 

GCE A LEVEL MARKING SCHEME

**AUTUMN 2021**

**A LEVEL**

**COMPUTER SCIENCE - COMPONENT 1 A500U10-1**

# INTRODUCTION

This marking scheme was used by WJEC for the 2021 examination. It was finalised after detailed discussion at examiners' conferences by all the examiners involved in the assessment. The conference was held shortly after the paper was taken so that reference could be made to the full range of candidates' responses, with photocopied scripts forming the basis of discussion. The aim of the conference was to ensure that the marking scheme was interpreted and applied in the same way by all examiners.

It is hoped that this information will be of assistance to centres but it is recognised at the same time that, without the benefit of participation in the examiners' conference, teachers may have different views on certain matters of detail or interpretation.

WJEC regrets that it cannot enter into any discussion or correspondence about this marking scheme.

# EDUQAS GCE A LEVEL COMPUTER SCIENCE AUTUMN 2021 MARK SCHEME



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 1. (a) | **One mark for each correct point up to a maximum of** |  |  |  |  |  |
|  | **four** |  |  |  |
|  | This linked list is a dynamic data structure as it can grow | 1 |  |  |
|  | and shrink in size after declaration. |  |  |  |
|  | Each element in this linked list is known as a node, the | 1 |  |  |
|  | first element is the head node. |  |  |  |
|  | Each node consisted of the data itself and the | 1 |  |  |
|  | address/reference to the next node. |  |  |  |
|  | The last node 95 references null. | 1 |  |  |
|  | This linked list uses more memory than an array as it | 1 |  |  |
|  | needs to store the address/reference of the next node in |  |  |  |
|  | addition to the data. |  |  |  |
|  | A node in this linked list cannot be directly accessed and | 1 |  |  |
|  | each node needs to be traversed until the correct node is |  |  |  |
|  | accessed / sequential access. |  |  |  |
|  |  |  | 2a | 4 |
| (b) | 41 32 73 84 95 55  Award 1 mark for node in correct position  Award 1 mark for arrow to and from correct node. |  |  |  |  |  |
|  | 1 |  |  |
|  | 1 |  |  |
|  |  | 2a | 2 |
| (c) | 41 32 84 95 55  Award 1 mark for node removed  Award 1 mark for arrow to and from correct node. |  |  |  |  |  |
|  | 1 |  |  |
|  | 1 |  |  |
|  |  | 2a | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 2. | **1 mark for each criterion and 1 mark for each** |  |  |  |  |  |
|  | **description.** |  |  |  |
|  | **Max of 8 marks** |  |  |  |
|  | Requirements - evaluate the solution against the original |  |  |  |
|  | requirements. All requirements should be met for a |  |  |  |
|  | solution to be successful. |  |  |  |
|  | Cost – evaluate the solution against costs which include | 2 |  |  |
|  | financial costs, human costs and resource costs. A |  |  |  |
|  | solution must not exceed any negotiated costs to be |  |  |  |
|  | successful. |  |  |  |
|  | Robustness - evaluate the solutions against its test | 2 |  |  |
|  | results. A solution should use error trapping and validation |  |  |  |
|  | methods to be successfully robust and reduce the chance |  |  |  |
|  | of system errors and failures. |  |  |  |
|  | Usability – evaluate the solution against the ease of use | 2 |  |  |
|  | for the end user. A solution should use an intuitive user |  |  |  |
|  | interface suitable for the end user to be successful. |  |  |  |
|  | Performance - evaluate the performance of the solution, it | 2 |  |  |
|  | should be fully optimised to reduce memory usage. A |  |  |  |
|  | solution should complete specific task within a given time |  |  |  |
|  | frame to be successful. |  |  |  |
|  |  |  | 1b | 8 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 3. (a) | 1 mark for identifying the outer loop will execute n times | 1 |  |  |  |  |
|  | 1 mark for identifying inner loop will execute n2 times | 1 |  |  |
|  | 1 mark for correct numbers of operations 6n2 + 2n | 1 |  |  |
|  | 1 mark for determining that the order will be dominated by n2 | 1 |  |  |
|  | 1 mark for determining that the growth rate for time | 1 |  |  |
|  | performance is O(n2) |  |  |  |
|  |  |  | 3c | 5 |
| (b) | The algorithm only uses one data structure, a one- | 1 |  |  |  |  |
|  | dimensional array. Therefore, total storage requirements = |  |  |  |
|  | 1. |  |  |  |
|  | As only one data structure is being used, the growth rate | 1 |  |  |
|  | for memory will be constant O(1). |  |  |  |
|  |  |  | 3c | 2 |
| (c) |  |  |  |  |  |  |
|  | Identifying polynomial complexity | 1 |  |  |
|  | Time axis labelled correctly | 1 |  |  |
|  | Size axis labelled correctly | 1 |  |  |
|  | Correct gradient of line | 1 |  |  |
|  |  |  | 2a | 4 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 4 (a) | Correct answer can be established using different steps / |  |  |  |  |  |
