GCE

Computer Science

Advanced GCE **H446**

OCR Report to Centres June 2017

Oxford Cambridge and RSA Examinations

OCR (Oxford Cambridge and RSA) is a leading UK awarding body, providing a wide range of qualifications to meet the needs of candidates of all ages and abilities. OCR qualifications include AS/A Levels, Diplomas, GCSEs, Cambridge Nationals, Cambridge Technicals, Functional Skills, Key Skills, Entry Level qualifications, NVQs and vocational qualifications in areas such as IT, business, languages, teaching/training, administration and secretarial skills.

It is also responsible for developing new specifications to meet national requirements and the needs of students and teachers. OCR is a not-for-profit organisation; any surplus made is invested back into the establishment to help towards the development of qualifications and support, which keep pace with the changing needs of today's society.

This report on the examination provides information on the performance of candidates which it is hoped will be useful to teachers in their preparation of candidates for future examinations. It is intended to be constructive and informative and to promote better understanding of the specification content, of the operation of the scheme of assessment and of the application of assessment criteria.

Reports should be read in conjunction with the published question papers and mark schemes for the examination.

OCR will not enter into any discussion or correspondence in connection with this report.

CONTENTS

Advanced GCE Computer Science (H446)

OCR REPORT TO CENTRES

H446/01 Computer systems

General Comments:

In general, candidate responses demonstrated subject knowledge appropriate to the specification. The majority of candidates were evidently well prepared for the rigour of the examination. Some candidates found questions challenging when they were required to write CSS, SQL and programming statements.

The presentation of work was generally good. Candidates' handwriting on some scripts was difficult to read. Centres should make candidates aware that they may not gain credit for creditworthy responses if their handwriting is illegible.

Comments on Individual Questions:

Question	Comment
1a)	Some candidates did not apply the use of GPU to the scenario. Those who did, generally gained full marks for this question.
1bi)	Candidates who correctly cited paging and segmentation as the methods of dividing memory, invariably went on to achieve full marks.
1bii)	Most candidates achieved both marks on this question. Those who did not, either explained multi-tasking **or** gave appropriate examples. The question asked for both.
1ci)	Many candidates offered advantages and disadvantages of networks in general as opposed to those of a client-server over a peer to peer setup.
1cii)	To achieve this mark, candidates were required to show an understanding that firewalls monitor traffic going to and from a network, many only discussed one-way traffic.
1ciii)	Most candidates gave 'to stop malicious attacks' which was awarded as an interpretation of 'to protect company data'.
2ai)	Surprisingly few candidates achieved full marks on this question. Many received some marks but in general responses lacked detail. Centres should advise candidates that the number of marks awarded for questions gives an indication of the number of different points required in the response.
2aii)	Those candidates who scored well in 2ai) went on to achieve at least some of the marks here. Many candidates found it challenging to clearly explain how the linked list was manipulated. If the question states that 'you may use the diagram to illustrate your answer', centres should encourage candidates to do so.
2b)	Most candidates gained some credit on this question by explaining why hash tables are better suited than linked lists for searching. Those who did not gain credit described in some detail how hash tables were structured, but did not apply their response to the scenario.
3a)	This question was well received by most candidates, invariably scoring most marks.
3b)	Candidates were assessed on the quality of their extended response in this question. Most candidates could describe each of the given types of compression appropriately, with many applying them to the scenario. Many candidates correctly concluded that dictionary encoding was the most appropriate in this case, but few then went on to give clear and appropriate justification for their assertion. In general, most candidates scored well on this question.

