M13/5/COMSC/HP2/ENG/TZ0/XX/M

International Baccalaureate®
Baccalauréat International
Bachillerato Internacional

MARKSCHEME

May 2013

COMPUTER SCIENCE

Higher Level

Paper 2

14 pages

General Marking Instructions

1. Once markscheme is received mark in pencil until final markscheme is received.

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

3. 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.

4. 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.

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

6. 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.

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

8. Record the mark awarded for each of the four questions answered in the Examiner Column on the cover sheet. Add up the marks awarded and enter this in the box marked TOTAL in the Examiner Column on the cover sheet.

9. 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.**

10. 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.

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 HL Paper 2 Markscheme

Mark Allocation

Candidates are required to answer ALL questions ***[20 marks]*** for question 1, ***[20 marks]*** for question 2, ***[20 marks]*** for question 3 and ***[40 marks]*** for question 4. Maximum total = ***[100 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 that part of a question.

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

- Each statement worth one 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.
- If the candidate's answer has the same meaning or can be clearly interpreted as being the same as that in the markscheme then award the mark.
- Mark positively. Give candidates credit for what they have achieved, and for what they have got correct, rather than penalizing 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. In this subject effective communication is more important than grammatical accuracy.
- 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**".

1. (a) the possibility of playing against many other users;
(real-time) interaction with other user (if different from playing. i.e. chatting);
the ability to receive updates;
online help / diagnostics;
easy possibilities of switching games;
comparison of scores with others;
can play form any device (that can connect);
no need to install software; ***[2 marks max]***

(b) can divide into teams;
therefore speeding up the development / greater efficiency in the development;

modules are smaller than the complete programs;
therefore easy to locate (and fix) errors / easier to maintain;

modules can be reusable;
therefore speeding up development of this (or other) programs; ***[4 marks max]***

(c) 8; ***[1 mark]***

(d) array;
of integer; ***[2 marks]***
Accept an example showing the above.

Question 1 continued

(e) *Award marks as follows up to* ***[7 marks max]****.*
Award ***[1 mark]*** *for creation of digits (can be passed as a parameter);*
Award ***[1 mark]*** *for use of nested loops;*
Award ***[1 mark]*** *for correct use of nested loops;*
Award ***[1 mark]*** *for checking each grid element and incrementing of digits array;*
Award up to [max 3 marks] for checking the digits array as follows:
Award ***[1 mark]*** *for a correct loop*
Award ***[1 mark]*** *for checking of digits structure;*
Award ***[1 mark]*** *for correct return statements and string returned;*

Example answer 1 (adjusting the cell content to match the array index):

```
String checkGrid()
{
  int cell;
  int[] digits = new int[9];


  for (int i = 0; i < 3; i = i + 1)
  {
    for (int j = 0; j < 3; j = j + 1)
    {
      cell = grid[i][j]- 1;
      digits[cell] = digits[cell] + 1;
    }
  }
  for (int k = 0; k < 9; k = k + 1)
  {
    if (digits[k] != 1) return "Failure";
  }
  return "Success";
}
```

Example answer 2 (ignoring digfits[0]:

```
String checkGrid()
{
  int cell;
  int[] digits = new int[10];


  for (int i = 0; i < 3; i = i + 1)
  {
    for (int j = 0; j < 3; j = j + 1)
    {
      cell = grid[i][j];
      digits[cell] = digits[cell] + 1;
    }
  }
  for (int k = 1; k < 10; k = k + 1)
  {
    if (digits[k] != 1) return "Failure";
  }
  return "Success";
```

[7 marks]

(f) *Award marks as follows:*
Award ***[1 mark]*** *for confirmation that the team* ***is*** *able to reuse the* `checkGrid()` *method together with at least some justification;*

Award up to ***[2 marks]*** *for details of the changes (e.g. start and end values in the nested loops would have to be passed as parameters and method would have to called several times, once for each 3x3 grid, once for each row and once for each column);*

Award ***[1 mark]*** *for an example or detail relating to each 3x3 square;*

Award ***[1 mark]*** *for an example or detail relating to each row and/or column;* ***[max 4 marks]***

<u>Example</u>:

