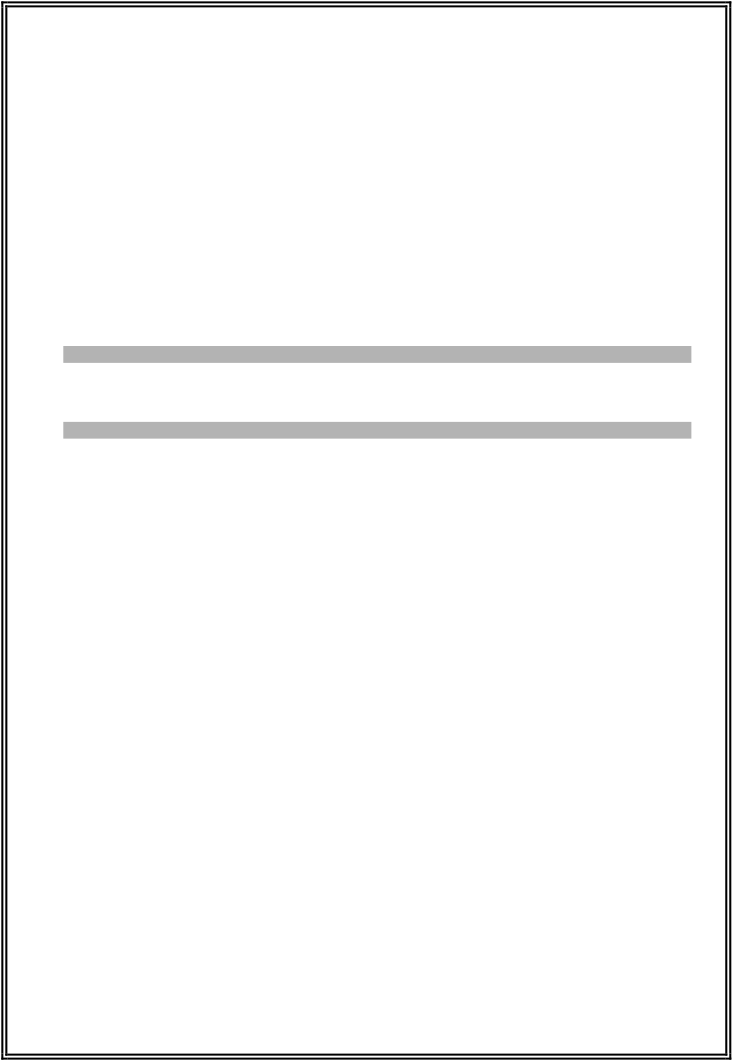
**Evaluation Only. Created with Aspose.Words. Copyright 2003-2023 Aspose Pty Ltd.**



# **GCE A LEVEL MARKING SCHEME**



**SUMMER 2019**

**A LEVEL (NEW)**

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

# **INTRODUCTION**

This marking scheme was used by WJEC for the 2019 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.