Question	Comment
4 a, bi, bii)	In general, most candidates achieved all of the available marks in these questions.
4c)	Some candidates lost credit on this question by failing to mention encryption.
5ai)	Surprisingly few candidates gained full marks on this question. Many responses did not use appropriate assembly language terminology e.g. label, memory location.
5aii)	Very few candidates did not gain full marks on this question.
5bi, bii)	Most candidates identified correctly, an instruction which changed the value in the Accumulator but fewer correctly identified an instruction which changed the value in the Program Counter.
5biii, biv)	Candidates invariably gave both correct output values.
6a,b)	Again, these questions were very well received by candidates with most scoring full marks.
6c)	Generally most candidates stated that two bit shifts were required but some went on to state the incorrect direction i.e. left.
6d)	Candidates whose solution was presented in a logical manner tended to score at least 4 marks on this question. Candidates used different methods to find the solution, all of which were accepted (provided the logic of the calculation could be followed). Centres should advise candidates to present the layout of their responses to this type of question in a logical manner.
7a)	Well received and answered by most candidates.
7b)	Many candidate achieved the mark in part i) few achieved both marks in part ii) mostly stating as opposed to describing the advantage e.g. 'those who gain unauthorised access cannot access passwords' without going on to say 'hash functions are one way'.
7c, d)	In most cases, candidates who achieved marks in c) went on to achieve marks in d) with few candidates achieving all marks in either. Many candidates did not use correct SQL statement structure or syntax e.g. confusing attribute names with string literals.
7e)	Candidates were asked to complete a function in this question. Although many students demonstrated reasonable logic in solving this problem, some used output statements rather than returned values from the function, therefore, not gaining full marks.
8)	Candidates were assessed on the quality of their extended response in this question. Many candidates offered a balanced discussion although some of the examples used did not demonstrate that the candidate understood the difference between AI and robotic automation. Conclusions were often not fully justified/ reasoned. Many candidates scored in the mid-level band on this question.
9a)	Those candidates who achieved credit on this question, generally achieved both marks.
9b)	Few candidates scored more than two marks on this question. There was a general lack of attention to detail resulting in fundamental mistakes e.g. missing close bracket }; equals (=) instead of colon (:) when setting attributes.
9c)	Well attempted by most candidates with many scoring two out of a possible three marks. Some candidates did not gain credit in iii) because they cited *'…time taken to process'* as a disadvantage.
10a)	Most candidates gained some marks on this question but few achieved full marks. In general responses lacked attention to detail and clarity of expression. Centres should advise candidates that the number of marks awarded for questions gives an

	indication of the number of points required in the response.
Question	**Comment**
10b)	Many candidates achieved some of the available marks on this question for attempting to traverse each letter in the word and each letter in the random word - a loop with a nested loop. Some achieved more marks for comparing the current letters and outputting the length of the valid word. Fewer candidates achieved the final marks for checking if the letter was in the word or duplicated.
10c)	Very few candidates did not achieve this mark, most correctly stating the advantage '*faster to search'*.
10di)	Those candidates who cited generic advantages of using subroutines as opposed to library routines did not gain credit. The question asked for advantages to the team of using a library.
10dii)	Candidates were assessed on the quality of their extended response in this question. Many candidates explained the stages of compilation very well. Some went on to describe how code from the library becomes part of the finished program equally well. Few justified why each stage was necessary. Many candidates scored well on this question.
11a, b)	Most candidates scored well on these questions demonstrating their understanding of logic gate circuits. Some candidates simplified the circuit in part b) which achieved full marks provided the resultant circuit gave the same output.

H446/02 Algorithms and programming

General Comments:

The paper differentiated the candidates effectively and scripts included some very strong candidate responses.

Questions that targeted Knowledge and Understanding required candidates to have studied the whole specification and to have learnt the relevant definitions. Some candidates had not been prepared by covering the whole specification and thus failed to achieve marking points targeted at lower grades for basic recall.

Questions targeting Application required higher order skills to be able to use knowledge gained in context to solve problems. There was clear differentiation between candidates who understood the concepts and who could apply them, and those who displayed little ability to apply what they had learnt.

A number of candidates struggled to write pseudocode. Structured English is insufficient for examination questions that specifically require pseudocode to be written. Candidates are not required to write pseudocode to the standard presented in the specification, but would benefit from doing so.

Comments on Individual Questions:

Question	Comment
1ai	Many candidates found it difficult to apply the logic required to calculate the correct solution. Stronger candidates could do so even if they did not know the algorithm for insertion sort.
1aii	Some candidates confused insertion sort with other sorting algorithms, but many candidates gave good answers in diagrammatic form. Answers in diagrammatic form after each pass of the loop were often far clearer than prose descriptions. This form of answer should be encouraged.
1b	Whilst many candidates had some knowledge of ‘Big O’ notation fewer could apply it correctly within the context given.
1c	Most candidates achieved some credit, especially for a description of the bubble sort. Fewer candidates could compare the relative merits of both bubble and insertion sort in terms of the best / average / worse case.
2a	Many candidates struggled to apply the context given to computer science concepts and hence answer with the relevant properties of a linked list that would be relevant in context.
2b	Most candidates scored well in part (i), but fewer understood how the pointers in a linked list could be updated in part (ii) to allow the insertion of the new item in the next free space.
2c	Few candidates could give a clear answer in part (i) using the correct technical vocabulary that the array index/subscript could be used as the node number. Many candidates could work through the logic required in the trace table in part (ii), but fewer could actually explain what it was doing in (iii) within the context of the scenario. Part (iv) was often best answered by those candidates who used the diagram to give the solution. Candidates should be encouraged to use diagrams where they can be used to good effect rather than lengthy or vague prose descriptions.