Start and end values in the nested loops would have to be passed as parameters, and the method would have to called several times, once for each 3x3 grid, once for each row and once for each column;

e.g. for the top right-hand corner, the start and finish parameters would be 0 and 3 for the row loop and 6 and 9 for the column loop;

e.g. for the first row, the start and finish parameters would be 0 and 1 for the row loop and 0 and 9 for the column loop;

2. (a) each node has an field/element that links it with the next one;
this element points to the address of the next node; ***[2 marks]***

(b) in order to move through the list a variable must successively point to the next node in the list;
it will initially point to the head of the list;
it cannot be one of the (permanent) list values (*e.g.* the head) as it would then be lost;

***Note:** Answers must address traversal - not adding to a list.*

[2 marks max]

(c) to allow for changes in the number of students/marks;
and the number of marks for each student might be different ;
can be used each year with a different number of students;
means no re-programming needed;
saves (wasting) memory; ***[2 marks max]***

(d)
$$\begin{tikzcd}
\text{head} \arrow[r] & \text{Kuyt, 475, firstMark} \arrow[r] \arrow[d] & \text{Peters, 386, firstMark} \arrow[r] \arrow[d] & \text{null} \\
 & 23 \arrow[d] & 54 \arrow[d] & \\
 & 12 \arrow[d] & 87 \arrow[d] & \\
 & 43 \arrow[d] & \text{null} & \\
 & 12 \arrow[d] & & \\
 & \text{null} & &
\end{tikzcd}$$

Diagram should also show links between the nodes in the marks' linked list.
showing three linked lists;
correct position of marks' linked list;
all pointers (including head and null) are correct; ***[3 marks]***

Question 2 continued

(e) *Award marks as follows up to* ***[7 marks max]****.*
Award ***[1 mark]*** *for initialization;*
Award ***[1 mark]*** *for use of temp;*
Award ***[1 mark]*** *for correct loop;*
Award ***[1 mark]*** *for incrementing total and count;*
Award ***[1 mark]*** *for moving through loop,*
Award ***[1 mark]*** *for correct check for count;*
Award ***[1 mark]*** *for calculation of average (even if list is empty);*
Award ***[1 mark]*** *for returning values correctly (taking account of possible empty list);*

Example answer:

```
double averageMark(Mark firstMark)
{
  double average = -1;
  int total = 0;
  int count = 0;
  Mark temp = firstMark;
  while (temp != null)
  {
    total = total + temp.score;
    count = count + 1;
    temp = temp.nextMark;
  }
  if (count > 0)
  {
    average = total / count;
    return average;
  }
  else
  {
    return -1;
  }
}
```

[7 marks]

(f) *Award marks as follows:*
Award ***[1 mark]*** *for creating Student object*
Award ***[2 marks]*** *for correctly calculating averageMark (****[1 mark]*** *if no dot notation)*
Award ***[1 mark]*** *for returning the mark*

```
public double studentAverage(int id)
{
  Student temp = locateStudent(id);
  double averageMark = average(temp.firstMark);
  return averageMark;
}
```

[4 marks]

3. (a) it provides fast access to the records;
as it is direct access;
as it has O(1) efficiency; ***[2 marks max]***

(b) (i) 456 ***[1 mark]***

(ii) *e.g.* 4456 ***[1 mark]***

(c) (i) hash algorithm can generate 1000 values;
therefore table needs to have 1000 locations; ***[2 marks]***

(ii) there must be less than 1000 records (accept a value such as 500);
as provisions (space) need to be left to deal with possible collisions; ***[2 marks]***

Or

(theoretically) a max of 1000 records could be handled;
although this would leave no room for collisions;

(d) *Award marks as follows up to **[8 marks]**.*
*Award **[1 mark]** for use of* `hash()`*;*
*Award **[1 mark]** for correct loop (allow looping until the end of the table);*
*Award **[1 mark]** for correct comparison;*
*Award **[1 mark]** for early exit;*
*Award **[1 mark]** for moving on in case of collisions;*
*Award **[1 mark]** for wrapping around if end of array;*
*Award **[2 marks]** for correct use of* `displayRecord()` *(award **[1mark]** for a reasonable attempt];*

Example answer:

```
void findRecord(int id)
{
  Boolean found = false;
  int hashValue = hash(id);
  while (!found)
  {
    if (table[hashValue].idNumber == id)
    {
      found = true;
    }
    hashValue = hashValue + 1;
    if (hashValue > 999)
    {
      hashValue = 0:                // wrapping around
    }
  }
  displayRecord(table[hashValue].memoryLocation);
}
```

