



General Certificate of Education

Computing 6510

**CPT 4 Processing and Programming
Techniques**

Mark Scheme

2008 examination - June 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

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.

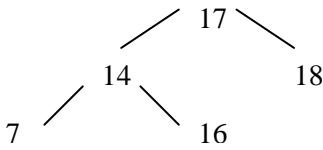
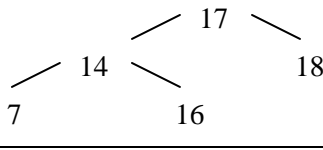
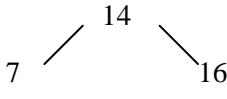
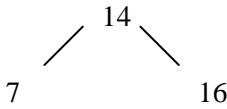
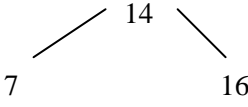
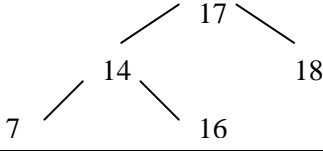
The following annotation is used in the mark scheme:

- ; - means a single mark
- // - means alternative response
- / - means an alternative word or sub-phrase
- A - means acceptable creditworthy answer
- R - means reject answer as not creditworthy
- I - means ignore.

Qu	Part	Sub Part	Marking Guidance	Mark
1	a		984;	1
1	b		984;	1
1	c	(i)	-13.0;; Allow method marks 1 mark for correctly identifying negative number 1 mark for integer value correct 1 mark for fraction (dependant on correct integer value)// 01101.000	3
1	c	(ii)	To maximise precision <u>in a given number of bits</u> // To minimise rounding errors; A to maximise accuracy <u>in a given number of bits</u>	1
1	c	(iii)	leftmost 2 digits/bits are different// a significant bit is stored after the binary point// bit after point different from bit before point; A the first bit after the sign bit is a '0'; A The second bit is a '0'; A an answer that <u>clearly</u> implies a '0' follows the '1'	1
1	c	(iv)	127// 2^7-1 ;; Max 1 for correct mantissa (01111111) or exponent (0111/7)	2
2	a	(i)	Empty entries waste memory // Maximum size// fixed size;	1
2		(ii)	Memory used by pointers//takes more time to <u>add</u> / <u>delete</u> nodes// indirect access takes more time; R programming difficulties	1
2	b		Place next item in first location/ location 0/ location 1// Implement a circular array/queue // allow wraparound;	1
2	c		IsFull/IsQueueFull;	1
3	a		Address of page in memory// <u>memory</u> frame <u>address</u> // <u>physical</u> <u>address</u> of page// <u>Base</u> <u>address</u> of page;	

			Page in memory or on disk// page in or out of memory// valid/invalid bit; Dirty/ modified Bit; Last time accessed;	Max 2
3	b		Page table is accessed using the page number as an index; Address found in page table; is added; to the offset;	Max 3
3	c		In cache memory// associative memory/storage// content-addressable memory/ storage// in main memory; Each executive process needs to be able to access its physical pages// needs to be accessed very quickly // Used every time memory is accessed;	Max 2
4	a	(i)	00010001 A 11//17	1
4		(ii)	00010101 A 15//21	1
4		(iii)	00001111 A 0F//15	1
4	b	(i)	Contents/offset; of <u>Index Register</u> ; added; to operand/ base address; A constant	4
		(ii)	Where a contiguous block of memory is to be referenced// Array/table manipulation/access; A <u>vectored</u> mechanism R relocation of code	1
5	a		a procedure/routine that calls itself/ is defined in terms of itself; A Function instead of procedure R re-entrant R program iteration (TO)	1

