UNIVERSITY OF CAMBRIDGE INTERNATIONAL EXAMINATIONS

GCE Advanced Level

MARK SCHEME for the May/June 2011 question paper

for the guidance of teachers

9691 COMPUTING

9691/32 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 May/June 2011 question papers for most IGCSE, GCE Advanced Level and Advanced Subsidiary Level syllabuses and some Ordinary Level syllabuses.

1 e.g.-Data bus
-to carry data from one location to another in processor // e.g. from MDR to CIR

-Address bus
-carries the address of a memory location // e.g. Address of location in memory from MAR

-Control bus
-Carries control signals around processor // to synchronise the operation of the processor components // by example: memory read/write completed // each line carries a different signal.
Accept: system bus, memory bus, firewire, USB, PCI + explanation
(2nd mark is dependent on correct bus name)
(2 per -, max 6) [6]

2 **(a)** -One to one
-Mnemonics are used to represent operation codes
-Labels are used to represent memory addresses
-machine code is binary codes (only)
-assembly code can not be executed // machine code can be executed
-machine code and assembly language are both low level languages (machine specific)
(1 per -, max 2) [2]

(b) -Labels added to a symbol table
-Labels are later looked up to determine the actual address / Assembler must allocate addresses to labels
-Mnemonic looked up in opcode table to find operation code
-Macro instructions used to stand for groups of instructions
(1 per -, max 2) [2]

(c) **(i)** -Address in instruction is the address of the address of / pointer to the location...
-which contains the data to be used [2]

(ii) -Address in the instruction has added to it
-the contents of the Index Register/IR [2]

(iii) -Address in the Instruction is the displacement
-from the address of the first/current instruction
-the value is added to the PC [2]

3 -Coaxial cable
-description/one transmission medium (copper) surrounded by insulation

-Twisted pair
-description/two conducting wires twisted around each other

-Optic fibre
-many fibres contained
-description/fine glass strands carry light signals // optic fibre is very fragile
-Interference free

-Wireless communication
-Radio signals
-open to interception / latency / uses WEP keys for security

-Infrared/Microwave
-restricted by line of sight
-transfer rate statement
-range statement

(1 per -, max 8) [8]

4 **(a)** -A system in which the output is produced quickly enough to affect the next input /current process
-a system that reacts fast enough to influence behaviour in the outside world [1]

(b) -A number of sensors stationed around apartment
-Temperature/humidity sensor sends temperature/humidity to processor
-Use analogue to digital converter to convert the temperature/humidity measurements
-Processor decides whether air conditioning system is in operation
-(processor) compares measured temperature/humidity to required temperature/humidity
-If necessary actuator is used to adjust settings/turn on cooling/heating/humidifier
-Delay before next reading is taken from temperature/humidity sensor. // temperature readings are sampled // taking readings is repeated

-Sensors on windows to warn if they are open during operation.
(1 per -, max 4) [4]

(c) Any suitable real-time or pseudo-real-time application e.g.
-To play a racing game
-So that the player can steer the car realistically
-any reservation type system
-to prevent double booking [2]

5 **(a)** -Temporarily storing data for output later
-several computers can send data to be printed at the same time
-when queuing jobs sent to a single device
(1 per -, max 2) [2]

(b) **(i)** -Jobs can be queued to ensure that none are missed
-Stops jobs being frozen/lost when printer unavailable
-complete documents are printed

(ii) -print jobs are stored on secondary storage
-jobs can be given a print priority
-jobs are maintained by a queue / priority queue data structure
-data structure consists of reference data to each print job
-When printer free, job with highest priority / at head of queue is printed
-print files are sent from secondary storage to print buffer.
(1 per -, max 4 per dotty, max 5) [5]

6 **(i)** -language to describe/alter table designs (NOT file)
-includes Identifiers/data type/relationships
-any validation rules that the data must adhere to…
(1 per -, max 2) [2]

(ii) -designed to allow a user to query/retrieve data/sort the database
-insert / delete / update
-data in the database / table(s)
(1 per -, max 3) [3]

7 **(i)** –106 [1]

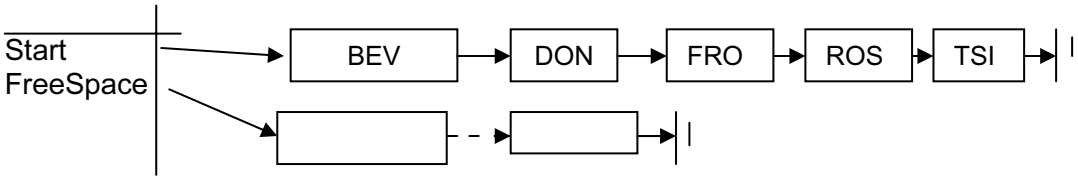
(ii) –22 [1]

(iii) 96 (1 per digit) [2]

