November 2016 subject reports

Computer Science

Overall grade boundaries

Higher level

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 15	16 – 31	32 – 43	44 – 52	53 – 62	63 – 72	73 - 100

Standard level

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 18	19 – 38	39 – 49	50 – 57	58 – 65	66 – 74	75 - 100

Internal assessment

Component grade boundaries

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 5	6 – 10	11 – 15	16 – 19	20 – 23	24 – 27	28 - 34

General comments

Generally the work submitted followed the instructions as laid out in the Computer Science Guide and the Handbook of Procedures for the Diploma Programme 2016, section B4.4.

Some recommendations from the Principal Moderator:

- The product folder must contain some evidence of the product in order to verify it exists – preferably both the final product and the product at design stage, for example both executable jar file and original java files, or an Access database client version (that opens a full-screen switchboard) and a design version that does not. In extreme cases, for example when the product is being developed on-line, the product folder should contain screenshots of the product being created.
- The video/screencast should be 5-7 minutes (maximum) and should only show the proper working of the final solution – the use of techniques should be described in Criterion C using extended writing. It is suggested that candidates use their *success criteria* identified in Criterion A and their test plan (Criterion B) to script the screencast.
- A few screencasts could not be properly viewed. The video format must be recognized by cross-platform video software like VLC player; the screen resolution should be adjusted before recording a screencast, lest the result becomes too small to be viewed; the screencast should be narrated to explain what is being shown.
- Even though it is not a requirement, teachers are asked to provide pertinent comments on how they awarded marks to the candidates in their sample. This facilitates the moderator's validation of the teachers’ marks.

The range and suitability of the work submitted

The scenarios described typically allowed for worthwhile projects. As expected, the majority of solutions concerned programming projects and the majority of those had been coded in Java. On the other hand, an encouraging number of candidates tried their hand at web design, [Access] databases and Android app design. It is hoped that the range of solutions continues to expand.

In this session a significant number of products were left unfinished and it is recommended that teachers set aside enough time for students to complete all stages in detail and to produce the product as described in the design.

The quality of the solutions showed a wide range and not all solutions had been developed to the level of complexity expected of IB DP candidates. Some examples of trivial products include: Java programs that mainly focus on GUI and not on actual functionality, Java programs that consist of one class only, rudimentary versions of freely available games (like Sudoku), Access databases that contain less than three tables or are non-relational, websites that are template-based (Wordpress, Wix or Weebly) or that have minimal content, basic Excel projects, Scratch projects that had not been properly designed.

A few schools adopted a standard approach to the product and/or report where all candidates used the same components, layout and content but applied to different problems, leading to generic reports and products. These approaches must be discouraged.

Highly successful solutions tended to incorporate features from more than one software. For example, website projects that incorporate JavaScript / PHP / SQL functionality, or programming projects that interact with an Access database or with on-line resources.

Candidate performance against each criterion

Criterion A Planning

This remains the most straight-forward criterion. However, several candidates did not follow the expected sequence:

- investigate a situation
- identify client/adviser
- explicitly consult the client (and/or adviser)
- describe the scenario with explicit reference to evidence of the consultation added in an appendix
- choose a solution
- describe the rationale for the solution and also for the software to be used
- outline comprehensive Criteria for Success for the chosen solution.

Too many candidates decided on a product ('I want to make a website/program a game') and then found a client to match. Contrived tasks and clients were routinely seen in the weaker samples submitted. Many candidates had generic success criteria – these criteria must be specific and testable. The *success criteria* are essential to the project and must be explicitly addressed in the test plan (Criterion B) and in the evaluation (Criterion E) and preferably also in the screencast.

Criterion B Solution overview

The quality of the solution overview has improved in this session with many projects showing a concerted effort to design the solution. Some even include an improved design after client feedback. However, most designs are not comprehensive and fail to achieve in the highest descriptor. Records of Tasks were generally only partially complete, typically because the final product had not been implemented / fully tested by the client. A wide variety of test plans were seen. The better ones aligned with the *success criteria*.

Please note: The use of the proper template in forms.zip is mandatory – the use of a different version, with incorrect columns and headings, will be penalized. If no Record of Tasks is included or if there is no evidence of a design then 0 marks will be awarded.

