N10/5/COMSC/SP2/ENG/TZ0/XX/M

International Baccalaureate®
Baccalauréat International
Bachillerato Internacional

MARKSCHEME

November 2010

COMPUTER SCIENCE

Standard Level

Paper 2

11 pages

https://xtremepape.rs/

*This markscheme is **confidential** and for the exclusive use of examiners in this examination session.*

*It is the property of the International Baccalaureate and must **not** be reproduced or distributed to any other person without the authorization of IB Cardiff.*

https://xtremepape.rs/

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). 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 TL and IB Cardiff. 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.

https://xtremepape.rs/

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.

https://xtremepape.rs/

Subject Details: Computer Science SL Paper 2 Markscheme

Mark Allocation

Candidates are required to answer ALL questions ***[20 marks]*** for question 1, ***[20 marks]*** for question 2 and ***[30 marks]*** for question 3. Maximum total = ***[70 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 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. 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**".

https://xtremepape.rs/

1. (a) *Award **[1 mark]** each for the definition and **[1 mark]** for an example, up to **[4 marks max]**.*
public marks a method/variable that is accessible by code outside the class;
In the `Student` class, the constructor and the `getName()` method are public;

private marks a method/variable that is accessible only by code within the same class;
In the `Student` class, the `name`, `yearOfBirth`, `monthOfBirth` and `dayOfBirth` variables are all private; ***[4 marks]***

(b) *Award marks as follows up to **[2 marks max]**.*
*Award **[1 mark]** for the correct signature line.*
*Award **[1 mark]** for the correct return line.*

Example answer:

```
public int getYearOfBirth()
{
  return yearOfBirth;
}
```
[2 marks]

(c) (i) *Award **[1 mark]** for each correct answer, up to **[2 marks max]**.*
Student A: 10 years
Student B: 10 years
Student C: 9 years ***[2 marks]***

(ii) The birthday has passed if the current month is greater than the birth month. Or, if the birth month is the same as the current month, the current day is greater than (or greater than or equal to) the birth day. ***[2 marks]***

(iii) *Award marks as follows up to **[5 marks max]**.*
*Award **[1 mark]** for signature line.*
*Award **[2 marks]** for correct condition.*
*Award **[1 mark]** for each return statement.*

Example answer 1:

```
public int getAge()
{
  if ((currentMonth > monthOfBirth)
          || ((currentMonth == monthOfBirth)
                  && (currentDay >= dayOfBirth)))
  {
    return currentYear - yearOfBirth;
  }
  else
  {
    return currentYear - yearOfBirth - 1;
  }
}
```

continued ...

https://xtremepape.rs/

Question 1 continued

Example answer 2:

```
public int getAge()
{
  boolean birthdayPassed = false;

  if (currentMonth > monthOfBirth) birthdayPassed = true;
  if (currentMonth == monthOfBirth)
  {
    if (currentDay > dayOfBirth) birthdayPassed = true;
  }

  if (birthdayPassed)
  {
    return currentYear - yearOfBirth;
  }
  else
  {
    return currentYear - yearOfBirth - 1;
  }
}
```

Accept either > or >= when comparing current and birth days. ***[5 marks]***

(d) *Award marks as follows up to **[5 marks max]**.*
*Award **[1 mark]** for correctly reading from the file.*
*Award **[1 mark]** for correctly determining the end-of-file condition and terminating.*
*Award **[1 mark]** for correctly checking if the student is 12.*
*Award **[1 mark]** for correctly printing the student's name.*
*Award **[1 mark]** for a correct loop.*

Example answer:

```
Student theKid = getNext();
while (!theKid.getName().equals("XXX"))
{
  if (theKid.getAge() == 12)
  {
    output(theKid.getName());
  }
  theKid = getNext();
}
```

[5 marks]

Total: [20 marks]

https://xtremepape.rs/

2. (a) (i) *Award up to **[2 marks max]**.*
A student can be identified by a simple index value;
You do not need separate variables for each student;
Names and grades can be changed easily; ***[2 marks]***

(ii) *Award marks as follows up to **[4 marks max]**.*
*Award **[1 mark]** for using a loop to search for the best average.*
*Award **[1 mark]** for correct loop start/end values.*
*Award **[1 mark]** for correctly initializing the best score variable.*
*Award **[1 mark]** for correct return of* `bestName` *as* `String`*.*

Example answer:

```
String bestStudent(String[] names, float[] averages, int numStudents)
{
  float bestScore = averages[0];
  String bestName = names[0];

  for (i = 1; i < numStudents; i = i + 1)
  {
    if (averages[i] > bestScore)
    {
      bestScore = averages[i];
      bestName = names[i];
    }
  }
  return bestName;
}
```