**GCE A LEVEL COMPUTER SCIENCE - COMPONENT 1 SUMMER 2019 MARK SCHEME**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mar** | **k AO1** | **AO2** | **AO3** | **AL**  **TO** |
| 1(a) | **One mark for each of the following up to a maximum of 4.**  High level languages are closer to the semantics of spoken language.  Each line of high level language translates in to multiple lines of machine code.  Low level languages such as assembly language uses mnemonics.  Each line of low level language is translated into one machine code instruction.  Identifiers can be long and meaningful  They allow use of more powerful commands that perform quite complex tasks  Allows the creation of modules that can be re-used and accessed by other parts of the program. | 1 1 1  1 1 1  1 | 1.1b 1.1b  1.1b  1.1b 1.1b 1.1b  1.1b |  |  | 4 |
| 1(b) | **One mark for stating a situation and one mark for the description**  Device drivers - low level language must be used to directly access memory addresses to fully control hardware.  Embedded software – software that runs on simple devices using simple microprocessors such as washing machines and microwaves will need direct access to the hardware  Real-time software – simulators or fly-by-wire systems that require precise processing, timings or accuracy could potential benefit from using a low-level language.  Assembly language can produce more compact code which can be important when placing on a chip. | 1 1  1 1 | 1.1b 1.1b  1.1b 1.1b |  |  | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mar** | **k AO1** | **AO2** | **AO3** | **AL**  **TO** |
| 2(a) | 1 mark for identifying i loop will execute n times. Accept n-1, n-2  1 mark for identifying j loop will execute n2 times. Accept n2-1 but not n2-2  1 mark for correct numbers of calculations 2n2 + n  1 mark for determining that the order will be dominated by n2  1 mark for determining that the growth rate for time performance is O(n2) | 1 1  1 1  1 |  |  | 3.1c 3.1c  3.1c 3.1c  3.1c | 5 |
| 2(b) | Time axis labelled correctly Size axis labelled correctly Correct gradient of line | 1 1 1 |  | 2.1b 2.1b 2.1b |  | 3 |
| 2(c) | The algorithm only uses one data structure, a one-dimensional array. Therefore, total storage requirements = 1.  As only one data structure is being used, the growth rate for memory will be constant O(1).  **Condone O(N)** | 1 1 |  |  | 3.1c 3.1c | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mar** | **k AO1** | **AO2** | **AO3** | **AL**  **TO** |
| 3(a) | Correct answer can be established using different steps / laws / rules / identities / dual relations.  One possible solution:  A.A + A.B + A.B  A + A + A.B + A + B A + A.B + B  (A + A). (A +B) + B  A +B + B  A + 1  1  Correctly applying identities to arrive at correct answer 5 marks  Correctly applying identities but arriving at wrong answer 1 mark for each correct step up to a maximum of 4 |  |  | 2.1a |  | 5 |
| 3(b) | Correct answer can be established using different steps / laws / rules / identities / dual relations.  One possible solution:  A.(B + B) + A.(A + B) A.(B + B) + A.A + A.B A.(B) + 0 + A.B  A.B + A.B  A + A  A  Correctly applying identities to arrive at correct answer 4 marks  Correctly applying identities but arriving at wrong answer 1 mark | 5 |  | 2.1a 2.1a |  | 5 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mar** | **k AO1** | **AO2** | **AO3** | **AL**  **TO** |
| 4 | **One mark for each of the following up to a maximum of 2.**  Alpha testing is conducted in-house by developers and occurs before the customer agrees to accept the final program.  Alpha builds are not shared with either the end user or with the customer.  Alpha builds are not final piece of software and often include limited functionality and many bugs.  **One mark for each of the following up to a maximum of 2**  Beta testing is conducted after alpha testing and later on in the software development life cycle.  Beta builds are shared with a limited number of end users to beta test the system with live data.  Beta builds contain all the main functionality but will still include some bugs.  Bugs reported by the beta testers are corrected by the development team.  **One mark for each of the following up to a maximum of 2**  Acceptance testing occurs is the final phase of testing during the software development life cycle.  Acceptance testing is undertaken by the actual end users of the system with real data.  The purpose of acceptance testing is to ensure the system has met the original requirements and specifications of the customer. | 1 1 1  1 1 1 1  1 1  1 | 1.1b |  |  | 6 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question** | **Answer** | **Mar** | **k AO1** | **AO2** | **AO3** | **AL**  **TO** |
| 5(a) | **One mark for each of the following up to a maximum of 4** Algorithms/programs can be broken down in to smaller parts.  These are named reusable pieces of code that can be called any number of times within an algorithm/program to perform a specific task.  Procedures are used to avoid the duplication of code.  Procedures are used to make an algorithm/program more efficient and secure.  Each procedure can be individually tested / debugged | 1 1  1 1 1 | 1.1b 1.1b  1.1b 1.1b 1.1b |  |  | 4 |
| 5(b) | Call by reference is where a value (address) is passed via a parameter into a subroutine and the original value is passed and used by that subroutine.  This is used if any changes made in the subroutine needs to be stored in the original value/variable outside the subroutine.  Call by value is where a value is passed via a parameter into a subroutine and a copy of the value is created for the duration of the subroutine call.  This ensures that the original value passed to the subroutine cannot be changed. | 1 1 1 1 | 1.1b 1.1b 1.1b 1.1b |  |  | 4 |

**This document was truncated here because it was created in the Evaluation Mode.**