|  | laws / rules / identities / dual relations. |  |  |  |
|  | One possible solution: |  |  |  |
|  | (A + C). ( A(. B) + (C. B) |  |  |  |
|  | (A + C). ( A( + C). B |  |  |  |
|  | C. (A + A(). B |  |  |  |
|  | C. (1). B |  |  |  |
|  | C. B |  |  |  |
|  | B. C |  |  |  |
|  | Correctly applying identities to arrive at correct answer 5 | 5 |  |  |
|  | marks |  |  |  |
|  | Correctly applying identities but arriving at wrong answer 1 | 1 |  |  |
|  | mark maximum 4 |  |  |  |
|  |  |  | 2a | 5 |
| (b) | Correct answer can be established using different steps / laws / rules / identities / dual relations.  One possible solution:  +B++.+A++. (B( + A) . A + C. 1  (B( + A(). (B( + A) . A + C  (B(. B( + B( . A + A(. B( + A(. A). A + C (B( + B( . A + A(. B( + 0). A + C  (B( + B( . A + A(. B(). A + C  (B( + B( .1. B(). A + C B(. A + C  Correctly applying identities to arrive at correct answer 5 marks  Correctly applying identities but arriving at wrong answer 1 mark maximum 4 |  |  |  |  |  |
|  | 5 |  |  |
|  | 1 |  |  |
|  |  | 2a | 5 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 5. | **1 mark for each method.** |  |  |  |  |  |
|  | **1 mark for each reason. Up to a maximum of eight** |  |  |  |
|  | **Indicative content** |  |  |  |
|  | External hard drive or memory sticks can be used to store |  |  |  |
|  | back-up files through copying and pasting. |  |  |  |
|  | This method is useful of backing up small personal data | 1 |  |  |
|  | such as documents and photos. Data can be overwritten | 1 |  |  |
|  | with new backups. |  |  |  |
|  | CD / DVD can be used to store back-up files but are | 1 |  |  |
|  | limited to the amount of space that is available. This | 1 |  |  |
|  | method of back-up can be used for archiving data due to |  |  |  |
|  | its read-only nature. Can be used for archiving files and |  |  |  |
|  | documents. |  |  |  |
|  | Magnetic tapes can be used to store and routinely back- | 1 |  |  |
|  | up network data. This method is used for backing up user | 1 |  |  |
|  | network data and shared data including roaming profiles |  |  |  |
|  | and intranet services. |  |  |  |
|  | A dedicated back-up server could be used to either mirror | 1 |  |  |
|  | a main storage server or routinely back-up network data. | 1 |  |  |
|  | Cloud storage solutions can be used to routinely back-up | 1 |  |  |
|  | data online and allows for synchronous version | 1 |  |  |
|  | management of online files. This method for backup can |  |  |  |
|  | be used to back up personal files that need to be access |  |  |  |
|  | from many different devices. |  |  |  |
|  | **Accepted**: answers related to speed, capacity, portability, |  |  |  |
|  | durability, GFS, incremental etc |  |  |  |
|  |  |  | 1b | 8 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 6. | **One mark for each correct point.** |  |  |  |  |  |
|  | <upper> ::= A|B|C . . . Y|Z |  |  |  |
|  | <lower> ::= a|b|c . . . y|z | 1 |  |  |
|  | <digit> ::= 0|1|2 . . . 8|9 |  |  |  |
|  | <special> ::= $|\_ | 1 |  |  |
|  | <first>::= <lower>|<special> | 1 |  |  |
|  | <letterdigit>::=<upper>|<lower>|<digit>|\_ | 1 |  |  |
|  | <character>::= <letterdigit>|<letterdigit><character> | 1 |  |  |
|  | <identifier>::=<first><lower><character> | 1 |  |  |
|  | Answer not correct if BNF notation used incorrectly. |  |  |  |
|  |  |  | 2a | 6 |
| 7. (a) | **Award up to two marks for a description and one for an example**  **Indicative content**  In-order traversal is applied by visiting the left subtree first, then root and finally the right subtree. This method could be when searching for a file in the file system.  Accept any suitable example | 3 |  | 2a |  | 3 |
| (b) | **Award up to two marks for a description and one for an example**  **Indicative content**  Post-order traversal is applied by visiting the left subtree first, then right subtree and finally the root. This method could be used to delete all files in the file system.  Accept any suitable example. | 3 |  | 2a |  | 3 |
| (c) | **Award up to two marks for a description and one for an example**  **Indicative content**  Pre-order traversal is applied by visiting the root first, then left subtree and finally the right subtree. This method could be used to create a copy files in the file system.  Accept any suitable example. | 3 |  | 2a |  | 3 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 8. (a) | **Award one mark for each of the following up to a** |  |  |  |  |  |
|  | **maximum of two.** |  |  |  |
|  | Analysing an existing system structure can be completed | 1 |  |  |
|  | using flow chart software or UML software. |  |  |  |
|  | These pieces of software allow the developers to produce | 1 |  |  |
|  | complete planning and design documents for individual |  |  |  |