Question	Comment
2d	Many candidates correctly identified a linear search and could justify the need for it. However, a lot of candidates did answer binary search without appreciating that the data set needed to be in order first.
2e	Most candidates achieved some credit for factual recall. However, weaker candidates often answered debugger rather than explaining the specific features of the debugger which would have been creditworthy.
2f	It was clear that many candidates had not covered the concept of concurrency and how it allows different processes to occur at the same time. Strong candidates appreciated that this could be simulated on a single core with time slicing or implemented within a parallel architecture. Many candidates lost sight of the fact that answers needed to be related to computer science rather than a restaurant chain and could not explain the underlying computer science that would allow a solution to be delivered.
3a	Part (i) was well answered where candidates had read the question stem and thought logically about the steps involved. Many candidates gained some credit in part (ii), but fewer could expand on the points they made to gain full credit.
3b	Many candidates struggled to produce good answers which could have been calculated and did not require factual recall.
3c	Many candidates scored well, but fewer scored full marks. The use of pseudocode rather than Python like syntax would have prevented errors with loop lengths.
3d	Most candidates scored some of the marks, but fewer appreciated that the characters needed to be popped from the stack initially, and that the converted characters would have to be concatenated into a string at the end of the process.
4	A number of candidates incorrectly identified the data structure as a binary tree which indicated that this was the only type of tree that they were familiar with. Descriptions of depth and breadth first traversals were often very vague, and precision in terms of the algorithmic steps involved would have produced stronger answers. A pleasing number of candidates produced the correct traversal of the tree, but of those, a number did not appreciate that the node was only output when it was popped from the stack, and hence missed location G before X was actually output.
5	Many candidates scored well in parts (a) and (b) and it was pleasing to see that recursion could both be identified and traced. Few candidates achieved full marks in part (b) because they did not appreciate that the function was inside a print statement so a final output of 2 would be produced after the value 2 was returned.
5c	Most candidates produced recognisable pseudocode. Weaker candidates produced logically incorrect solutions or did not understand the difference between an iterative and a recursive solution – reformulating another recursive solution. Where strong candidates produced good solutions they sometimes forget the necessity to have a temporary swap variable when swapping the values in two different variables over.
6a	Nearly all candidates achieved full marks after analysing the requirements in the stem of the question.
6bi	Nearly all candidates scored full marks for factual recall of the required definition.
6bii	Nearly all candidates achieved three or more marks after analysing the requirements in the stem of the question. A number gave incorrect multiplying factors for some of the required elements and thus lost marks where mathematical accuracy was required.

Question	Comment
6ci/ii	Most candidates scored some credit, but a disappointing number did not give a correct procedural declaration. The correct mathematical expression to increase the *intelligence* by 0.6% in (ii) was often incorrectly given.
6d	Many candidates struggled with the application of object oriented techniques and concepts and it was clear that many of these candidates had not had practical experience of object oriented programming. Stronger candidates did perform well and understood how to create instances from classes and how to use inheritance.
6e	Abstraction was well understood by the majority of candidates. Candidates needed to be able to give relevant examples in context and to be able to evaluate the advantages that abstraction gave to achieve marks in the top band. The level of clarity and analysis required for the top band was only seen in the strongest candidates' responses.
6f	Most candidates scored well for this section.

H446/03 & H446/04 Project

General Comments:

Many Centres submitted work in an electronic format, as PDFs, either via the Repository or on DVD/Memory stick via the postal route. This not only makes the moderation process simpler but often shows the work at its best; screenshots being much clearer when viewed on a screen rather than when printed.

Candidates produced a wide range of systems using a variety of programming languages. The best projects tended to be the more ambitious. Candidates need to be cautious not to choose too simplistic projects. They need to have enough scope to develop their system over a number of iterations. There needs to be enough substance to the system to have enough to demonstrate skills related to design, coding and testing. Simples quizzes, revision programs and trivial data storage systems by their nature tended not offer these opportunities. A project a candidate can complete without encountering any challenges will give them little to write about. Conversely a candidate can still receive very high marks for an ambitious project that doesn't achieve all its aims but offers plenty of opportunity discussing approaches taken.

Websites have the potential to make excellent projects and a number of good quality examples incorporating databases with server and client side processing were seen. There were, however, examples of websites that offered little opportunity to access development marks. These tended to be largely static sites with just a trivial coding element such as a simple login page and small amount of, relatively generic JavaScript.

Projects at the higher mark bands had a suitable user interface. In most cases this the most suitable interface was a GUI. (There are of course exceptions. One of the most impressive projects seen this session used a command line interface, but as this was a mini operating system coded in x86 assembly it was deemed acceptable!)

