

Computing

Advanced GCE

Unit **F453**: Advanced Computing Theory

Mark Scheme for June 2013

OCR (Oxford Cambridge and RSA) is a leading UK awarding body, providing a wide range of qualifications to meet the needs of candidates of all ages and abilities. OCR qualifications include AS/A Levels, Diplomas, GCSEs, Cambridge Nationals, Cambridge Technicals, Functional Skills, Key Skills, Entry Level qualifications, NVQs and vocational qualifications in areas such as IT, business, languages, teaching/training, administration and secretarial skills.

It is also responsible for developing new specifications to meet national requirements and the needs of students and teachers. OCR is a not-for-profit organisation; any surplus made is invested back into the establishment to help towards the development of qualifications and support, which keep pace with the changing needs of today's society.

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.

© OCR 2013

Annotations

| Annotation | Meaning |
|-------------------|--|
| ^ | Omission mark |
| BOD | Benefit of doubt |
| C | Subordinate clause/Consequential error |
| Cross | Cross |
| E | Expansion of a point |
| FT | Follow through |
| NAQ | Not answered question |
| NBOD | Benefit of doubt not given |
| P | Point being made |
| REP | Repeat |
| / | Slash |
| Tick | Tick |
| TV | Too vague |
| ZERO | Zero (big) |

| Question | | Answer | Marks | Guidance |
|----------|---------|--|-------|---|
| 1 | (a) | Two from: <ul style="list-style-type: none"> • Interrupt/Signal sent to processor • Request for processing time • Allows important tasks to be processed/to take precedence | 2 | Not message |
| | (b) | Two from: <ul style="list-style-type: none"> • Power (failure) • Peripheral/I/O interrupt • Clock interrupt • Software interrupt. | 2 | Not hardware unless explanation or example Not timer |
| | (c) | Two from: <ul style="list-style-type: none"> • To decide between interrupt & current task • To choose which interrupt to process if 2 (or more) occur together • To ensure most urgent task is performed first • To ensure most efficient use of processor. | 2 | |
| | (d) (i) | Two from: <ul style="list-style-type: none"> • A way of partitioning memory • Segments are not fixed size • Segments are logical divisions... • ...which hold complete sections of programs. | 2 | |
| | (ii) | <ul style="list-style-type: none"> • To allow programs to run when there is insufficient memory available. | 1 | Must be purpose Not instructions |
| | (iii) | Three from: <ul style="list-style-type: none"> • Use of backing store as if it were main memory/for temporary storage • Uses pages/fixed size units • Pages are moved from memory to backing store... • ...to make space (in memory) for the pages needed. | 3 | |

| Question | | Answer | Marks | Guidance |
|----------|---------|--|-------|--|
| | (iv) | <p>Two from:</p> <ul style="list-style-type: none"> • Disk thrashing/high rate of disk access • More time spent transferring pages than processing • Computer may 'hang'. | 2 | |
| 2 | (a) (i) | <ul style="list-style-type: none"> • Convert from source code\high level code\low level code... • ...to object code\intermediate\executable\machine. | 2 | |
| | (ii) | <p>Two from (max 2):</p> <p><i>Type:</i></p> <ul style="list-style-type: none"> • Compiler • Interpreter • Assembler. <p><i>Difference:</i> (<i>compiler & interpreter</i>)</p> <ul style="list-style-type: none"> • Compiler translates whole program as a unit (to intermediate language) • Compiler gives list of errors at end of compilation • Interpreter reports errors as they are found • Interpreter translates & runs 1 statement at a time. <p>(<i>compiler or interpreter, & assembler</i>)</p> <ul style="list-style-type: none"> • Compiler/interpreter uses high-level source code • Assembler uses low-level source code. • Assembler gives list of errors at end. | 4 | Max 2 for types, max 2 for difference |

| Question | Answer | Marks | Guidance |
|----------|---|-------|----------|
| (b) | <p><i>Code generation:</i></p> <ul style="list-style-type: none">• Occurs after syntax analysis/is last phase of compilation• Produces machine code program/executable code/intermediate code...• ...which is equivalent to the source program• Variables/constants are given addresses• Relative addresses are calculated. <p><i>Optimisation:</i></p> <ul style="list-style-type: none">• Makes code as efficient as possible• Increases processing speed• Number of instructions is reduced <p>Programmer can choose between speed & size.</p> | 6 | |

