GCE A LEVEL MARKING SCHEME

SUMMER 2019

A LEVEL (NEW)
COMPUTER SCIENCE - UNIT 3
1500U30-1

INTRODUCTION

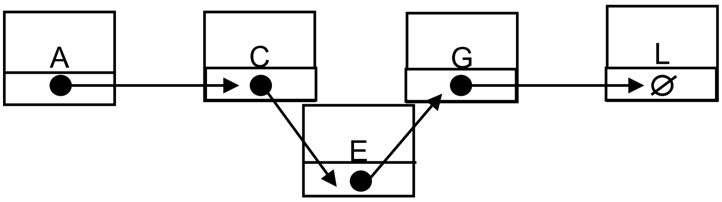
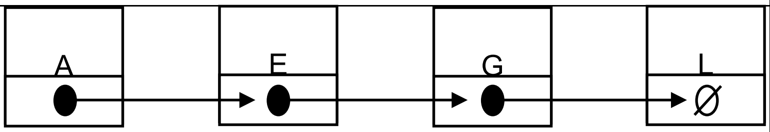
This marking scheme was used by WJEC for the 2019 examination. It was finalised after detailed discussion at examiners' conferences by all the examiners involved in the assessment. The conference was held shortly after the paper was taken so that reference could be made to the full range of candidates' 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 (NEW)
COMPUTER SCIENCE - UNIT 3

SUMMER 2019 MARK SCHEME

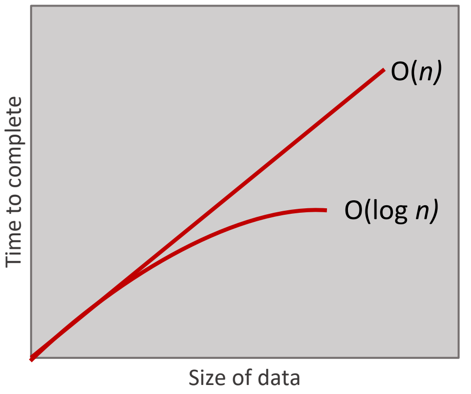
Question	Answer	Mark	AO1	AO2	AO3	Total
1(a)	**One mark for each principle plus one for the description** • A stack uses the last in first out (LIFO) principle. • In a stack the last or most recent item of data to be added to the stack is removed first. • Adding data to a stack is known as pushing, whilst removing data from a stack is known as popping. • A queue uses the first in first out (FIFO) principle. In a queue the last or most recent item of data added to a queue is the last to be removed. Give credit for references to pointers.	1 1 1 1 1	1.b			4
1(b)(i)	 Award 1 mark for node in correct position Award 1 mark for both arrows to and from correct nodes.	1 1		2.1		2
1(b)(ii)	 Award 1 mark for node in correct position Award 1 mark for both arrows to and from correct nodes.	1 1		2.1		2

Question	Answer	Mark	AO1	AO2	AO3	Total
2(a)	Correct answer can be established using different steps / laws / rules / identities / dual relations. One possible solution: $A.\overline{A} + A.B + A.\overline{B} + B.\overline{B}$ $A.B + A.\overline{B} + B.\overline{B}$ $A.B + A.\overline{B}$ $A.(B + \overline{B})$ $A.1$ A 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	5		2.1		5
2(b)	Correct answer can be established using different steps / laws / rules / identities / dual relations. One possible solution: $\overline{A.B} + A.C + B$ $\overline{A} + \overline{B} + A.C + B$ $\overline{A} + A.C + \overline{B} + B$ $\overline{A} + A.C$ $1 + C$ 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	5		2.1		5
3	01101001_2 $\underline{10000000_2}$ AND 00000000 State of left hand bit is 0 Award 1 mark for using an AND or XOR mask. Award 1 mark for correct result Award 1 mark for determining the state of the most significant left hand bit.	 1 1 1		2.1		3

Question	Answer	Mark	AO1	AO2	AO3	Total
4(a)	A natural language interface is where speech and linguistics is used (1 mark) to interact and control a software application. (1 mark)	2	1.b			2
4(b)	**Indicative content** One potential use for a natural language interface would be in translation software. Natural language could be processed in real time to allow for a seamless translation service. 1 mark for suitable example 1 mark for justification	 1 1	1.b			2
4(c)	**One mark per example up to a maximum of 3 marks.** Colloquialisms and words can be interpreted differently regionally. Accents could make is difficult for a natural language interface to identify the words being spoken. Ambiguity in spoken language where a word may have more than one interpretation. Background noise could cause problems. Illness such as sore throat	 1 1 1 1 1	1.b			3
5(a)	The algorithm sorts an array (1 mark) into ascending order (1 mark).	2		2.1		2
5(b)	This a recursive (1 mark) algorithm that calls itself (1 mark) and has a stopping condition (1 mark).	3		2.1		3
5(c)	**One mark for each point. Max of three marks.** Recursion has allowed for simpler design and for the original complex problem of sorting an array to be solved in a shorter time span than a non-recursive sorting algorithm. The algorithm is very compact The algorithm uses a divide and conquer design where the main problem is recursively broken into sub problems. The solutions of the sub problems are then combined to produce a solution for the original problem.	 1 1 1 1		2.1		3

