UNIVERSITY OF CAMBRIDGE INTERNATIONAL EXAMINATIONS

GCE Advanced Level

MARK SCHEME for the October/November 2011 question paper for the guidance of teachers

9691 COMPUTING

9691/33 Paper 3 (Written Paper), maximum raw mark 90

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

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

- Cambridge will not enter into discussions or correspondence in connection with these mark schemes.

Cambridge is publishing the mark schemes for the October/November 2011 question papers for most IGCSE, GCE Advanced Level and Advanced Subsidiary Level syllabuses and some Ordinary Level syllabuses.

1 **(a)** -A short sequence of machine code instructions
-Stored in the ROM
-available when the power is switched on
-Used to load the operating system
(1 per -, max 2) [2]

(b) -Power-up process places address of first instruction in PC
-Carries out the POST (power on self-test)
-Uses the user-defined parameters in the boot file to configure the system
-Mention of BIOS/autoexec.bat/config.sys…
-First instruction of OS is loaded into memory and address passed to PC
(1 per -, max 4) [4]

2 **(a)** -during the fetch stage the contents of the address in the MAR is copied into the MDR …
-The contents of MDR is the instruction to be used (this is copied to the CIR)
-during the execute stage (store instruction) the contents of the ACC is copied into the MDR
-during the execute stage (load/add instruction) the contents of the memory location is copied into the MDR
(1 per -, max 3) [3]

(b) -**Data bus** carries contents of a memory location/contents of a register/a data value/an address/an instruction

-**Address bus** carries an address of a memory location/device
-the address bus carries an address from the processor to main memory/a device

-**Control bus**
- Separate wires each used to carry a control signal
- the bus carries control signals to the various components
- by example e.g. read operation completed // interrupt

-Data bus is bi-directional // data bus used to read/write data // Address bus is uni-directional // the control bus is bi-directional

(mark as 3 × 2 per bus, max 6) [6]

3 (a) (i) +39 = 00100111

(ii) –47 = 10101111
(1 for both sign bits and 1 for each magnitude part) [2]

(b) (i) –3 = 1111 1101 [2]

(ii) –47 = 1101 0001 [2]

(in each case 1 mark per nybble)

(c) (i) = (1/8 + 1/32) * 2^4
= 5/32 *16
= 2½
OR:
= 0.00101 * 2^4
Hence move point 4 places
= 10.1 = 2 1/2
(1 mark for each underlined section, max 2. Note: Accept decimal values) [2]

(ii) 010100 0010
(1 for mantissa, 1 for exponent) [2]

(iii) M = ½ + ¼ + 1/32 OR = 25/32
E = 3
Number is 25/32 * 8
= 6 ¼
(1 per line, max 3) [3]

4 (a) -networked communication system.../// content provided by a web server
-probably provided on the Internet
-Restricted access...
-to specific members authorised by the health system
-Access is password controlled
-Content viewed using browser software
(1 per -, max 3) [3]

(b) Advantages
-Limited number of users speeds up access...
-Information being communicated is sensitive/confidential...
-needs protection from being seen by unauthorised people
-Information on system will be relevant/easily updated
-Less information makes it easier to navigate
-easier to control who can access the content
Disadvantages
-may involve additional set-up costs // need to set up a LAN
-extra administration // setting up users (& passwords)/access rights
(1 per -, max 5) [5]

5 **(a)** -Cost of creating the real thing == different braking units would need to be built
-Time that would be needed to create the real thing == the parameters of the sim braking system can be changed immediately
-Time taken to run the tests == test time can be greatly reduced
-a wide variety of conditions need to be tested == e.g. it may be necessary to drive for 100000 miles/at different speeds, this could be simulated
-Ability to change conditions immediately == e.g. not necessary to transport car to different parts of the world
-extreme case scenarios can be tested == conditions may never occur during real life testing
(3 x 2 per advantage) [6]

(b) -**Speed**
-in order to simulate stopping distances
-**Weight** of car and load…
-in order to simulate the effects of inertia with different loads
-Materials used/size of structure used/methods of fixing to car
-to try to reduce final cost while still remaining efficient
-**driving style**
-gentle braking/hard braking/cornering/reaction time
-**tyres**
-wear/type of tread/tyre material
-**road surface**
-roughness/material
-**weather conditions (temperature, wind, precipitation)**
-in order to replicate different climates…
(1 per -, max 5) [5]