|  | cases such as the end use or another developer. |  |  |  |
|  | Analysis software tools are also available to be used in | 1 |  |  |
|  | requirements engineering and management. This software |  |  |  |
|  | can be used to record and monitor requirements, use and |  |  |  |
|  | test cases. |  |  |  |
|  | Any suitable example of software used in system analysis. | 1 |  |  |
|  |  |  | 1b | 2 |
| (b) | **Award one mark for each of the following up to a** |  |  |  |  |  |
|  | **maximum of two.** |  |  |  |
|  | Designing a system structure can be completed using flow | 1 |  |  |
|  | chart software or UML software. |  |  |  |
|  | UX and UI designers use wireframing and mock-up tools | 1 |  |  |
|  | for user interfaces and experience. |  |  |  |
|  | Collaborative code editors could be used to produce | 1 |  |  |
|  | pseudocode for review by developers. |  |  |  |
|  | Any suitable example of software used in system design. | 1 |  |  |
|  |  |  | 1b | 2 |
| (c) | **Award one mark for each of the following up to a** |  |  |  |  |  |
|  | **maximum of two.** |  |  |  |
|  | Program version management software is store, record | 1 |  |  |
|  | and analyse different stages of code development. |  |  |  |
|  | Version of software can be submitted through a VCS | 1 |  |  |
|  | (version control system) e.g. Git, to an online repository |  |  |  |
|  | hosting solution to track and record the changes in |  |  |  |
|  | projects and files e.g. GitHub. |  |  |  |
|  | Version management software is useful when multiple | 1 |  |  |
|  | developers are working on a single project, it ensures one |  |  |  |
|  | develop does not overwrite another developers code. |  |  |  |
|  | Version management software can also be used to roll- | 1 |  |  |
|  | back software if a program becomes corrupt during the |  |  |  |
|  | development process. |  |  |  |
|  |  |  | 1b | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| (d) | **Award one mark for each of the following up to a** |  |  |  |  |  |
|  | **maximum of two.** |  |  |  |
|  | **Indicative content** |  |  |  |
|  | Quality assurance / control software to test that a solution | 1 |  |  |
|  | conforms to internal and external standards. |  |  |  |
|  | Test environments can be used to test the portability of | 1 |  |  |
|  | software on different platforms such as Linux and |  |  |  |
|  | Windows. |  |  |  |
|  | Version control repository can be used to report, monitor | 1 |  |  |
|  | and analyse code errors, defects and bugs. |  |  |  |
|  | Built-in automated testing features within IDEs, such as | 1 |  |  |
|  | breakpoints, to generate unit and system performance |  |  |  |
|  | testing. |  |  |  |
|  |  |  | 1b | 2 |
| 9. | 110011002  100000002  AND 10000000  State of sign bit is 1 which is negative  Award 1 mark for using an AND or XOR mask. Award 1 mark for correct result  Award 1 mark for determining the state of sign bit. |  |  |  |  |  |
|  | 1 |  |  |
|  | 1 |  |  |
|  | 1 |  |  |
|  |  | 2a | 3 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 10. | **Indicative content** |  |  |  |  |  |
|  | 1 Declare subprocedure QuickSort(myArray is integer, |  |  |  |
|  | indexLow is integer, indexHi is integer) |  |  |  |
|  | 2 |  |  |  |
|  | 3 Declare pivot is integer |  |  |  |
|  | 4 Declare temp is integer |  |  |  |
|  | 5 Declare low is integer |  |  |  |
|  | 6 Declare high is integer |  |  |  |
|  | 7 |  |  |  |
|  | 8 set low = indexLow |  |  |  |
|  | 9 set high = indexHi |  |  |  |
|  | 10 |  |  |  |
|  | 11 set pivot = myArray [(int((indexLow + indexHi)/2))] |  |  |  |
|  | 12 |  |  |  |
|  | 13 while (low <= high) |  |  |  |
|  | 14 |  |  |  |
|  | 15 while (myArray[low] < pivot and low < indexHi) |  |  |  |
|  | 16 set low = low + 1 |  |  |  |
|  | 17 end while |  |  |  |
|  | 18 |  |  |  |
|  | 19 while (pivot < myArray[high] and high > indexLow) |  |  |  |
|  | 20 set high = high – 1 |  |  |  |
|  | 21 end while |  |  |  |
|  | 22 |  |  |  |
|  | 23 if (low <= high) then |  |  |  |
|  | 24 set temp = myArray[low] |  |  |  |
|  | 25 set myArray[low] = myArray[high] |  |  |  |
|  | 26 set myArray[high] = temp |  |  |  |
|  | 27 set low = low + 1 |  |  |  |
|  | 28 set high = high – 1 |  |  |  |
|  | 29 end if |  |  |  |
|  | 30 end while |  |  |  |
|  | 31 |  |  |  |
|  | 32 if (indexLow < high) then QuickSort(myArray , indexLow, |  |  |  |
|  | high) |  |  |  |
|  | 33 if (low < indexHi) then QuickSort(my Array, low, |  |  |  |
|  | indexHi) |  |  |  |
|  | 34 |  |  |  |
|  | 35 End Procedure |  |  |  |
|  | One mark for each of the following up to a maximum of 9: |  |  |  |
|  | Declare / call sub procedure ‘QuickSort’ with a parameter | 1 |  |  |
|  | Declaring variables | 1 |  |  |
|  | Initialise pointers (high and low) | 1 |  |  |