8 (a) (i) -A dynamic data structure changes size // A static data structure has the same
-dynamic data structure matches size to data requirements // static data structu
no account of data requirements
-dynamic data structure takes memory from heap as required
-static data structure is predefined at compile time [2]

(ii) *Advantage:*
-Array is of fixed size which simplifies algorithms // or by example e.g. retrieval of data
-Array controls the maximum size of the queue
Disadvantage:
-Queue held in an array cannot expand beyond the size of the array
-If queue is small then memory space is wasted. [2]

(b) (i) Either:



Start: 5 NFree: 6

1	FRO	4
2	TSI	X
3	DON	1
4	ROS	2
5	BEV	3
6		

Start: 4 NFree: 5

0	FRO	3
1	TSI	X
2	DON	0
3	ROS	1
4	BEV	2
5		

Mark as follows ...
-Start pointer + some value/arrows
-All values included
-Null pointer
-Indication of free space

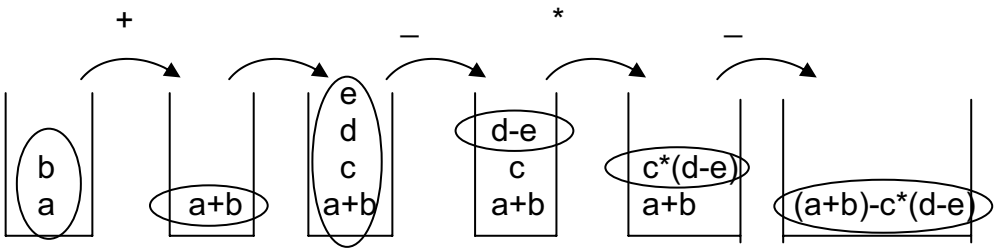
Diagram in arrival order
-in arrival order
-with correct pointers
OR
Diagram in alphabetical order
-with correct pointers
OR
Array diagram
-in arrival order
-correct pointers
(1 per -, max 5) [5]

(ii) -Input NewItem
-Store NewItem in next free space
-Set Current to value at Start
-Read values in list following pointers.
-until Current value in list > NewItem
-Pointer of Previous points to NewItem
-NewItem points to Current
-update free space list
-Mention of any special cases e.g. NewItem being First in list // list empty // list full // no free space

(1 per -, max 5) [5]

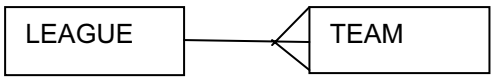
9 **(a)** -reverse Polish expressions can be processed directly from left to right
-Is free of ambiguities
-does not require brackets
-does not require use of rules of precedence
(1 per -, max 2) [2]

(b)



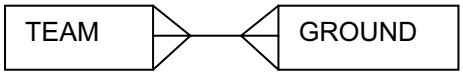
Mark points:
-at least two operators shown between transitions
-a and b in first stage
-a+b after first operator
-e,d,c, (a+b) in stack in correct order
-(d-e)
-c*(d-e)
-(a+b)-c*(d-e)
(1 per -, max 6) [6]

10 **(i)** One to many



[2]

(ii) Many to many



[2]

(iii) -Link table needed...
-with primary key made up of combination of primary keys of TEAM and GROUND
-Primary keys of TEAM and GROUND used as foreign keys in link table
-This turns the many to many relationship into..// a many-to-many relationship can not be implemented
-One-to-many and many-to-one/ 2x one-to-many relationships
(1 per -, max 4) [4]

11 (a) -Interpreter translates one instruction, runs it before going on to the next // translates all the instructions before run.
-Compiler creates object code/executable file // Interpreter does not
-Interpreter makes for easier debugging
-Compiled programs will execute faster // interpreted code will execute slower
-Interpreter must be present to run the program // compiler not needed at runtime
-Interpreter will translate code in loops more than once // Compiler only once
-once compiled no further translation needed // every program execution requires interpreter
(1 per -, max 3) [3]

(b) (i) -Contents copied from PC
-Contents changed to the operand/address part of CIR [2]

(ii) -Instruction copied <u>from memory/location</u> to MDR when contents of MAR are from PC
-Data copied <u>from memory/location</u> to MDR when instruction is LOAD
-Data copied <u>from ALU/Accumulator</u> to MDR when instruction is STORE [max 2]

12 -Must safeguard against unauthorised access to the computer system
-Firewall used to restrict access to known sources
-Control access to the network using accounts/user IDs with passwords // procedures in place for authentication
-File contents can be encrypted
-procedures in place to protect against malware

-all payments/communication can be made through a secure connection
-need to safeguard against bogus websites

-Procedures in place for authorisation of resources
-Users allocated access rights to various resources // users have access to certain files/folders only
-Files can be password protected / read-only
-users can access the network from certain terminals only / certain times of the day only
-use of digital signatures
(1 per -, max 6) [6]