6 **(a)** -Reduces repetition/duplication of data items // keeps physical volume of data to a minimum // minimises redundant data
-Increases data integrity // reduces data inconsistency
-Simpler data retrieval through queries // reports are easy to generate
-Amending/searching/sorting data is easier
-Amending the data structures is simpler to implement
-Changes to the data structure will not affect existing applications programs // Program/data independence
(1 per -, max 3) [3]

(b) **(i)** -GuestID or similar
-unique [2]

(ii) -e.g. Type of charge (bar/restaurant/laundry/…)
-so that items can be accessed according to a different criteria other than by primary key [2]

(iii) -Attribute/field in one table which links to the primary key in another table [1]

(iv) -GuestID…
-to link each account to the relevant guest // to link ACCOUNT and GUEST tables [2]

7 **(a)** -during lexical analysis …
-identifiers and keywords are differentiated
-checked against rules (e.g. length) for identifiers
-Error messages produced if identifier does not match the expected rules
-variable identifiers will be tokenised
-variable identifiers entered into symbol table
-Data type will be added to the entry in the symbol table
-addresses in memory allocated to variables
-during syntax/semantic analysis stage ….
-assignment of illegal types of data to variables is reported
(1 per -, max 7) [7]

(b) **(i)** -object code is difficult to interfere with
-object code runs faster than interpreted source code
-compiler can optimise executable code
-the code is not translated each time the program is run
-Compiler does not need to be present when the program is run
-Compiled code will be free from syntax errors
(1 per -, max 2) [2]

(ii) -Errors are more easily located…
-reports errors when source code is present...
-stopping at the point of the error
-Parts (only) of program can be tested/testing can be started before all the program is written
-errors when found can be immediately corrected.
(1 per -, max 2) [2]

8 Paging
-memory is divided into equal-sized units called page frames
-program/data file is divided into equal-size units called pages
-one or more pages may be loaded into memory at any one time
-Pages may be discontiguous
-Pages swapped in and out as required
-pages not in main memory are stored in virtual memory/backing store
-page table/Index of pages/processes kept
-absolute address is calculated by adding page address to relative address in instruction
-paging is transparent to the programmer

Segmentation
-Memory is divided into variable length blocks
-Programs can consist of many segments
-Segments normally match natural divide in jobs/logical blocks
-Index of segments stored which must…
-store base address and length of segment
-programmer will organise code modules into segments
(1 per -, max 4 per dotty, max 6) [6]

9 (a) (i) -describes what is to be accomplished
-not how (no algorithm written)
-the user states what is to be found/set a goal
-Consists of a set of facts and rules
-Rules are applied to the data until the goal is reached
-Mention of backtracking/instantiation
(max 1)

(ii) -Program describes how to solve the problem in a sequence of steps/algorithm
-lends itself to top-down design/modularisation
-using procedures/functions
(max 1) [2]

(b) (i) -A class is the "blueprint" from which objects are defined // a class consists of the properties and methods that define each object
-Plant/Tree/Bulb is a class [2]

(ii) -One class can use the properties and methods from a parent/base/super class
-Tree/Bulb inherits the properties and methods of Plant [2]

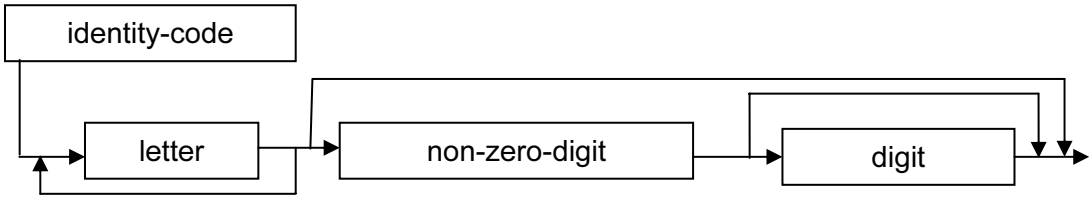
(iii) -An object can only read/write a property value using methods of the class // Class contains both properties and the methods to use it
-e.g. The CountryOfOrigin property can only be output using the getCountryOfOrigin method in the class Tree [2]

10 (a) (i) Must begin with at least one <letter>

(ii) X is not defined as a <letter>

(iii) A maximum of 2 digits is allowed at the end [3]

(b)



Mark Points:
-Only one entry and one exit point used
-Order correct (letter, non-zero-digit, digit)
-Loop around **letter**
-alternative path to omit number
-alternative path to omit 2nd digit
(1 per -, max 4) [4]