| Question | Answer | Marks | Guidance |
|----------|---|-------|--|
| 3 (a) | <p>MAR</p> <ul style="list-style-type: none"> • Receives address of instruction/data (from PC) • Holds address of data/instruction to be used • Receives operand part of instruction from CIR. <p>MDR</p> <ul style="list-style-type: none"> • Receives an instruction/data • ...from memory location in MAR • Receives data/instruction from... • ...memory location in address part of instruction/accumulator. | 6 | |
| (b) | <p><i>Common features:</i></p> <ul style="list-style-type: none"> • More than one processor... • ...working together... • ...to perform a single job • Job is split into tasks • Processors are controlled by a single operating system • Job is completed more quickly*. <p><i>Difference:</i></p> <ul style="list-style-type: none"> • Parallel processors can use any processor for a task • Co-processor performs specific types of task only... • ...e.g. floating point arithmetic... • ...so job is only completed more quickly if task is appropriate*. • Parallel processors need a more complex operating system/harder to program • Parallel processors are synchronised | 4 | <p>Allow only one point marked * unless explained Max 3 in a section</p> |
| 4 (a) | <ul style="list-style-type: none"> • X & Z • First two bits of mantissa are different. | 2 | |

| Question | | Answer | Marks | Guidance |
|----------|---------|--|-------|--|
| | (b) (i) | <ul style="list-style-type: none"> Exponent 0011 = 3 Mantissa 0.101, move point 3 places right becomes 0101. Value is +5. | 3 | Accept different methods of working |
| | (ii) | <ul style="list-style-type: none"> 1.5 in pure binary is 1.1 Mantissa is 01.10, move point 1 place to 0.110 Exponent is 0001. (Binary value is 0110 0001) | 3 | |
| | (iii) | <ul style="list-style-type: none"> +6 pure binary is 0110, so -6 is 1001+1 = 1010 Mantissa 1010. move point 3 places to 1.010 Exponent is 0011. (Binary value is 1010 0011) | 3 | |
| 5 | (a) (i) | <ul style="list-style-type: none"> Start at bluebell Compare each item in turn with tulip. | 2 | Accept simple diagram, but answer must relate to specific example in question. |
| | (ii) | Three from: <ul style="list-style-type: none"> Generally slower/Faster for binary (especially for large set of data)... ...as all data must be checked... ...until item found/or end of list is reached. Binary search halves the list each time | 3 | |
| | (b) (i) | <ul style="list-style-type: none"> Apple banana grape melon pear pineapple raspberry. | 1 | |

| Question | Answer | Marks | Guidance |
|----------|---|-------|---|
| | <p>(ii) <i>Eg</i> open existing files create new file check existing files are not empty use pointers/counters to identify records for comparison repeat compare records indicated by pointers if records are different then copy earlier value record to new file move correct pointer else copy one record only to new file move both pointers endif until end of one file copy remaining records from other file close files Mark points</p> <ul style="list-style-type: none"> • opening and closing files • use of pointer(s) for comparison • compare data and copy to new file • move pointer • correctly handling the duplicate • correct condition to end loop • copy rest of remaining file. • Check files contain data. <p><i>Assumption</i></p> <ul style="list-style-type: none"> • Assume common key • Removal of duplicates/no removal of duplicates • Assume same sort order/common sort. <p>(Max 5 for mark points and max 1 for assumption)</p> | 6 | <p>Accept equivalent while loop with correct condition in place of repeat ...until</p> <p>Must open and close files</p> |

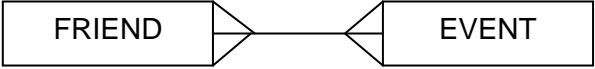
| Question | Answer | Marks | Guidance |
|----------|---|-------|--|
| 6 (a) | <p>Mark band 6–8. High level response.</p> <p>Candidate has explained all 4 terms and used relevant examples. Candidate has used appropriate technical terminology throughout. There are few, if any, spelling errors or grammatical errors.</p> <p>Mark band 3–5. Medium level response.</p> <p>Candidate has explained at least 3 of the terms, or explained at least 2 of the terms with relevant examples. Candidate has used some technical terminology in the response. There may be spelling errors or grammatical errors, but they are not obtrusive.</p> <p>Mark band 0–2. Low level response.</p> <p>Candidate has listed some relevant points but failed to describe the terms in any detail. There is a lack of cohesion in the response. Candidate has failed to use correct technical terms in the response. Spelling and grammatical errors affect the readability of the response.</p> <p><i>Points may include:</i></p> | 8 | Examples given must be from the garden centre diagram provided. |

| Question | | Answer | Marks | Guidance |
|------------|-------------|---|-------|----------|
| | | <p><i>Class:</i></p> <ul style="list-style-type: none"> • A template for a set of objects that have state & behaviour • They have methods & attributes. • represented by the rectangle on the diagram • eg Product has Price & setPrice(). <p><i>Derived class:</i></p> <ul style="list-style-type: none"> • A class that has all the attributes & operations of its superclass & may also have attributes & operations of its own • eg Plant has Price & setPrice() from Product, & also has Variety & findVariety(). <p><i>Inheritance:</i></p> <ul style="list-style-type: none"> • A derived class inherits all the attributes & operations from its superclass • shown by the arrow on the diagram • eg Plant inherits ProductCode from Product. <p><i>Encapsulation:</i></p> <ul style="list-style-type: none"> • Data hiding • Data can only be accessed using operations/methods defined for the class • Maintains data integrity • eg Price can only be changed using setPrice(). | | |
| (b) | (i) | <ul style="list-style-type: none"> • Declarative. | 1 | Cao |
| | (ii) | <ul style="list-style-type: none"> • eats (P,Q) if mouse (P) and food (Q). | 1 | Cao |