Accept alternative algorithms such as keeping track of the index of the highest average and then using that after the loop terminates to look up the name of the student with the highest average. ***[4 marks]***

continued ...

https://xtremepape.rs/

Question 2 continued

(b) (i) *Award up to **[2 marks max]**.*
The elements of the `studentNames` and `gradeAverages` arrays must be kept aligned;
Sorting only one array would break the association between the two;
Changes to element order in either array must be duplicated in the other; ***[2 marks]***

(ii) *Award marks as follows up to **[8 marks max]**.*
*Award **[2 marks]** for correct outer loop.*
*Award **[3 marks]** for correct inner loop (including varying loop limit).*
*Award **[1 mark]** for correctly swapping adjacent elements via a temp variable.*
*Award **[1 mark]** for swapping elements in both arrays together.*
*Award **[1 mark]** for sorting based on averages array (not names).*

Example answer:

```
void rankStudents(String[] names, float[] averages, int numStudents)
{
  String tempName;
  float tempAverage;

  for (int k = 0; k < numStudents - 1; k = k + 1)
  {
    for (int i = 1; i < numStudents - k; i = i + 1)
    {
      if (averages[i] > averages[i - 1])
      {
        tempName = names[i];
        names[i] = names[i - 1];
        names[i - 1] = tempName;

        tempAverage = averages[i];
        averages[i] = averages[i - 1];
        averages[i - 1] = tempAverage;
      }
    }
  }
}
```

[8 marks]

(c) *Award up to **[4 marks max]**.*
Could create `Student` object class;
`Student` objects would store both `name` and `average` as attributes (variables);
The collection of `Student` objects could be stored in a single array;
The array could be sorted simply since there is only one array;
The `Student` class would need either a method to run the average of a student or a method to compare the averages of two student objects; ***[4 marks]***

Total: [20 marks]

https://xtremepape.rs/

3. (a) (i) *Award up to **[2 marks max]**.*
More robust than other two;
Versatile (can be used for other purposes/buying food, drinks from vending machines on credit basis);
More difficult to fraud;
Hold sophisticated programs that could enhance security: adding PIN numbers, finger print *etc.*; ***[2 marks]***

(ii) *Award up to **[2 marks max]**.*
Finger print;
Voice print;
Retina scan; ***[2 marks]***

(iii) *Award up to **[4 marks max]**.*
Converts analog input data (such as spoken words, signature identification);
Into computer usable code;
By comparing the patterns inputted;
With a set of pre-recorded patterns;
After the computer matches the patterns;
It executes the appropriate command; ***[4 marks]***

(iv) *Award **[2 marks]** for any comparison, up to **[4 marks max]**.*

STAFF	*PASSENGER*
Higher level of security because they can enter any area.	Lower level, only restricted areas.
Identification of an employee can be verified using smart cards linked to biometric system.	The only form of identification is passport (can be scanned and matched to the central base).
etc.	

[4 marks]

(b) (i) *Award up to **[1 mark max]**.*
Operating system
Software;
That starts up the computer;
Coordinates the hardware components;
The application software programs;
Controls all the programs;
Manages the use of primary and secondary storage;
etc. ***[1 mark]***

continued ...

https://xtremepape.rs/

Question 3 continued

(ii) **Multi-user**
Many users/clients can be supported at the same time; ***[1 mark]***

(iii) **Multi-tasking**
OS can execute more than one task at a time; ***[1 mark]***

(iv) *Award up to **[3 marks max]**.*
Needs to see position of airplanes;
Communicate with the persons on the ground;
Coordinate all departments such as fire department, first aid, if needed;
Communicate with pilots;
etc.
[3 marks]

(v) More than one user can check in at different desks/kiosks at the same time;
All accessing the same server; ***[2 marks]***

(c) (i) **Batch processing**
Technique whereby data is collected over a period of time;
Then is input at one time for storage or for processing and output; ***[2 marks]***

(ii) **Real-time processing**
Immediate processing;
Each input is processed and there is immediate feedback (action can be taken right away); ***[2 marks]***

(iii) *Award up to **[2 marks max]**.*
Online processing
Anything in a computer that is online;
Is linked up to current processing operation;
And similarly to real-time;
Immediate processing;
Each input is processed and there is immediate feedback (action can be taken right away); ***[2 marks]***

(d) *Answer should outline any example of airport systems and an explanation (referring to that system).*
*Award **[2 marks]** for correct (**[1 mark]** for vague example), up to **[4 marks max]**.*
'Client' refers to a computer (on which there is a software application) that requests actions;
'Server' means a computer that provides services for the clients that are available via network; ***[4 marks]***

Total: [30 marks]

https://xtremepape.rs/