MARKSCHEME

November 2005

COMPUTER SCIENCE

Standard Level

Paper 1

14 pages

If you do not have a copy of the current Computer Science Guide, please request one from IBCA.

General Marking Instructions

After marking a sufficient number of scripts to become familiar with the markscheme and candidates' responses to all or the majority of questions, Assistant Examiners (AEs) will be contacted by their Team Leader (TL) by telephone. The purpose of this contact is to discuss the standard of marking, the interpretation of the markscheme and any difficulties with particular questions. It may be necessary to review your initial marking after contacting your TL. **DO NOT BEGIN THE FINAL MARKING OF YOUR SCRIPTS IN RED INK UNTIL YOU RECEIVE NOTIFICATION THAT THE MARKSCHEME IS FINALIZED.** *You will be informed by e-mail, fax or post of modifications to the markscheme and should receive these about one week after the date of the examination. If you have not received them within 10 days you should contact your Team Leader by telephone. Make an allowance for any difference in time zone before calling.* **AEs WHO DO NOT COMPLY WITH THESE INSTRUCTIONS MAY NOT BE INVITED TO MARK IN FUTURE SESSIONS.**

You should contact the TL whose name appears on your "Allocation of Schools listing" sheet.

Note:
Please use a personal courier service when sending sample materials to TLs unless postal services can be guaranteed. Record the costs on your examiner claim form.

General Marking Instructions

1. Follow the markscheme provided, do **not** use decimals or fractions and mark only in **RED**.

2. Where a mark is awarded, a tick (✓) should be placed in the text at the **precise point** where it becomes clear that the candidate deserves the mark.

3. Sometimes, careful consideration is required to decide whether or not to award a mark. Indeed, another examiner may have arrived at the opposite decision. In these cases write a brief annotation in the **left hand margin** to explain your decision. You are encouraged to write comments where it helps clarity, especially for moderation and re-marking.

4. Unexplained symbols or personal codes/notations on their own are unacceptable.

5. Record subtotals (where applicable) in the right-hand margin against the part of the answer to which they refer. Show a mark for each part question (a), (b), *etc.* Do **not** circle sub-totals. Circle the total mark for the question in the right-hand margin opposite the last line of the answer.

6. Where an answer to a part question is worth no marks, put a zero in the right-hand margin.

7. **Section A**: Add together the total for each question and write it in the Examiner Column on the cover sheet.
Section B: Record the mark awarded for each of the three questions answered in the Examiner Column on the cover sheet.
Total: Add up the marks awarded and enter this in the box marked TOTAL in the Examiner Column on the cover sheet.

8. After entering the marks on the cover sheet check your addition of all marks to ensure that you have not made an arithmetical error. Check also that you have transferred the marks correctly to the cover sheet. **We have script checking and a note of all clerical errors may be given in feedback to all examiners.**

9. Every page and every question must have an indication that you have marked it. Do this by **writing your initials** on each page where you have made no other mark.

10. If a candidate has attempted more than the prescribed number of questions, mark only the required number of answers in the order in which they are presented in the script and ignore any excess material, regardless of its quality. Make a comment to this effect in the left hand margin. **This is unless the candidate indicates, otherwise on the front cover.**

11. A candidate can be penalized if he/she clearly contradicts him/herself within an answer. Once again make a comment to this effect in the left hand margin.

Subject Details: Computer Science SL Paper 1 Markscheme

Mark Allocation

Section A: Candidates are required to answer ALL questions. Total 30 marks.
Section B: Candidates are required to answer any three questions (10 marks each). Total 30 marks.
Maximum total = 60 marks.

General

A markscheme often has more specific points worthy of a mark than the total allows. This is intentional. Do not award more than the maximum marks allowed for part of a question.

When deciding upon alternative answers by candidates to those given in the markscheme, consider the following points:

- Each marking point has a separate line and the end is signified by means of a semi-colon (;)
- An alternative answer or wording is indicated in the markscheme by a "/"; either wording can be accepted.
- Words in (…) in the markscheme are not necessary to gain the mark.
- The order of points does not have to be as written (unless stated otherwise).
- If the candidate's answer has the same "meaning" or can be clearly interpreted as being the same as that in the mark scheme then award the mark.
- Mark positively. Give candidates credit for what they have achieved, and for what they have got correct, rather than penalising them for what they have not achieved or what they have got wrong.
- Remember that many candidates are writing in a second language; be forgiving of minor linguistic slips. Effective communication is more important than grammatical niceties.
- Occasionally, a part of a question may require a calculation whose answer is required for subsequent parts. If an error is made in the first part then it should be penalized. However, if the incorrect answer is used correctly in subsequent parts then **follow** through **marks should be awarded. Indicate this with** "FT"**.**

SECTION A

1. *Award **[1 mark]** for any explanation that the analysis/design stage is separate from implementation (programming). This may be implied and not stated explicitly.*

