each accumulator bit is compared with its corresponding operand bit, if both are  
1 the result for this bit position is 1, otherwise 0;  
**A** by example 1

5 (a) (ii) AND #0F; **A** AND #CF  
Allow ft to (b) 1

6 (b) (i)

5 (b)

Label	Opcode	Operand	Comment	
	LD	015A	Load first character	
	AND	#0F;	And convert to a value	A AND #CF
	MUL	#10;	Move to upper nibble	
	ST	01A5;	Store in work area	A 01A6
	LD	015B	Load second character	
	AND	#0F;	And convert to a value	A AND #CF
	ADD	01A5;	Combine two values	
	ST	01A6;	Store result	

Or

	LD	015B	Load second character	
	AND	#0F;	And convert to a value	A AND #CF
	ST	;	Store in work area	A 01A6
	LD		Load first character	
	AND		And convert to a value	A AND #CF
	MUL		Move to upper nibble	
	ADD	;	Combine two values	
	ST	;	Store result	

6

6 (a) a procedure/routine that calls itself/ is defined in terms of itself;  
**A** Function instead of procedure  
**R** re-entrant **R** program **R** iteration

1

6 (b)(i)		Output
Procedure Call	T	
P <sub>1</sub>	<pre>       14      / \     8   18    / \   5  11             </pre>	
P <sub>2</sub>	<p style="text-align: center;">18</p> <div style="text-align: right; border: 1px solid black; padding: 2px; display: inline-block;">1 mark</div>	18;
P <sub>1</sub>	<pre>       14      / \     8   18    / \   5  11             </pre>	14
P <sub>3</sub>	<pre>       8      / \     5  11             </pre> <div style="text-align: right; border: 1px solid black; padding: 2px; display: inline-block;">1 mark</div>	
P <sub>4</sub>	<p style="text-align: center;">11</p> <div style="text-align: right; border: 1px solid black; padding: 2px; display: inline-block;">1 mark</div>	11
P <sub>3</sub>	<pre>       8      / \     5  11             </pre>	8
P <sub>5</sub>	<p style="text-align: center;">5</p> <div style="text-align: right; border: 1px solid black; padding: 2px; display: inline-block;">1 mark</div>	5
P <sub>3</sub>	<pre>       8      / \     5  11             </pre>	
P <sub>1</sub>	<pre>       14      / \     8   18    / \   5  11             </pre>	

1 mark correct order

6 (b) (ii) Reverse Inorder// Reverse order; (tree) traversal;

2

- 7 (a) student(jim);  
parent(rachel,jim);  
male(jim)  
female(rachel);  
**I** order  
Penalise case once only 3
- 7 (b) mother(X,Y) IF parent(X,Y) AND female(X)  
mark for IF and AND  
mark for parent(X,Y) and female(X)  
Penalise case once only 2
- 7 (c) grandfather(X,Y) IF father(X,Z) AND parent(Z,Y)//  
grandfather(X,Y) IF male(X) AND parent(X,Z) AND parent(Z,Y)  
1 mark for IF and AND  
1 mark for father(X,Z) or male(X) AND parent(X,Z)  
1 mark for parent(Z,Y)  
Penalise case once only 3