Games proved a popular option, often through Unity, or Python with PyGame. They offer a lot of opportunity to demonstrate very high level skills and some excellent projects were submitted. Care must be taken that all the 'hard work' isn't being done by the underlying framework and the candidate has built sufficiently on top of it.

Comments on Individual Questions:

Analysis

Justifying why a problem was suitable for computational methods caused some confusion. At the top end, candidates gave clear rationale but a good number were unclear on what was expected. This section requires candidates to reflect on why their problem/system is suitable to be solved with a computer rather than using alternative methods. This may involve discussions on speed, accuracy, removal of the need for other people, interactivity and such like.

A number of candidates opted to carry out interviews as part of their analysis. These are not essential but can be useful when solving a problem specific to the stakeholders. A lot of effort was spent typing out interview transcripts. This is an unnecessary use of candidates' time and a summary of key points taken from the interview is sufficient.

There was a significant range of approaches to the research of existing solutions. At the top end candidates had taken time to gain a significant insight into these solutions. Where possible they had used them, where not, they had reviewed documentation and watched videos on their operation. This meant they were easily able to distil and reference the key aspects of the systems they wished to replicate in their own objectives. Good quality research naturally lent itself to good quality objectives.

Where candidates had only vague objectives ("My system must be easy to use.", "My game must be fun to play", "My system should load quickly." etc) they tended to not only drop marks in the analysis but also later on. Time spent on good quality, measurable objectives pays dividends further on in the project as they will form the backbone of strong testing and evaluation sections.

Design

Many candidates used top down design diagrams and variants of JSP diagrams to decompose their system. This, on the whole, was done well. There was more variance in how well the algorithms for the decomposed elements were shown.

There was a trend for some candidates to not show their algorithms in enough detail. At the top end there should be a clear decomposition of the problem into key elements and algorithms to match each of these. Where algorithms require further investigation it is acceptable to design them at a high level initially refine them in more depth later on, in the development. Reverse engineered code was given no credit.

It is quite acceptable for a project to deviate from its initial design and this is something candidates can discuss during their development. There is nothing to stop candidates adding detail to their designs at later stages. Some excellent projects were seen where the design was integrated into the development rather than as a discrete section.

Developing the Coded Solution

Some candidates had a better grasp of the iterative process than others. Those scoring lower marks tended to present their project in a linear fashion. At the top end candidates had clear and discrete iterations. Each iteration had a stated set of objectives which were tested and evaluated at the iteration's end.

Prototyping proved a challenge. It is recognised that some iterations lend themselves to prototypes more easily than others and leeway was given as to how candidates interpreted a prototype at each iteration.

Many candidates had developed good coding habits presenting easy to follow code with sensible variable names and helpful commenting.

Testing to Inform Development

Approaches to this section were variable. There is an expectation that candidates demonstrate they have tested throughout the development process and used this to help evaluate each iteration and drive on the next. There is no need to exhaustively demonstrate each test but there should be enough evidence to convince the reader substantial testing took place. The best projects put the most focus on the more unusual problems that cropped up during developmental testing (getting libraries to work, nuances of the programming language, issues with file types and such like) rather than trivial syntax errors.

Testing to Inform Evaluation

It is important that evidence is provided of tests. Where there are a number of similar tests it is not necessary to provide a screenshot of all of them but there should be sufficient evidence to demonstrate the extent to which the system functions. The best projects gave thorough evidence of testing for robustness.

A number of candidates provided video evidence of their testing giving appropriate time codes for each testing their documentation. This worked extremely well and left the moderator in no doubt the system worked as claimed.

Evaluation

A number of candidates struggled to provide evidence of accomplishing their objectives. Candidates should aim to take each objective decided upon in their analysis, cross reference it with the tests they have carried out and discus how successfully the objective has been met. Showing chunks of code is not in itself evidence of an objective being met.

OCR (Oxford Cambridge and RSA Examinations)
1 Hills Road
Cambridge
CB1 2EU

OCR Customer Contact Centre

Education and Learning
Telephone: 01223 553998
Facsimile: 01223 552627
Email: general.qualifications@ocr.org.uk

www.ocr.org.uk

For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored

Oxford Cambridge and RSA Examinations
is a Company Limited by Guarantee
Registered in England
Registered Office; 1 Hills Road, Cambridge, CB1 2EU
Registered Company Number: 3484466
OCR is an exempt Charity

OCR (Oxford Cambridge and RSA Examinations)
Head office
Telephone: 01223 552552
Facsimile: 01223 552553