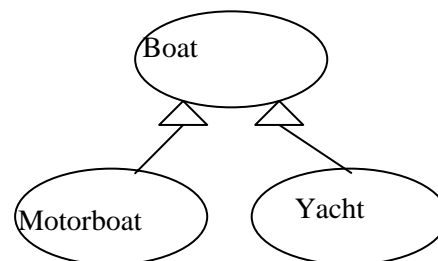
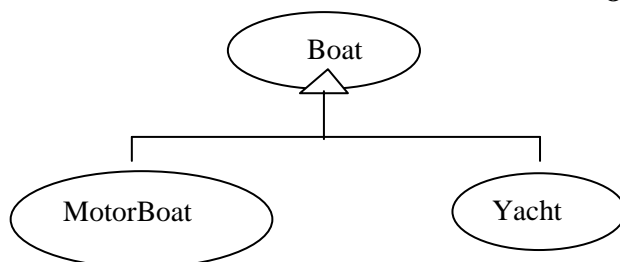
5	b	(i)		7
---	---	-----	--	---

Procedure Call	T
P ₁	
P ₂	18 ;
P ₁	
P ₃	 ;
P ₄	16 ;
P ₃	
P ₅	7 ;
P ₃	
P ₁	

Output	18;	17;	16	14	7;
--------	-----	-----	----	----	----

5		(ii)	Reversed Inorder; Tree <u>traversal</u> ; I Sort/ Re-arrange	2
6	a			2

OR



1 mark for all three classes in appropriate boxes;
1 mark for arrows in correct directions;

6	b		Insert a SetColour Procedure; A Function into the Public section; R make Colour Public	2
6	c		Yacht = Class/ subclass (<u>Boat</u>) (Public) Procedure SetBoatDetails (Override) Function GetMasts Function GetEngine <u>Private</u> Masts : Integer Engine : Boolean End A Procedure SetMasts and Procedure SetEngine/ AddNewYacht/SetYachtDetails instead of Procedure SetBoatDetails Masts and Engine must be private P1 if extra functions/ variables are included R any diagrams I any parameters to methods	1 1 1 1 1 1

			<p>OR</p> <p>Public class/subclass Yacht extends/inherits Boat</p> <pre>{ public void SetBoatDetails() public int GetMasts() public boolean/int GetEngine() private int Masts private boolean/int Engine }</pre> <p>A public void SetMasts and public void SetEngine// public void AddNewYacht/SetYachtDetails instead of public void SetBoatDetails</p>	<p>1</p> <p>1</p> <p>1</p> <p>1</p> <p>1</p> <p>1</p>
7	a	(i)	<p>Need to access/address registers/exact memory addresses/hardware directly;</p> <p>Fast speed of operation needed;</p> <p>Code needs to take up little <u>memory</u>// minimise the size of the executable code;</p> <p>A no compiler/interpreter exists yet for the machine// no other translator exists;</p> <p>R manipulate bits</p> <p>R comparison with machine code</p>	<p>Max 2</p>
7		(ii)	<p>Takes longer to program;</p> <p>Leads to more errors // more difficult to detect errors;</p> <p>Requires more skill;</p> <p>Difficult to understand;</p> <p>Difficult to maintain;</p> <p>Processor dependant// not portable// not problem oriented;</p>	<p>Max 1</p>
7	b		Give full marks for any correct method using instruction set provided.	6

	Opcode	Operand(s)	Comment
repeat	LD	pstatus;	Load first character into accumulator A LD pstatus+1
	AND;	#08;	Obtain ready bit A AND #0008
	CMP	#08;	Is printer ready? A CMP #0008
	JNE;	repeat;	Loop if not ready

8	a		region(norfolk, england); town(norwich,norfolk); Only penalise wrong case once	2
8	b		dorking, reigate 1 mark for each	2
8	c		in_same_country(Town1, Town2) IF town(Town1,Region1) AND region(Region1,Country) AND town(Town2,Region2) AND region(Region2,Country) 1 mark for IF 1 mark for town(Town1,Region1) 1 mark for AND region(Region1,Country) 1 mark for AND town(Town2,Region2) AND region (Region2,Country)	4
			OR in_same_country(Town1, Town2) IF town(Town1,Region1) AND region(Region1,Country1) AND town(Town2,Region2) AND region (Region2,Country2) AND Country1 = Country2 Only penalise wrong case once	4