UNIVERSITY OF CAMBRIDGE INTERNATIONAL EXAMINATIONS

GCE Advanced Level

MARK SCHEME for the June 2005 question paper

9691 COMPUTING

9691/03   Written Paper, maximum mark 90

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 Examiners were initially instructed to award marks. They do not indicate the details of the discussions that took place at an Examiners' meeting before marking began. Any substantial changes to the mark scheme that arose from these discussions will be recorded in the published *Report on the Examination*.

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 must be read in conjunction with the question papers and the *Report on the Examination*.

- CIE will not enter into discussion or correspondence in connection with these mark schemes.

CIE is publishing the mark schemes for the June 2005 question papers for most IGCSE and GCE Advanced Level and Advanced Subsidiary Level syllabuses and some Ordinary Level syllabuses.

Grade thresholds for Syllabus 9691 (Computing) in the June 2005 examination.

	maximum mark available	minimum mark required for grade:		
		A	B	E
Component 3	90	60	54	30

The thresholds (minimum marks) for Grades C and D are normally set by dividing the mark range between the B and the E thresholds into three. For example, if the difference between the B and the E threshold is 24 marks, the C threshold is set 8 marks below the B threshold and the D threshold is set another 8 marks down. If dividing the interval by three results in a fraction of a mark, then the threshold is normally rounded down.

June 2005

Advanced Level

MARK SCHEME

MAXIMUM MARK: 90

SYLLABUS/COMPONENT: 9691/03

Computing
Written Paper 3

UNIVERSITY *of* CAMBRIDGE
International Examinations

1   TO   -Contains only characters represented in ASCII/on keyboard
RT   -allows addition of fonts/colours/bold…
Advs -Smaller file size/faster transmission
       -More likely to be compatible with other user's software/more
       likely to be readable at destination.   **(4)**

2   -Address of instruction in PC…
-is copied to MAR
-PC is incremented
-Contents of address in MAR…
-copied to MDR
-(Contents of MDR) copied to CIR
-Decode instruction in CIR
-Load address (300) in CIR into PC
(1 per -, max 7)   **(7)**

3   **(a)**   -Each value, in turn, from the left, starting with second value…
-inserted in to list to the left in correct place
-by moving appropriate numbers to the right to create space
-1st pass   7 10
-2nd pass   5 7 10
-3rd pass   5 6 7 10
-4th pass   5 6 7 9 10 (Only if IS attempted)
(1 per -, max 4)   **(4)**

(b)   -1. First value in each list is compared
-2. Smallest written to new list
-3. Next value read from list which has had value written to new
-Repeat steps 2 and 3, with comparison, until one list is empty.
-Copy rest of the remaining list to the new list.
-E.g. 2 and 5 compared
-2 written to new
-3 read from list A
-3 and 5 compared
  - - - - - - - - -
-10 written to new
-remainder of A (11 and 17) written to new
(1 per -, max 6)   **(6)**

4 **(i)** -Change from old to new system immediately
-Problems of training of staff on new system…
-Lower administration costs
-preparation of database
-No fall back if system does not work
-Immediate changeover good for morale

(ii) -One part of new system changed…
-for example customer file is changed
-Allows staff to get used to one part before trying the next
-Problem because the two systems probably not compatible.
-Can repair faulty module without affecting others

(iii) -Whole system covering one area is changed…
-e.g. the sports sales/the Lancaster warehouse
-Nothing else is changed until sure new system is working correctly
-Spreads cost of installation/training
-Benefits/training of staff of parallel running without the costs
-When spread to other parts of the company, trained staff can act as mentors
-May not be possible to isolate one area of the business.
-Is not using full data.
(1 per -, max 4 per dotty, max 9) **(9)**

5 **(a) (i)** -Restricted to the organization
-Limited number of users/pages
-(Access controlled by) use of passwords
-Allows for confidential/sensitive data
(1 per -, max 2) **(2)**

(ii) -Enhancement of text by use of colour/bold/font…
-second e.g. colour/bold/font…
-by enclosing text in tags.
-Use of blank lines…
-to format text…
-by using special tags.
-Use of head and body…
-to convey information to search programs…
-provide titles to work.
-Hot buttons
-to allow simple searching
-jumps to different headers
-Frames
-to allow insertion of tables/diagrams
-to enhance understanding of document
-Style sheets
-to define formats to be used on whole areas
NB Accept also specific commands such as GET/POST
(1 per -, max 3 methods, max 2 per method, max 6) **(6)**

(b) (i) -Connects different types of network/parts of network
-Use addresses to create routes between networks
-Must keep tables of information about addresses.

