GCSE EXAMINERS' REPORTS

GCSE (LEGACY)
COMPUTER SCIENCE

SUMMER 2018
Grade boundary information for this subject is available on the WJEC public website at: https://www.wjecservices.co.uk/MarkToUMS/default.aspx?l=en

**Online Results Analysis**

WJEC provides information to examination centres via the WJEC secure website. This is restricted to centre staff only. Access is granted to centre staff by the Examinations Officer at the centre.

**Annual Statistical Report**

The annual Statistical Report (issued in the second half of the Autumn Term) gives overall outcomes of all examinations administered by WJEC.

<table>
<thead>
<tr>
<th>Unit</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit 1</td>
<td>1</td>
</tr>
<tr>
<td>Unit 2</td>
<td>3</td>
</tr>
<tr>
<td>Unit 3</td>
<td>5</td>
</tr>
</tbody>
</table>
General Comments

This report is produced to support teachers by indicating where responses were generally good or poor and to indicate why candidates were not gaining marks on specific questions. The report should be used in conjunction with the question paper, mark scheme and item level data.

The wording of questions is designed to suggest the level of detail required, by the use of words or phrases like name, describe, describe in detail, explain or discuss, but it is very important that candidates also pay attention to the maximum number of marks available in each part question, as stated in the rubric for each paper.

The standard of language used was generally very good. Candidates should be reminded that legible handwriting is important throughout the whole paper. Examiners are not prepared to guess the word: if it is not clear, then no mark can be awarded.

Most candidates demonstrated that they understood and could answer questions on most of the specification. Many candidates were obviously well prepared for this exam and excellent answers were evident with numerous candidates giving extended answers where appropriate. However it was clear that some candidates were not prepared to answer all questions.

1. Most candidates were able to rank order storage capacity.
2. Most candidates were able to state that ROM is read only memory and is non volatile, however fewer were able to give an example of its use.
3. Generally well answered as candidates were able to state the purpose of each component of the CPU and the benefit of a dual core CPU. Fewer were able to name the two main buses that are used by the CPU.
4. Generally this question was not well answered. Many candidates were unable to name the two protocols that allow packets to be sent and received. A large majority were unable to describe the contents of a packet and describe packet switching.
5. Generally well answered however some candidates gave an explanation of the data type and did not provide suitable examples of data that could be held in each data type given.
6. This question was not well answered as many candidates were not able to state the number of bits that are used or explain the limitations of ASCII.
7. Generally well answered as candidates were able to calculate compression ratios.
8. Well answered by some candidates however it was disappointing that far too many candidates were unable to explain how each topology met the given requirement.
9. Answered well by most candidates.
10. Generally well answered as candidates were able to convert between denary, binary and hexadecimal. Fewer candidates were able to explain why hexadecimal numbers are used in computer science.

11. Well answered by some candidates however it was disappointing that far too many candidates were unable to identify typical functions of a firewall.

12. This question was not well answered. Many candidates were not able to explain how the operating system manages printing, the processor or RAM.

13. Most candidates were able to identify the syntax error and correct the code. Fewer were able to identify the logic error.

14. Some very good responses were evident as candidates were able to identify five tools provided by an IDE. They were able to use their theoretical and practical knowledge to explain how the tools are used during the development of a computer program. Many candidates however were unable to identify more than one tool and a significant minority failed to attempt the question.
COMPUTER SCIENCE
GCSE (NEW)
Summer 2018
UNIT 2: SOLVING PROBLEMS USING COMPUTERS

General Comments

There was a new method for submitting candidate scripts to the WJEC (using SecureAssess) and this largely worked well. Candidates generally prepared the electronic media for delivery to examiners well. However, a small number did have local technical issues. Centres are reminded that they should test that the examination accounts can properly run Greenfoot / Java before the examination. The vast majority of centres had no issues with the on-screen nature of this examination and any feedback via the subject officer would be welcome.

Good answers were seen for the HTML and Greenfoot components, the responses seen for the algorithm were much improved on previous years. Those candidates who gained full marks in the question produced very elegant solutions.

Unit 2: Comments on Individual Questions

1. This was answered well by most candidates. However, a significant number of candidates found the link (using <a href= . . . ) difficult. Candidates also found the <p> tag difficult. This was the first time this was assessed and contributed to the slightly increased difficulty of the paper this year. Some candidates also forgot to close the HTML tags resulting in incorrect formatting. Candidates should not use generated code or CSS, the specification states the tags that should be used.

2. Some good answers were seen to this question, however, only a few candidates had a fully working algorithm. Within the question marks were awarded for sections – such as inputting data, loops, conditions (if statements) and output. Of these sections most candidates had input and some outputs. Some candidates had a correct if statement. Very few candidates had a working loop.