Criterion C Development

Most candidates made a good attempt to document the development of their product and the techniques used. However, the quality of the explanations and the completeness of techniques typically left something to be desired. The complexity of the product must be justified by the candidate in the write-up. A seemingly complex product without proper explanations of complex techniques used in the product, only achieves moderate complexity. Similarly, high levels of ingenuity must be justified by algorithmic thinking (e.g. explanations of complex data structures, algorithms or macros).

Criterion D Functionality and extensibility of product

The screencast should only show the proper working of the solution as outlined by the success criteria in Criterion A. Many screencasts focused instead on the development of the solution, which made them too lengthy. Others only showed the working of the interface, without showing actual functionality of the intended solution.

Criterion E Evaluation

Most evaluations did not achieve in the highest descriptor because of very limited client involvement. The final product (after testing) is expected to be implemented and used/tested by the client before client feedback is given. For full marks evidence of client feedback against the *success criteria* must be included (typically in the appendix) and it must be discussed and referred to in the candidate's evaluation against the success criteria. Recommendations should be realistic in relation to the actual product – for example 'adding network capability' is not a realistic improvement for a low-level product.

Recommendations for the teaching of future candidates

- The aim, in most cases, of the IA in Computer Science is to create a working solution for a real client. The consultation (included as an appendix) should be the basis for the description of the scenario, leading to a range of specific and testable success criteria for a chosen solution. All high scoring projects showed ample evidence of client involvement.
- Criterion B should provide evidence of a rigorous design stage with an overview of all five stages of the project (including the actual intended use of the product by the client) in the Record of Tasks, detailed layout design sketches that include annotations for complex techniques, evidence of algorithmic thinking (in the form of flowcharts, UML diagrams, pseudo-code, ER diagrams, structured database decomposition using 1NF – 3NF, query and macro design), and a test plan that addresses all success criteria identified in Criterion A. All high scoring projects included a thorough design stage.
- Criterion C provides candidates with the opportunity to demonstrate their knowledge and understanding of the tools and techniques used in creating the product. The use of tools/techniques should be explained in relation to screenshots that show their use.

- Criterion D does not require written documentation. The screencast should be limited to about 5-7 minutes and should only show the proper working of the final solution. The structure of the screencast should be scripted by the candidate. For example, the screencast could show the testing of the implemented solution following the test plan from criterion B. Successful screencasts showed comprehensive evidence of the solution's functionality with lots of data, but were edited to avoid viewing tedious data entry. Candidates are advised to test their screencasts on different media players and devices to ensure the playback is correct.
- Extensibility is evidenced by a detailed design in Criterion B, by a detailed description of the creation process in Criterion C and, in case of a programming project, by a properly structured and annotated code listing in an appendix.
- Criterion E should provide evidence of a rigorous evaluation stage. The client feedback (added in an appendix) should be discussed by candidates as part of their own evaluation of the solution. A table showing the *success criteria* identified in Criterion A with a 'met/not met' evaluation is not sufficient to achieve in the highest descriptor. Recommendations for improvement should go beyond the success criteria that have not been met.
- A word of caution: treating the project as a purely academic exercise typically means that there is no proper client and that the solution is not being implemented, which will have an impact on Criterion A, Criterion D and Criterion E.
- The recommended word count for each section, as indicated in the Teacher Support Material (TSM), is only for guidance. The overall word count of 2000 words however, is a fixed limit and a moderator is not required to read beyond this limit, which could cause a loss in marks in Criterion E.

Further comments

For additional information regarding the Computer Science IA, please consult:

- Computer Science Guide (pages 56-72).
- Teacher Support Material (Internal Assessment) available on the OCC.
- Forms.zip templates.
- Submission of the Computer Science IA in the Handbook for Procedures for the Diploma Programme 2017 (Section B4.4). Note that the Handbook is updated yearly.
- IB Coordinator Notes.

For additional professional development regarding the Computer Science IA, please consider:

- Getting involved in the Computer Science OCC discussion forum.
- Registering for Computer Science workshops (either face-to-face or online).