(ii) -Able to learn layout of network to route data efficiently
-Access available to all areas of all networks
-Produces a single logical network by connecting a number of networks.

(iii) -Convert A to D/D to A/audio to digital/phone signal to digital
-to allow communication via telephone line
-connects widely dispersed parts of company/company to WAN
(1 per -, max 2 per dotty, max 6) **(6)**

(c) (i) -Users are unaware of the hardware and software…
-they believe they are the sole users of a standalone
-Users are unaware of communications

(ii) -Allow users to manage access to their own files
-while maintaining lack of access to others.
-Maintain directory of software/services
-available to specified users.

(iii) -Security of files by restricting access
-Maintain file of users and their log ins…
-and their rights.
(1 per -, max 2 per dotty, max 6) **(6)**

6 **(a)** -Interpreter translates line of code and then runs it
-Compiler translates entire program before run.
-Compiler creates an object code
-Interpreter retains source code.
-Compiler must be present for translation
-Interpreter must be present for run
(1 per -, max 2, must include one for each of interpreter and compiler) **(2)**

(b) -Creates a stream of tokens…
-each group of characters is replaced by a token
-Symbol table created…
-accessed by hashing algorithm
-which initially stores just the variable names
-Redundant characters removed…
-white spaces/tabs/comments…
-(Some) error diagnostics created…
-e.g. illegal variable names.
(1 per -, max 5) **(5)**

(c) -Address of variables calculated
-and stored in symbol table
-Intermediate code produced…
-which can then be turned into executable code/machine code
-Code optimized…
-which involves using rules to make code as small/efficient as possible.
(1 per -, max 2 pairs, max 4) **(4)**

7 **(a) (i)** 01001110 (1 per nibble) **(2)**

(ii) 4E (1 per digit) **(2)**

(iii) 01111000 (1 per nibble) **(2)**

(b) -Bits arranged in threes from the right
-Need to add leading zero
- 001 001 110
-Each group of three bits converted (to denary/octal)
(1 per -, max3) **(3)**

(c) (i) 11000001
10100010 **(2)**

(ii) 1,01100011
(1 for answer, 1 for indication of overflow, allow ft) **(2)**

(iii) -Overflow
-Answer is positive…
-because of overflow from +ve bits into –ve bit
-Processor recognizes error because carry in to MSB is different from carry out.
(1 per-, max 2) **(2)**

8. a) (i) -A particular variable is assigned a value
           -e.g. if male(X) then dis is an instance of X/X is instantiated to dis

     (ii) -The intention to find all instances that satisfy a rule/set of facts
           -e.g. If rule is male(X) then the goal is to find dis and david and john

     (iii) -If the result of one rule does not apply in a second rule, then go back
           to find another result of the first rule
           -e.g. parent (john,dis) is found if we are searching for mother of dis.
           This fails the second part of the rule for mother because john is male,
           so backtracking is used to return to the next example satisfying the
           first part of the rule.
           (1 per -)   (6)

   b) (i) grandmother (x,y) :- grandparent (x,y), female (x)
           (1 mark for grandparent (x,y) 1 mark for female(x))   (2)

     (ii) -Ignores parent (john,dis) parent (john,may) parent (dis,sto) because
           Y<> david
           -finds parent (dis,david)
           -searches for parent (X,dis), finds X = john
           -finds male (john), rejects X = john because not female
           -backtracks to find next occurrence of (X,dis)
           -finds parent (minah,dis)
           -finds female (minah), reports minah is grandmother
           (1 per -, max 2)   (2)

9.   -Some simulations are time sensitive
           -and require large amounts of processing
           -where processes are interrelated
           -calculations can be done at same time which speeds up processing
           -e.g. weather forecasting
           (1 per -, max 4)   (4)