3. Responses here were highly variable. A very small number of candidates did not save the file as the required name. Populating the world proved problematic for a small number of candidates. Most candidates could get an object to move around the screen. A significant number could get an object to respond to key input. The method for implementing movement on key press varied by centre and all working solutions were awarded credit. Some candidates could implement collision detection and trigger a sound. Very few candidates could implement the counter which is seen as an A/A* differentiator. The code written was generally of a very good standard demonstrating a good understanding of objects, methods and properties within Java. The WJEC emphasises that centres should ensure that the machines on which this
4. online examination is to be taken are tested to ensure their functioning status before the examination begins. We would also like to stress that candidates should sit the exam using the version of Greenfoot that the WJEC specifies. A very small number of centres reported that they had used a different version of Greenfoot in the exam as this was used during teaching. This is a centre based issue and should be prevented at all times as candidates will be disadvantaged by using the wrong version.
General Comments

Examples of good work were seen at moderation this summer. As usual, two scenarios had been made available to the candidates. The scenarios were of similar demand and each allowed candidates to gain access to the full range of marks. Candidates' outcomes were not affected by their choice of scenario. In most cases centres made use of one or the other scenario although there were a few centres that submitted work from both scenarios. It should be noted that it is essential that candidates undertake the correct scenario for the current series.

Requirements of the scenarios.

Each scenario has a bullet pointed list of requirements. To access full marks for the implementation of the solution to the given problem all bullet points should be covered. However, many candidates were not able to produce a solution that covered all bullet points for their chosen scenario.

For the ‘Check it out’ scenario candidates were asked to:

- Allow the user to enter the customer’s details: name, postcode and loyalty card details
- Check if the card has expired
- Check the loyalty card number is valid by:
  - Allowing the user to enter the 8 digits shown on the front of the card
  - Removing the 8th digit and storing it as a ‘check_digit’
  - Reversing the numbers
  - Multiplying the 1st, 3rd, 5th and 7th digits by 2
    - If the result of the multiplication is greater than 9 then subtract 9 from the result
  - Adding together the resulting 7 digits
  - Checking if the sum of the added digits plus the ‘check_digit’ is divisible by 10
- Output whether the loyalty card is valid or not
- Output customer and loyalty card details.

Most candidates could create an interface that allowed users to enter the card details. The majority of candidates allowed users to enter the customers' personal details.

Many candidates were able to produce code that could carry out the required calculations. Candidates also created validation routines. However, these routines often caused issues when running the code as the interfaces did not provide clear enough instructions for the user to be able to enter data in the correct format.
For the ‘Guess the word’ scenario candidates were asked to:

- Select, at random, a word of six characters from a list
- Allow the user to input a character
- Check if the character input is in the word and how many times it occurs
- Limit the number of guesses allowed
- If the character is in the word, display the character(s) on the screen in the correct positions
- If the character is not in the word adjust the number of guesses remaining
- If the player is not successful display the word on the screen

Most candidates were able to provide an interface that allowed the user to enter guessed letters. Many provided information on whether or not the letter was included in the word and on the number of guesses remaining. Some better candidates provided information on the letters already guessed to avoid the user guessing the same letters repeatedly.

Requirements for the Report

The specification states that the candidates should produce a report that:

- Describes the design, implementation and testing of their solution e.g.
  - Sequencing
  - choice (if - then - else)
  - iteration (loops)
  - language constructs that support abstraction (typically a ‘procedure’ or ‘function’ with parameters)
  - some form of interaction with the program’s environment (input/output or event-based)
- Finds and correct errors in their code
- Reflects thoughtfully on their program, including assessing its correctness and fitness for purpose; understanding its efficiency; and describing the system to others.

Design of Solution

Most candidates were able to carry out the required analysis and many were able to outline the objectives for their solution to the given problem.

In a significant minority of instances, candidates were neither able to justify their choice of programming language using appropriate technical terminology nor relate the features and facilities of the language to their proposed method of solution.

Many candidates were able to describe some of the process stages required for their solutions in pseudo code and/or flowcharts. However, fewer candidates covered all processing stages for their proposed solution. In several cases it was not clear that this work had been completed before implementation. Retrospective designs will not be given any credit at moderation.

Most candidates were able to describe some type of testing strategy and some evaluation criteria. In future candidates may benefit from considering their objectives when describing their testing strategies and evaluation criteria ensuring that they plan to test and evaluate against each objective.
Program Documentation

Many candidates were able to demonstrate at least a reasonable understanding of the tools and techniques they had used through annotation. However, in some instances candidates struggled to use the correct technical terminology to describe the constructs of their programs.

Testing

Most candidates were able to design tests that would demonstrate the functioning of parts of their solution. However, some test plans would have benefited from focusing on the logic of the solution rather than repeatedly testing the less complex parts of the system.

In future candidates would benefit from using their objectives and success criteria as a framework for their test plans and ensure that these are met by their solution.

Where solutions carry out calculations candidates should ensure that the data entered produces the correct result and the output is correctly formatted. Both scenarios provided many opportunities for candidates to test the logic of their solutions using mathematical data.

Evaluation

Many candidates did not appear to have realised the importance or the demands of this section of work. It should be noted that 12 of the 50 marks are awarded for the review section of the work. This being almost a quarter of the total marks an extended piece of writing was required to cover the requirements of the specification.

The specification calls for an evaluation that allows the candidate to: Reflect thoughtfully on their program, including assessing its correctness and fitness for purpose; understanding its efficiency; and describing the system to others.

Many of the evaluations produced by the candidates were brief and tended to be narrative rather than reflective and evaluative in nature.

In future candidates would benefit from assessing their work against their objectives and success criteria to ensure that their discussion is correctly structured.

Few candidates offered valid and detailed suggestions for future improvements. However, a minority were able to discuss their solutions in light of their structure and suggest viable improvements that could be created using their chosen language.