Higher level paper one

Component grade boundaries

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 17	18 – 35	36 – 47	48 – 57	58 – 67	68 – 77	78 - 100

General comments

The areas of the programme and examination which appeared difficult for the candidates

Terminology is not always used in an appropriate and competent way and questions are often approached (hence, answered) by providing superficial common knowledge on IT. It is helpful to remind that this is an exam in computer science: the focus is always on the side of the computer/system/network/algorithm. For example by "usability issues of a smartphone" we intend to address the technical issues that components of a smartphone may present, and we do not expect a list of any possible disability a user may have.

Several questions were not read carefully: programs have been constructed not addressing the specification; generic wording (rather than pseudocode) was used to illustrate algorithms when programs were explicitly asked.

Procedures and terminology on trees happened to be incorrect in several responses.

Some marks have been lost for a tendency of going off-course or answering with too vague, out-of-context answers.

In relation to the stack structure, many responses were returning just the basic knowledge on what a stack is, without showing the ability of using a stack for the purposes defined: in other words, the basic knowledge is solid, but the answers were not informative enough to address objective 3 questions (Explain how to use X for a certain purpose Y given the specific context Z).

Some responses on truth tables were any number between 5 and 7 of possible row values for the three input. Some with 8 values were containing repeated entries. This shows a methodological problem: not just that the connectives are maybe not known, rather that there is no structured method and solid understanding that out of 3 input one gets just 8 different combinations.

Mathematical writing: the vagueness in using the language for algorithms has shown little respect for the boundaries of loops, where <, =<, >, "at least" have been wildly used in any possible way. Similarly some candidates seem not to appreciate the difference in naming a power function from an exponential or a factorial, or the difference between MOD and DIV. Not being familiar with the basic mathematical notation and (understanding it mathematically) can just contribute to the creation of unsafe software.

The areas of the programme and examination in which candidates appeared well prepared

Good basic knowledge, most candidates have completed reasonably well or very well section A as well as most questions relative to Assessment Objective 1 or Assessment Objective 2 in Part B. This shows a good coverage of the syllabus, but not necessarily as deep as expected at HL.

The strengths and weaknesses of the candidates in the treatment of individual questions

I will list here only the problematic questions that need reflection and consolidation, meaning that the others are generally fine.

1. Answers of the kind "a person with no fingers cannot use the phone; etc.."; the focus must fall on the device.

4. Tables with 5,6,7 inputs or 8 inputs with duplications.

6. Question poorly understood: marks lost in a and c. Some answers for a were essentially following the code ("if N is 0 or less, multiplication is not executed.."). Answers in c were describing general knowledge on recursive algorithms rather than addressing the purpose of "the given one"

7. Some responses were identifying also B as a leaf. Usually at least one traversal is known. To consolidate.

8a. Some responses were not returning a 'system' flowchart, rather any diagram where the system components were not present: this has caused a loss of marks for lack of information (for example, a database accessed only in one direction or in one operation).

8d. The three questions sub-questions have been attempted with some repetitions: this shows general knowledge or 'general situation awareness' of when and how data can be at risk and why. However many responses were vague enough not to describe the situation with reference to the basic technical terminology of the devices/systems. Marks have been lost.

9b. Many responses were going off course, answering "why user interfaces are useful" rather than "identifying two features of UI".

9d. Responses to these sub-questions were just saying what an interpreter/compiler is, rather than convincing how that knowledge can be applied in the specific scenario.

9e. Question not read/understood (despite the given example). Many incorrect responses, aiming in constructing the algorithm that outputs the sum of the odd numbers up to N, rather than the sum of the first N odd numbers as specified.

10a. Analog data very poorly understood. Most responses were just saying "something not digital" or "something that can be converted into digital". If digital is taught then also analog can be taught. These are basic notions that were answered with layman knowledge.

10b. Poor understanding of transducers (and their relation to sensors). Some answers, however trying to put all elements together, contained incorrect/imprecise sentences (confusion between the role of a thermometer and a thermistor.. which were not required, but if you write it, write it well!). Most answers not addressing the feedback. Marks lost.

10c. Mostly answered with general knowledge on polling and interrupt without reference to the scenario. Marks lost.