   *Award **[1 mark]** for any explanation of the need to change the software as user needs change, and thus the cycle from design to programming then back to redesign then back to reprogramming.* ***[2 marks]***

2. 64; ***[1 mark]***

3. *Award **[2 marks]** for correct answer, **[1 mark]** for a correct formula but a calculation error leading to an incorrect result.*
   (8*60*60) / 1024 = 28.15 MB
   Accept 8 * 60 * 60 / 1000 = 28.8 MB (e.g. permit the use of 1000 instead of 1024)
   [2 marks]

4. 3 : 6
   6 : 12
   9 : 24 ***[2 marks]***

   *Award **[1 mark]** for all 3 C values correct.*
   *Award **[1 mark]** for all 3 length values correct.*

5. (a) Award ***[2 marks]*** *for* compiled modules can execute without further software (compiler or interpreter) while source code would require extra software to execute or distributing source code is less reliable, as it will be compiled/interpreted later but a different compiler might change the functioning of the program
   Award ***[1 mark]*** for distributing the source code enables pirates (other programmers) to steal the code. ***[2 marks]***

   (b) *Award **[1 mark]** for any of the following up to **[2 marks max]**.*
   End-users probably cannot learn anything by reading the code;
   Reading the code does not give a good picture of how the program should be used;
   Source code is usually protected to prevent misuse/piracy; ***[2 marks max]***

6. *Award **[1 mark]** for any function of the ALU*:
   calculations;
   performing Boolean logic operations;

   *Award **[1 mark]** for any function of the CU*:
   controlling the data bus;
   sending addresses to RAM;
   retrieving data from RAM;
   controlling cache or internal buses; ***[2 marks max]***
   Do not award marks for stating the names "Arithmetic Logic Unit" and "Control Unit".

7. (a) Direct-access; ***[1 mark]***

   (b) Sequential-access; ***[1 mark]***

   (c) *Award **[1 mark]** for any of the following up to **[2 marks max]**.*
   Higher storage capacity;
   Removable and thus safer for archiving;
   Tapes can be exchanged easily; ***[2 marks max]***

8. *Award **[1 mark]** for implying that "tasks" are programs, which run in the memory.*
   *Award **[1 mark]** for stating that a multi-tasking system can keep several programs active simultaneously.* ***[2 marks]***

   Do not award any marks if the candidate indicates that different tasks are actually different users, unless it is clear they are talking about processes executing on a server to serve various users.

9. When files are fragmented (pieces scattered around haphazardly on the disk) the defrag utility moves the pieces back together so they are closer to each other. ***[2 marks]***

10. (a) *Award **[1 mark]** for the following or equivalent using different names*:
Analysis, Research, Investigation; ***[1 mark]***

Do not accept: Design, implementation, debugging.

(b) *Award **[1 mark]** each for any of the following up to **[3 marks max]**.*
If the same idea is repeated in different words, only award a single mark for that idea.

Programmers need specific goals to work toward.

Goals should be set by analysts, not by programmers, as the analysts are familiar with the user's needs and wishes.

Programs can only be tested and assessed according to whether they meet the original goals.

Goals need to match the USER'S needs and wishes, not the programmers.

Programmers should not waste time guessing or wondering what goals they are trying to reach.

Lazy programmers might try to finish the job quickly, using inappropriate shortcuts, if the goals are not clearly stated.

The **design specification** should be completed before implementation, and this must include a list of goals.

Accept any reasonable statements that make it clear that:
The **programmers** should not be setting the goals, but rather working toward goals, which were developed from, and agreed to by, the users, because this speeds up the development process by clearly focusing effort. ***[3 marks max]***

Do not accept statements that imply that it is not possible for:
Programmers to set their own goals. The issue is that it is more **efficient** if programmers concentrate on writing good code rather than researching the problem or setting goals.

11. *Award **[1 mark]** for any of the following up to **[2 marks max]**.*
faster debugging through module level testing;
easier to read the code;
modules are reusable;
documentation is simpler; ***[2 marks max]***

12. *Award **[1 mark]** for any of the following*:
fewer cables;
shorter cables;
no hubs/switches;
do not need a central server; ***[1 mark max]***

Do not award marks for comments about the cost of network cards or any devices other than cables and/or hubs.

13. *Award **[1 mark]** for a reason and **[1 mark]** for the justification.*

For example:

The server does not have a user sitting at it and the browser only displays things for a user.
Award no marks for "saving space" or "saving money". ***[2 marks]***

SECTION B

14. (a) (i) 1.00; ***[1 mark]***

(ii) 1.75 ***[2 marks]***

(iii) 0; ***[1 mark]***

(b) Candidates may program this with loops (similar to part a) or simply with if..then.. commands. Either is acceptable.