<table>
<tr><th>Question</th><th>Answer</th><th>Mark</th><th>AO1</th><th>AO2</th><th>AO3</th><th>Total</th></tr>
<tr><td>6(a)</td><td>The purpose of a shortest path algorithm is to find the shortest path between two vertices on a graph.
Each path between adjacent vertices is weighted with a cost.
These weightings are used to calculate the total costs of different paths between two vertices.
The path with the smallest cost is the shortest path.</td><td>1
1
1
1</td><td>1.b</td><td></td><td></td><td>4</td></tr>
<tr><td>6(b)(i)</td><td><table>
<tr><th>Node</th><th>Cost</th><th>Node</th></tr>
<tr><td>A</td><td>3</td><td>B</td></tr>
<tr><td>A</td><td>4</td><td>C</td></tr>
<tr><td>B</td><td>12</td><td>D</td></tr>
<tr><td>C</td><td>10</td><td>D</td></tr>
<tr><td>D</td><td>6</td><td>E</td></tr>
</table>
<b>Indicative content</b>
Award one mark for correct paths
Award one mark for correct costings</td><td>1
1</td><td></td><td>2.1</td><td></td><td>2</td></tr>
<tr><td>6(b)(ii)</td><td>p = {a,b},{b,d},{d,e}
$|p|$ = 3 + 2 + 12 + 2 + 6 = 25
q = {a,c},{c,d},{d,e}
$|q|$ = 4 + 2 + 10 + 2 + 6 = 24
$|q| < |p|$
Path q is the shortest.</td><td>1
1</td><td></td><td>2.1</td><td></td><td>2</td></tr>
</table>

Question	Answer	Mark	AO1	AO2	AO3	Total
7(a)	A programming paradigm describes the different types or approaches in programming languages (1 mark) that are needed to solve different problems more effectively (1 mark).	2	1.b			2
7(b)	Procedural languages are those that solve a problem in a linear fashion through a sequence of step-by-step instructions and involve the use of selection, iteration and callable procedures. Procedural languages are more suited to problems that require a linear algorithm solution. Event-driven programming is used to solve problems that require heavy user interaction through a graphical interface. Listeners are attached to objects i.e. buttons, which in turn will execute a subroutine based on the type of event triggered i.e. a single click. Event-driven programming language are more suited to problems that require rapid application development and a graphical user interface.	1 1 1 1	1.b			4
7(c)	**Maximum of 4 marks.** **Indicative content** Object orientated programming could be use the development of a large distributed software application that requires a large team of developers. Event driven programming could be use the development of a graphical user interface software application. Logic programming could be use the development of artificial intelligence software. Functional programming could be use the development of software applications requiring complex mathematical transformations. Procedural programming could be use the development of command line interface software applications.	4	1.b			4

<table>
<tr><th>Question</th><th>Answer</th><th>Mark</th><th>AO1</th><th>AO2</th><th>AO3</th><th>Total</th></tr>
<tr><td>8</td><td>
<table>
<tr><th>A</th><th>B</th><th>A NOR B</th><th>NOT A AND NOT B</th></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
</table>
Award one mark for each correct row
Accept additional columns</td><td>4</td><td></td><td>2.1</td><td></td><td>4</td></tr>
<tr><td>9(a)</td><td>Each iteration in the loop halves the number of elements until you get 1 left.
For n values we halve the number until we get one value left which is $\log_2 n$.
There are two instructions in the loop there for $2\log_2 n$.
$\log_2 n$ dominates (1 mark) therefore time performance is O(log <i>n</i>) (1 mark)</td><td>1
1
1
2</td><td></td><td></td><td>3.3</td><td>5</td></tr>
<tr><td>9(b)</td><td>

Award one mark for both correct axes
Award one mark for correct O(<i>n</i>)
Award one mark for correct O(log <i>n</i>)
Award one mark for correct graph</td><td>1
1
1
1</td><td></td><td>2.1a</td><td></td><td>4</td></tr>
<tr><td>9(c)</td><td>O(log <i>n</i>) algorithms are more time efficient than O(<i>n</i>) algorithms when searching for a data item.</td><td>1</td><td></td><td></td><td>3.3</td><td>1</td></tr>
</table>