11d. Mostly answered by just describing the stack as LIFO structure with push and pop, not in context. Marks lost.

12. This question has been entirely attempted (and in some cases with very good success) by very few candidates. The majority of them struggled, some of them also with time.

12b. Incorrect answer giving the values of some elements in the matrix pointed by the indexes, rather than their own initial values. This has carried through the rest of b.

12c. The pseudocode contains essentially a portion of code that is almost identical and repeated 4 times: what really changes are the indexes. Those candidates that understood well (or had time to read and understand well) the specification realized this feature, and easily got full marks. The others did not complete the exercise or completed it incorrectly, suggesting they were running short of time and were in doubt on how to manipulate indexes. Consequently, they lost many marks.

Recommendations and guidance for the teaching of future candidates

Please focus on the use of correct terminology and concepts: we are interested in computer science and not in the sociology of the use of IT devices. It is in this spirit that we write question and Markschemes.

Pseudocode exists to be used. When pseudocode is explicitly asked in some question, it is because this is the maximal level of abstraction that would allow the examiners to assess certain competencies in programming and algorithmic thinking, ones that can be turned into valuable marks.

The exam paper is written in a way to allow for some questions to be answered in descriptive way and other to be answered with pseudocode.

Please keep teaching well, staying on the technical/scientific side, and ensure students can apply the concepts they learn: in objective 3 questions there were too many answers that were showing a plain regurgitation of notions as if they were quizzes.

Attending workshops may be a valuable experience to quickly clarify in which directions and how to strengthen and consolidate some competencies.

Further comments

A general impression is that the majority of students struggled with time in order to complete the exam paper. The effect of this lack of time (or mismanagement of time) is that some responses that definitely carry HL value (towards the end of the exam paper) were presented in a very sketchy way or not answered at all.

Given that the last questions address programming and algorithmic thinking, it is unclear whether these competencies are to be considered a weakness that needs consolidation: it is related to the strategy of attempting answering the questions, or on how verbosely previous questions have been answered.

In some cases the general quality of some HL responses is worse than the average SL: I do not know whether schools are supposed to support candidates in their choices at HL.

Standard level paper one

Component grade boundaries

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 15	16 – 31	32 – 37	38 – 42	43 – 46	47 – 51	52 - 70

General comments

The areas of the programme and examination which appeared difficult for the candidates

Please refer to the HL report for the common question.

All students completed well the exam paper, showing that the level of difficulty and length of the exam paper was in line with previous sessions. Some students do not confidently use/apply mathematical notation so that modulus or division (truncated) operations are used inappropriately.

Some candidates cannot say confidently the purpose of a cache when in relation to a RAM and processor, and without making use of drawing, seem to indicate the merits of paging.

Usability issues of smartphones were addressed (and answered) as a list of possible disabilities of some users -- off context.

The areas of the programme and examination in which candidates appeared well prepared

General factual knowledge is good, across the entire curriculum. Some questions were answered too vaguely or off-context, leading to a loss of mark: use of vague common-sense terminology, rather than technical and complete terminology is a general weakness that needs to be addressed.

Programming -- several students attempted the code, using pseudocode when requested. AT times, the use of pseudocode was not detailed enough to award marks: for example "loop i" is much less informative than "loop i to n", which approaches the correct (and expected) "loop i from 1 to n". By lack of information, we can acknowledge that there is a good attempt, but the result is not enough to award the mark.

The strengths and weaknesses of the candidates in the treatment of individual questions

I will list here only the weaknesses, meaning that other elements are already good. Please refer to HL for common questions.

6a This has received a variety of answers: 1, P, N.

6b. The trace has been usually completed well, with some exception on the P column.

6c. Many answers were addressing the operation of the algorithm rather than what it computes.

10b. The majority of students attempted it and many got it right of almost right. The weakest answers exhibit little ability of using pseudocode and in general programming/algorithmic thinking.

10c.ii Some students did not attempt it in pseudocode. Other wrote it showing uncertainty in how to use pseudocode. However, some got it partial marks.