Award ***[1 mark]*** *for a correct function header.*
Award ***[1 mark]*** *for handling the first category correctly (under 1 kg).*
Award ***[2 marks]*** *for handling the second category correctly, but* ***[1 mark]*** *for a good attempt with a small error.*
Award ***[2 marks]*** *for handling the third category correctly, but* ***[1 mark]*** *for a good attempt with a small error.* ***[6 marks max]***

Handling a category correctly means both the if.then.. (or loop) is correct AND the calculation is correct

Award ***[6 marks]*** *for no errors.*
For example:

```
function COST(val WEIGHT real)
result real                     1 mark for the header

   if (WEIGHT < 1) then
        return 2.00             1 mark for category 1
    elsif (WEIGHT < 10) then
        return WEIGHT * 2.50    2 marks for category 2
    else
        return WEIGHT * 3.00     2 marks for category 3
   endif
endfunction COST
```

Award ***[4 marks]*** *for small errors.*
For example:

```
function COST(WEIGHT)                    0 marks, incorrect header

    declare C integer
     declare T real
     T <-- 0
     C <-- 0
     repeat
       T <-- T + 2.00
       C <-- C + 1                     1 mark for category 1 correct
    until C >= 1

     if (WEIGHT < C) then
        return T
     endif

    while C < 10 do                    1 mark - should stop at WEIGHT
       T <-- T + 2.50
       C <-- C + 1
     endwhile

     if W < 10 then
        return T
     endif

     while C <= WEIGHT do              2 marks for category 3 correct
       T <-- T + 3.00
       C <-- C + 1
     endwhile

     return T
endfunction COST
```

15. (a) *Award **[1 mark]** for any of the following*:
conversion of sensor data (temperature or wind speed or air pressure);
conversion of analog phone-line signals to digital by a modem at the server;
[1 mark max]

(b) Everything is public, so secrecy is not an issue, and it is hard to imagine someone corrupting the data on purpose; ***[1 mark]***

(c) *Award **[1 mark]** for any of the following*:
most graphics software expects standard formats;
it can be more easily displayed by browsers and other systems;
any other answer that deals with **further use** of the photos with standard software;
[1 mark max]

(d) (i) *Award **[1 mark]** for naming the method, plus **[1 mark]** for a brief description.*
A parity check can be used to check each single numeric value when transmitted.

OR

A checksum can be used to check the integrity of a large set of numbers by adding them up and checking the total at the other end. ***[2 marks]***

(ii) *Award **[1 mark]** for each of the following*:
Client must wait until line is up again;
then it must transmit the data (again);
For this to work, data must be collected and saved at the client until the line is available; ***[3 marks]***

(e) *Students may write long, detailed answers, but here are some generic issues to include*:
The data does not need to be displayed;
HTML contains lots of display information, but this is not needed, allowing the file structure to be simpler and download faster;
A simpler DB structure is simpler for the computer to manipulate; ***[2 marks max]***

16. (a) *Award **[1 mark]** for an advantage and **[1 mark]** for a disadvantage.*
Advantage:
Software in ROM is less likely to be lost or corrupted;

Disadvantage:
ROM is usually a lot smaller than hard-disk storage;
ROM software is difficult to upgrade; ***[2 marks max]***

(b) *Award **[1 mark]** for a statement, **[1 mark]** for a justification.*
Calculation results and temporary variables (e.g. data) must be stored in RAM, as they require values to change. ***[2 marks]***

(c) (i) Batch-processing; ***[1 mark]***

(ii) Backing store is usually disk storage. This is necessary because data is collected all day long. It would be unacceptable to simply store the data in RAM in a running computer, as this would be lost if power failed. ***[2 marks]***

(iii) *Award **[1 mark]** for each of the following*:
For example:
A transaction file contains a collection of records, each with the data for one rental ***[1 mark]***.
The master file is a permanent collection of data, usually accumulated as totals rather than individual transactions ***[1 mark]***,
For example, lots of cars rented to various employees of a single company might be collected as charges in a single account for the company in the master file.
[1 mark]
[3 marks max]

17. (a) (i) syntax error; ***[1 mark]***

(ii) an interpreter runs the program, and then stops when it encounters the error; ***[1 mark]***

(b) (i) *Award* ***[1 mark]*** *for any of the following*:
bubble;
shell;
selection;
quicksort; ***[1 mark max]***

(ii) If the file is sorted, a binary search can be used;
(or a sequential search could have an early exit)
If not, a much slower sequential search must be used; ***[2 marks]***

(c) (i) The error message file contains a line number or some other indication of where the error occurs.
The programmer must find that command, change it, and run the compiler again, repeating this process until all compile-time errors have been eliminated. ***[2 marks]***

(ii) The executable file (or object module) - that is the program that actually runs; ***[1 mark]***

(d) Comments are marked (delimited) by /* .... */.;
The compiler must ignore the words between the delimiters - when it finds a /*, it continues reading until it find */, and then resumes checking words; ***[2 marks]***