|  | Set pivot to data at midpoint | 1 |  |  |
|  | Outer loop with terminating condition | 1 |  |  |
|  | Compare data with pivot | 1 |  |  |
|  | Increment / decrement pointers | 1 |  |  |
|  | Swap elements | 1 |  |  |
|  | Recursion index low, temp high | 1 |  |  |
|  | Recursion index high, temp low | 1 |  |  |
|  |  |  | 3a | 9 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 11. (a) | Award one mark for each correct column  Award one mark for correct combinations of X and Y | 3  1 |  | 2a |  | 4 |
| (b) |  | 4 |  | 2a |  | 4 |
|  | Award one mark for each correct column | 4 |  |  |  |  |
|  |  |  |  | 2a |  | 4 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *X* | *Y* | X NAND Y | 𝐗 𝐀𝐍𝐃 𝐘 | 𝐍𝐎𝐓(𝐗 𝐍𝐀𝐍𝐃 𝐘) |
| *0* | *0* | *1* | ***0*** | ***0*** |
| *0* | *1* | *1* | ***0*** | ***0*** |
| *1* | *0* | *1* | ***0*** | ***0*** |
| *1* | *1* | *0* | ***1*** | ***1*** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *A* | *B* | A NOR B | 𝐀 𝐎𝐑 (𝐀 𝐍𝐎𝐑 𝐁) | NOT B | 𝐀 𝐎𝐑 𝐍𝐎𝐓 𝐁 |
| *0* | *0* | *1* | ***1*** | *1* | ***1*** |
| *0* | *1* | *0* | ***0*** | *0* | ***0*** |
| *1* | *0* | *0* | ***1*** | *1* | ***1*** |
| *1* | *1* | *0* | ***1*** | *0* | ***1*** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
| 12. | **Indicative content**  A programming paradigm is the structure and approach of a programming language.  The are many different types of programming paradigms all of which are aimed at solving different programming problems.  Procedural programming languages are executed sequentially and are composed a series of programming commands and constructs executed on after the other.  Procedural programming is an imperative language as the programmer controls the flow and state of the computer programming.  These programming languages include sequence commands such as PRINT and INPUT and sections and iteration constructs such as IF and WHILE.  Procedural languages are usually employed when implementing algorithms as they are programmed linearly, they are not suitable for GUI based problems given the even driven nature of these applications.  Procedural programming is very easy to learn and implement although it not very suitable for complex problems. Examples of procedural programming languages are C and Pascal.  Event-Driven programming is a form of object-orientated programming which contain objects such as buttons, combo boxes and list boxes to create a GUI.  Event-driven programming uses event listeners to listen for events such a Click or Hover Over to which code can be attached and executes when a specific event occurs.  OOP event-driven programming is very effective when an application needs a rapid development of a GUI.  Some languages are known as multi-paradigm languages and are highly versatile and can be used in many different ways e.g. Python.  Different programming languages use a variety of different translators to convert programming code into machine code.  Compliers translate high-level language into machine code. |  | 1b |  |  | 12 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mark** | **AO1** | **AO2** | **AO3** | **Total** |
|  | The compiler produces a standalone executable file or package.  A compiled program can be executed without the need for the complier to be installed on the destination machine.  The complied application can be distributed to any machine with the OS it was targeted at, it can be run multiple times.  Compiled programs give some form of protection of intellectual property as the source code is not distributed to the destination machine.  Compilers target a specific platform e.g. iOS or Windows. Once compiled for this platform the executable can only be run on that platform.  Compiled programs cannot be compiled if they have any errors throughout any of the source code. This can be difficult to debug without the use of an IDE.  IT software needs to be distributed to more than one OS or architecture it will need to be recompiled and possibly reprogrammed.  Examples of compiled languages include Java and C++.  Interpreters, unlike, compilers translate one line of high- level source code into machine code line-by-line.  Each line of machine code is executed before the next is translated. Each time the program is run the code needs to be interpreted again.  For an interpreted application to be run you must have the source code, this mean intellectual property is difficult to manage.  If an error is found in an interpreted application, it with crash at that specific line meaning it can be easier to debug.  Code can be distributed to machine different OS and platforms as long as they have installed an appropriate interpreter.  Examples of interpreted languages include JavaScript and Python.  One line of compiled or interpreted high-level source code and be many lines of machine code. |  |  |  |  |  |