10d. The quality of these answers, in reading, shows the "imprinting" that these students have in describing the algorithms: these needs to be addressed. These responses can be verbose and story-telling oriented, while missing crucial information that would award the mark. In other words, candidates are not describing the steps of an algorithm, rather a recipe to achieve something as if it were commented on YouTube. This is a problem: with storytelling one has stories, and not algorithms. If the key structures are not identified (example ".. now I go in the other array" -- which one? There are three, and they have unique names), marks are lost. As a matter of fact, candidates cannot even address the "efficiency" of a story (but they could speak about the efficiency of an algorithm!). Only very few students addressed the efficiency aspect.

Recommendations and guidance for the teaching of future candidates

Please focus on the use of correct terminology and concepts: we are interested in computer science and not in the sociology of the use of IT devices. It is in this spirit that we write question and Markschemes.

Do not use storytelling ONLY: it badly influences the weakest or confused candidates. It is preferable to do less but well than doing a lot badly.

Pseudocode exists to be used. When pseudocode is explicitly asked in some question, it is because this is the maximal level of abstraction that would allow the examiners to assess certain competencies in programming and algorithmic thinking, ones that can be turned into valuable marks.

The exam paper is written in a way to allow for some questions to be answered in a descriptive way and others to be answered with pseudocode.

Please keep teaching well, staying on the technical/scientific side, and ensure students can apply the concepts they learn: in objective 3 questions there were too many answers that were showing a plain regurgitation of notions as if they were quizzes.

Attending workshops may be a valuable experience to quickly clarify in which directions and how to strengthen and consolidate some competencies.

Paper two

Higher level grade boundaries

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 10	11 – 20	21 – 27	28 – 32	33 – 36	37 – 41	42 - 65

Standard level grade boundaries

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 8	9 – 17	18 – 21	22 – 24	25 – 28	29 – 31	32 - 45

General comments

The areas of the programme and examination which appeared difficult for the candidates

This option requires more than the others the ability to apply the knowledge learned in class. In this respect there are still schools entering candidates who have not had the necessary experience and are consequently unable to offer more than a token effort in providing solutions to the method construction questions.

The areas of the programme and examination in which candidates appeared well prepared

The majority of candidates seemed to be able to use the basic elements used in class construction such as the constructors and accessor/mutator methods.

The strengths and weaknesses of the candidates in the treatment of individual questions

OOP Option HL

14. Both (a) and (b) required knowledge that comes more easily to those students who have extensively experimented with OOP programming and are able to see the direct result of the use of OOP features and the effect of altering the modifiers such as “protected”. Consequently the answers here reflected that level of involvement provided each school, with a mixture of

very clear answers and ones which showed a lack of understanding. The majority of students did recognize at least that polymorphism was the feature involved in part (a).

13. (a) and (b) were answered well. Candidates should realize that id fields would not normally be declared as numeric unless they are to be used in calculations, which is unlikely.

Questions such as part (c) are generally introduced to allow students to gain an insight into the classes involved in the paper and to pave the way for the constructions questions that follow. There were quite a few full marks here.

16. Not all of the students were able to follow the correct logic here and exclude the weather related delays. Most were able to construct the basic method even if this exclusion was missed out. It is noticeable that each year more students are correctly identifying both the appropriate way of accessing/assigning values to the object attributes (accessor and mutator methods) and the difference between the representation of methods and variables. Although the OOP option is not intended to be a pure Java course, these are basic features that must be understood.

17(a).This was reasonably well answered although the +/- identifiers were often missing. The important fact about these diagrams is that they are a short-hand way of accurately representing the design of a class.

The method requested in part (b) was a good test of a student's ability to think out the solution to a more complicated problem. Quite a few candidates not so proficient in programming failed to realize that this method (as well as the one in the previous question) was an object method and not a method called from the main class that searched through all objects.

18. Some students provided standard definition type answers here which did not take into account that context in which the question was set.

Part (b) was poorly answered with few understanding the correct relationships between the classes. Either the correct phrase (e.g. "has a") or the correct arrow must be used to show the relationship.

Similarly the responses to the use of dependencies showed some understanding but with only a few students demonstrating complete understanding by providing a suitable example.

Part (c) was answered well.