Question	Answer	Mark	AO1	AO2	AO3	Total
10	Data compression is reducing the amount of memory a file uses. The level of file compression is measured using a compression ratio. A compression ratio is the size of the compressed file divided by the original file size. A compression algorithms are used to employ different methods to reduce file size depending on the type of file. Accept a suitable example. There a two main types of file compression algorithms lossy and lossless. Lossy compression algorithms reduce a file size but some data is lost during this process and cannot be retrieved. Lossless compression algorithms reduce a files size without the loss of any data and the original file can be retrieved.	1 1 1 1 1 1	1.b			6
11(a)	A class is a template or blueprint for a specific object. It defines an object's variables (attributes/properties) and behaviour (methods). An object is an instance of a class. 1 mark for template/blueprint/contract 1 mark for defines variables (attributes/properties) 1 mark for defines behaviour (methods) 1 mark for stating an object is an instance of a class	1 1 1 1	1.b 1.b 1.b 1.b			4
11(b)	A method is a programmed behaviour/subroutine that is included in an object of a class. A method can only access data within its own object (encapsulation). 1 mark for a method is a programmed behaviour/subroutine 1 mark for stating a method can only access its own objects data (encapsulation).	1 1	1.b 1.b			2

<table>
<tr><th>Question</th><th>Answer</th><th>Mark</th><th>AO1</th><th>AO2</th><th>AO3</th><th>Total</th></tr>
<tr><td>12</td><td>

Indicative Content

```
Start Procedure Sort
declare myArray[]
i,j,temp is integer

set i = 0
while i <= len(myArray[])
     set j = 0
     while j <= len(myArray[] - 1)
          if myArray[j] < myArray[j + 1]
               temp = myArray[j]
               myArray[j] = myArray[j + 1]
               myArray[j + 1] = temp
          end if
     j = j + 1
     end while
i = i + 1
end while

End Procedure
```

Award one mark for correct declarations
Award one mark for correct outer loop structure
Award one mark for correct inner loop structure
Award one mark for correct if structure
Award one mark for correct placement of swap
Award one mark for correct placement of instantiation of j and i
Award one mark for any corrections (in bold)

- `set j=0`
- `set i=0`
- `i = i + 1`
- `myArray [j+1] = temp`
- `temp set twice`

</td><td>1
1
1
1
1
1
1</td><td></td><td></td><td>3.1</td><td>7</td></tr>
<tr><td>13</td><td>

Indicative Content

- Compilers, interpreters and assemblers are all examples of a translators. Translators are pieces of software used to convert one type of programming languages to another.
- Compilers coverts high-level programming language source code into object and machine code, run through a single executable file.
- The compilation process can throw multiple errors which at times can make debugging more difficult.
- Once software is compiled it does not need to go through recompilation unless changes are made to the original source code.
- One executable file produced by compilation can be executed many times.
- Many languages such as C++ and VB.Net

</td><td></td><td>1.b</td><td></td><td></td><td>13</td></tr>
</table>

Question	Answer	Mark	AO1	AO2	AO3	Total
	produce a single executable targeted to one platform or operating system i.e. an EXE file for a Windows platform. • If a program needs to be run on a different platform is will need to be recompiled and targeted at the required platform. i.e. Mach-O file for Mac OS. • However, some programming languages such as Java are executed within its own virtual machine and can be compiled into byte code and executed cross-platform within an installed Java Virtual Machine (JVM). • Once an application is compiled it is difficult to review the source code making intellectual property easier to protect. • Unlike compilers, interpreters covert high-level programming language source code line-by-line. • An interpreter translates a single line of code into machine code then executes it before moving onto the next. • An interpreted application does not produce an executable file, meaning source code must be interrupted each time the application is run. • To executed interpreted source code the needs to be a relevant interpreter installed on the running platform. • The same high-level source could can be interpreted on many different platforms, making the application highly portable. • Interpreted code could potentially be easier to debug as it will throw an exception at the current line being translated. • Interrupted applications need the source code to run, making intellectual property harder to protect. • An assembler is used to translate low-level assembly language mnemonics into machine code to directly program the CPU. • Each assembly language instruction has a one-to-one relationship with a machine code instruction unlike high-level languages where one instruction it translated into multiple machine code instructions. • This means that assembly is faster that compiling and interrupting and allows greater control over memory usage. • Directly writing code in binary machine code as this method would be prone to errors and highly time consuming hence using an assembly language and assembler.					
Total	100					

Band	Q13 AO1b - Max 13 marks					
3	**10 – 13 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 of the purpose and differences between compilers, interpreters and assemblers, 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 of the purpose and differences between compilers, interpreters and assemblers, 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 of the purpose and differences between compilers, interpreters and assemblers, which relate to a limited amount the indicative content. • used limited technical terminology.					
0	Response not credit worthy or not attempted.					