[max 8 marks]

(e) there will be no problems regarding collisions (therefore do not have to deal with them);
as every index points to a unique location;
if number of records increases;
there are no concerns with changing hash function/size of hash table;

if the full index comes with a binary search then search efficiency is O($\log_2$ n) which comes close to O(1) for less than 1000 records;
this might be close to or better than a hash table whose efficiency deteriorates to linear search when it gets full;

space can be wasted in hash table when making provision for collisions:
whereas indexing only using the memory as required;

[max 4 marks]

4. (a) its contents change whenever a program using it changes / memory that does have a fixed life;
MUST include example *e.g.* could contain a downloaded app one moment and then an MP3 file the next; ***[2 marks]***

(b) start-up instructions;
Accept operating system; do not accept RISC without comment. ***[max 1 mark]***

(c) with the use of sub-menus;
selecting one option/icon leads to another set of choices;

AND second example with some comment

by "sliding" the screen sideways;
revealing more icons/choices; ***[4 marks]***

(d) 3G/4G phones allow users to connect to the Internet;
users are now downloading files that use a large bandwidth;
putting a strain on/filling the capacity of the Internet lines; ***[3 marks]***

(e) *Award* ***[2 marks]*** *for good discussion of positive examples;*
Award ***[2 marks]*** *for good discussion of negative examples;*
Award up to ***[2 marks]*** *for a justified conclusion;*

Examples could include amongst others:

positive:
quizzes downloaded from the teacher (via Bluetooth)
real-time research on lesson topic

negative:
using Internet for cheating
distraction by social networking sites ***[6 marks]***

(f) the ADC chip is needed to translate digital signals (*e.g.* music files);
to analogue signals (sound); ***[2 marks]***
Award ***[1 mark max]*** *if no example given.*
Allow any example in either direction.

(g) *Award up to* ***[4 marks max]***
Answer could include the following:

low power consumption prolongs battery life;
which is an advantage with mobile devices;
cannot only transmit over short distances;
minimizes chance of privacy loss;
proximity requirements could be inconvenient (give example); ***[4 marks]***

(h) *Award* ***[1 mark]*** *for a good example (e.g. testing water quality).*
Award up to ***[2 marks]*** *for a clearly explained possible use (eg sensor could be connected to the phone by Bluetooth).* ***[3 marks]***

(i) if the phone is lost (back of taxi for example);
then the network provider can disable it, preventing data loss/maintaining security;

Award ***[1 mark]*** *for preventing data loss / help current security issue – ie phone can be disabled if being found to be misused,* ***[2 marks]***

(j) *Award up to* ***[3 marks max]***
a pixel is the smallest element on a screen that can be addressed by the memory;
by increasing the pixel density more elements can be addressed on the same area of screen;
therefore more data from an image can be displayed in the same area;
making the image sharper;
although continually increasing resolution will reach a point where the human eye no longer perceives the difference; ***[3 marks max]***

(k) (i) *There are many examples. Award up to* ***[2 marks]*** *for a good, clearly described answer.*
e.g. an incoming call will cause the phone to ring;
putting on hold / interrupting a song being played; ***[2 marks]***

(ii) all essential parameters from the song program will be pushed onto a stack;
ready to be popped back into the CPU registers when call is over; ***[2 marks]***

(l) *Suggestions could include, amongst others*:
Address book could be used as a BST in which each node is also a BST *etc*.
Trees could then be recursively searched.
ArrayList could be used ordered by the first letter. Direct access to list then sequential search.
An array of linked lists could be used (one for each letter), then sequential search.

Award marks according to the bands below.

Marks	**Details**
1–2	Minimal knowledge and understanding of the relevant issues or concepts. The answer may be little more a description of the process from the point of view of the user.
3–4	A response with some knowledge and understanding of the relevant issues and/or concepts. There is either some attempt to describe either suitable structures or processes, or both structures and processes are addressed but without demonstrating a detailed understanding.
5–6	A response with a detailed knowledge and understanding of the both structures and processing relevant to the question. Full marks would require an answer that would successfully carry out the process described in the question.

[6 marks]