19. This was the first HL question. It was notable that the HL section was not answered nearly as well as the SL one.

The static nature of arrays was generally recognized but not many provided examples from the classes that highlighted the problems generated by this. The use of a binary tree as a solution in part (b) was surprisingly not identified by the majority. Students should always look to the number of marks awarded. The five marks allotted to this question suggested that a detailed response was required.

20. This question seemed to catch many by surprise. Although they will have used libraries many clearly don't completely understood the reason why. Both (b) and (c) showed a lack of

preparation regarding the use of ArrayLists. Students are expected to be able to appreciate the difference between the use of libraries and the writing of code from basic principles. Not many were able to correctly incorporate the ArrayList methods.

Recommendations and guidance for the teaching of future candidates

The main recommendation is the same as always: that students experiment with use of classes and the OOP features through extensive practical work. As shown by the IA, too many are able to produce solutions without being able to demonstrate a clear understanding of the principles involves. At every stage of the programming course students should be able to explain exactly what is happening and not just be able to mimic solutions presented to them in class. This weakness shows itself in many of the answers to this paper.

HL students must be able to comfortably work with and without libraries. This implies than they can both program a linked list solution from basic principles and also by making use of the LinkedList library. Similarly with arrays and ArrayLists.

Higher level paper three

Component grade boundaries

Grade:	1	2	3	4	5	6	7
Mark range:	0 – 4	5 – 8	9 – 10	11 – 13	14 – 16	17 – 19	20 - 30

The areas of the programme and examination which appeared difficult for the candidates

This remains unchanged from previous years. It is the depth of understanding required in this exam – particularly for questions 3 and 4 – that is not being shown by many candidates

The areas of the programme and examination in which candidates appeared well prepared

The majority of candidates seem to be able to describe the various glossary terms.

The strengths and weaknesses of the candidates in the treatment of individual questions

Question 1

(a)This turned out to be a more difficult start than normal with a lot of very general answers dealing with the amount of light in the scene. This does illustrate the level of understanding that is required through the paper.

(b)This was better-answered than the previous although some examples referred to 2D instead of 3D.

Question 2

(a)Quote a few students are learning the various terms without linking them together (a mind map early on in the course would help in this respect). This was shown by the ability to define both Mocap and avars in 3D models separately but without clearly appreciating how they are related.

(b)This wasn't well answered. Some had memorized a definition for this but definitions learnt this way do not always demonstrate understanding. The important point here is that the algorithms determine the various joint angles in order to arrive at a specified end point.

Question 3

There were many complete answers here and also many others who at least appreciated that many rays (starting from the light source) would not be involved in the final rendering of the scene. What was not so clear was that the exact number of rays (equal to the number of pixels) would be known when tracing from the eye to the light source.

Question 4

Each year this question gives a clear picture of the extent to which the schools are preparing their students for this paper. The question requires not only the demonstration of knowledge and understanding but the ability to answer in context. In this session the context involved a small company (limited resources), a short commercial (no real time rendering or interaction from the user required) and three specified requirements that could be achieved in different ways.

The whole range of marks from 0-12 was achieved with the better responses continually linking their technical knowledge of the algorithms and processes to the defined context.

There were clearly a couple of schools who had sent their students into this exam with minimum preparation. Before a school makes the decision to enter students for HL they need to seriously consider whether they are able to provide both the time and resources that are needed to prepare the students adequately.

Recommendations and guidance for the teaching of future candidates

Schools that have accessed previous reports will find these recommendations familiar. There are new schools taking on computer science each session and for those the temptation must be to leave this component until the end while they concentrate on the components that are, perhaps, more prescriptive. However, this approach will inevitably lead to a failure on the students' part to provide more than superficial responses to questions which demand a reasonable level of understanding. It is clear from assessing this latest paper that the more experienced schools have recognized that each case study and its respective exams have a similar structure which in turn allows for a similar approach to be taken from year to year.

This approach should be based around a 12-month plan and be one that at the very least identifies all the computer science aspects within the case study allowing for a programme to be drawn up, part taught part student researched, that will cover these aspects at such a depth that will provide the students with enough understanding to do themselves justice when taking this exam.

Students can then be further encouraged to research around the topics in order to build up a solid layer of knowledge that will, in particular, aid their discussion in Question 4.