| Question | | Answer | Marks | Guidance |
|----------|---------|--|-------|----------|
| | (iii) | <p>Five from:</p> <ul style="list-style-type: none"> • Attempt to solve mouse (X) • Finds X=fluffy • Set X=fluffy • Attempt to solve food (Y) • Finds Y=cheese • Set Y=cheese • A solution is X=fluffy, Y=cheese. | 5 | Cao |
| | (iv) | <ul style="list-style-type: none"> • Backtracking. | 1 | cao |
| | (v) | <ul style="list-style-type: none"> • food (biscuits) is not a fact. | 1 | |
| 7 | (a) | <ul style="list-style-type: none"> • (Binary) tree. | 1 | |
| | | <p>Answer $pq+rs^*$- Alternative answer is $pqrs^*+$ Marks for:</p> <ul style="list-style-type: none"> • Starts with pq • Includes rs^* • Complete answer. | 3 | |
| | (b) | <ul style="list-style-type: none"> • Stack. | 1 | |
| | | <p>Answer 13 Marks for:</p> <ul style="list-style-type: none"> • $(w-x)*y+z$ (or equivalent with values) • Demonstration/explanation • 13. | 3 | |
| | (c) (i) | <p>$(a+b)*c$</p> <ul style="list-style-type: none"> • $ab+$ • $ab+c^*$ OR $cab+^*$ | 4 | |

| Question | | | Answer | Marks | Guidance |
|----------|--|------|--|-------|----------|
| | | | $a+b*c$ <ul style="list-style-type: none">• bc^*• $abc^*+ \text{ OR } bc^*a+$ | | |
| | | (ii) | One from: <ul style="list-style-type: none">• Avoid ambiguity• No need for brackets. | 1 | |

| Question | | Answer | Marks | Guidance |
|----------|---------|---|-------|---|
| 8 | (a) | <p>Three from:</p> <ul style="list-style-type: none"> • Provide fast access to data... • ...for specific purposes/used during fetch-execute cycle • ...when frequent access is needed • Used for temporary storage. | 3 | |
| | (b) (i) | <p>Two from:</p> <ul style="list-style-type: none"> • Temporary storage in ALU • Holds data being processed/used during calculations • Deals with input & output in the processor. | 2 | |
| | (ii) | <ul style="list-style-type: none"> • Before, holds 123... • ...as this is address of next instruction • During, changes to 124... • ...as PC is incremented • Then changes to 46... • ...as this is the operand. | 6 | Accept 125 in place of 124 if explained |
| 9 | (a) (i) | <ul style="list-style-type: none"> • One-many. | 1 | cao |
| | (ii) | <p><i>Primary key:</i></p> <ul style="list-style-type: none"> • Unique identifier ... • e.g. FriendNo in Friend • In Ticket, may use composite key... • ...as Ticket is a link entity <p><i>Foreign key:</i></p> <ul style="list-style-type: none"> • Primary key from Friend is an attribute in Ticket... • ...to link the tables/represent relationship • ...foreign key is at “many” end of the relationship. | 4 | <p>Max 2 for each type of key Must use example in answer</p> <p>Example for foreign key must be FriendId in Ticket.</p> |

| Question | | Answer | Marks | Guidance |
|----------|---------|--|-------|---|
| | (b) |  | 1 | |
| | | <ul style="list-style-type: none"> • It creates one to many relationships/acts as a link table • It makes it in 3NF/normalised • Reduces data inconsistencies | 3 | <ul style="list-style-type: none"> • Many-many relationship... • ...is not allowed in 3NF/is not normalised • ...causes duplication of data/causes errors. |
| | (c) (i) | <p>Three from:</p> <ul style="list-style-type: none"> • To write schema • Describes data items to be stored... • ...and the relationships between them • Create tables • Define primary/foreign keys • Define validation rules. | 3 | |
| | (ii) | <ul style="list-style-type: none"> • To access/query/sort/search data • To store/insert data • To change/update/delete data. | 3 | |

| Question | Answer | Marks | Guidance |
|----------|---|------------|----------|
| (d) | <p><i>Explanation:</i></p> <ul style="list-style-type: none"> • A file containing descriptions of data in database... • ...stored in tables/as a relational database • Uses metadata • Used by database managers • Used when altering database structure. <p>(Max 3)</p> <p><i>Examples:</i></p> <ul style="list-style-type: none"> • Names of tables/columns/field • Data characteristics • Relationships between data • Access rights (people/programs) • Identifies primary keys... • ...& foreign keys. <p>(Max 2)</p> | 5 | |
| | Total | 120 | |

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

© OCR 2013