|  |  |
| --- | --- |
| **Band** | **Q12 AO1b - Max 12 marks** |
| **3** | **10-12 marks**  **The candidate has:**   * written an extended response that has a sustained line of reasoning which is coherent, relevant, and logically structured. * shown clear understanding of the requirements of the question and a clear knowledge of the topics as specified in the indicative content. Clear knowledge is defined as responses that provide relevant detailed points, which relate to an extensive amount of the indicative content. * addressed the question appropriately with minimal repetition and no irrelevant material * has presented a balanced discussion and justified their answer with examples * effectively drawn together different areas of knowledge, skills and understanding from all relevant areas across the course of study * used appropriate technical terminology confidently and accurately. |
| **2** | **5-9 marks**  **The candidate has:**   * written a response that has an adequate line of reasoning with elements of coherence, relevance, and logical structure * shown adequate understanding of the requirements of the question and a satisfactory knowledge of the topics as specified in the indicative content. Satisfactory knowledge is defined as responses that provide relevant points, which relate to the indicative content. * presented a discussion with limited examples * drawn together different areas of knowledge, skills and understanding from a number of areas across the course of study * used appropriate technical terminology. |
| **1** | **1-4 marks**  **The candidate has:**   * written a response that that lacks sufficient reasoning and structure * produced a discussion which is not well developed * attempted to address the question but has demonstrated superficial knowledge of the topics specified in the indicative content. Superficial knowledge is defined as responses that provide limited relevant points, which relate to a limited amount the indicative content. * used limited technical terminology. |
| **0** | Response not credit worthy or not attempted. |

A500U10-1 EDUQAS GCE A Level Computer Science – Component 1 MS A21/LG