GCSE

Computer Science

Paper 1 Additional Questions
Mark scheme

V1.0
21/01/15

Mark schemes are prepared by the Lead Assessment Writer and considered, together with the relevant questions, by a panel of subject teachers. This mark scheme includes any amendments made at the standardisation events which all associates participate in and is the scheme which was used by them in this examination. The standardisation process ensures that the mark scheme covers the students' responses to questions and that every associate understands and applies it in the same correct way. As preparation for standardisation each associate analyses a number of students' scripts: alternative answers not already covered by the mark scheme are discussed and legislated for. If, after the standardisation process, associates encounter unusual answers which have not been raised they are required to refer these to the Lead Assessment Writer.

It must be stressed that a mark scheme is a working document, in many cases further developed and expanded on the basis of students' reactions to a particular paper. Assumptions about future mark schemes on the basis of one year's document should be avoided; whilst the guiding principles of assessment remain constant, details will change, depending on the content of a particular examination paper.

Further copies of this Mark Scheme are available from http://www.aqa.org.uk/

COMPONENT NUMBER:	**Paper 1 Additional Questions**
COMPONENT NAME:	
STATUS:	
DATE:	**21 January 2016**

To Examiners:

- **When to award '0' (zero) when inputting marks on CMI+**
  A mark of 0 should be awarded where a candidate has attempted a question but failed to write anything credit worthy.

  Insert a hyphen when a candidate has not attempted a question, so that eventually the Principal Examiner will be able to distinguish between the two (not attempted / nothing credit worthy) in any statistics.

- This mark scheme contains the correct responses which we believe that candidates are most likely to give. Other valid responses are possible to some questions and should be credited. Examiners should refer responses that are not covered by the mark scheme, but which they deem creditworthy, to a Team Leader.

The following annotation is used in the mark scheme:

;	- means a single mark
//	- means alternative response
/	- means an alternative word or sub-phrase
A	- means acceptable creditworthy answer
R	- means reject answer as not creditworthy
NE	- means not enough
I	- means ignore
DPT	- in some questions a specific error made by a candidate, if repeated, could result in the failure to gain the mark. The **DPT** label indicates that this mistake should only result in a candidate losing one mark on the first occasion that the error is made. Provided that the answer remains understandable, subsequent marks should be awarded as if the error was not being repeated.

Level of response marking instructions.

Level of response mark schemes are broken down into a number of levels, each of which has a descriptor. The descriptor for the level shows the average performance for the level. There are a range of marks in each level. The descriptor for the level represents a typical mid-mark performance in that level.

Before applying the mark scheme to a student's answer read through the answer and annotate it (as instructed) to show the qualities that are being looked for. You can then apply the mark scheme.

Step 1 Determine a level

Start at the lowest level of the mark scheme and use it as a ladder to see whether the answer meets the descriptor for that level. The descriptor for the level indicates the different qualities that might be seen in the student's answer for that level. If it meets the lowest level then go to the next one and decide if it meets this level, and so on, until you have a match between the level descriptor and the answer. With practice and familiarity you will find that for better answers you will be able to quickly skip through the lower levels of the mark scheme.

When assigning a level you should look at the overall quality of the answer and not look to pick holes in small and specific parts of the answer where the student has not performed quite as well as the rest. If the answer covers different aspects of different levels of the mark scheme you should use a best fit approach for defining the level and then use the variability of the response to help decide the mark within the level. ie if the response is predominantly level 3 with a small amount of level 4 material it would be placed in level 3 but be awarded a mark near the top of the level because of the level 4 content.

Step 2 Determine a mark

Once you have assigned a level you need to decide on the mark. The exemplar materials used during standardisation will help. There will be an answer in the standardising materials which will correspond with each level of the mark scheme. This answer will have been awarded a mark by the Lead Examiner. You can compare the student's answer with the example to determine if it is the same standard, better or worse than the example. You can then use this to allocate a mark for the answer based on the Lead Examiner's mark on the example.

You may well need to read back through the answer as you apply the mark scheme to clarify points and assure yourself that the level and the mark are appropriate.

Indicative content in the mark scheme is provided as a guide for examiners. It is not intended to be exhaustive and you must credit other valid points. Students do not have to cover all of the points mentioned in the indicative content to reach the highest level of the mark scheme.

An answer which contains nothing of relevance to the question must be awarded no marks.

01	**1**	**Marks are for AO1 (recall)** (An algorithm is) a sequence of steps; to complete a task; **A.** alternative correct definition	**2**

02	**1**	**Marks are for AO1 (recall)**
		
		Subroutine name/identifier;
		Parameters/input variables;
		Return type;
			3

02	**2**	**Marks are for AO1 (recall)** **Max 2** marks from: Subroutines can be developed in isolation/independently/separately; Easier to discover errors/testing is more effective (than without a structure); Subroutines can be updated without affecting the overall program; **A.** Other valid reason	**2**

03

Marks are for AO3 (program)

7

Marks awarded as follows:

A. Define a subroutine with the identifier `find_minimum`;
B. The subroutine has an array as an input variable (inspect code to check this input variable is an array)
C. Declare a variable to keep track of the minimum value (award even if logic to calculate minimum is incorrect)
D. The variable in mark B has the capability in the pseudo-code to be assigned different values from the array
E. Use of iteration to loop over entire array
F. Use of selection to check if current element in array is less than variable in mark B (award even if iteration logic is incorrect or absent)
G. The variable in mark B is returned at the end of the subroutine.

Example

```
SUBROUTINE find_minimum(arr)  [A][B]
   min ← arr[1]               [C]
   FOR x ← 2 TO LEN(arr)      [E]
      IF arr[x] < min THEN    [F]
         min ← arr[x]         [D]
      ENDIF
   ENDFOR
   RETURN min                 [G]
ENDSUBROUTINE
```

04 **1**

Marks are for AO1 (understanding)

1

Abstraction	●
Conversion	○
Decomposition	○
Validation	○

04	**2**	**Marks are for AO2 (understanding)** All friends have different first names; The time is rounded up to the nearest half-hour;	**2**

05	**1**	**Marks are for AO2 (apply)** The algorithm takes two inputs (xs and ys); The inputs are both arrays/strings/any iterable data structure; The output is the variable result;	**3**

05	**2**	**Marks are for AO2 (apply)** (The algorithm) adds together/sums; the length/size/number of elements of both arrays/inputs;	**2**

05	**3**	**Mark is for AO1 (recall)** A variable only accessible/visible within the subroutine; **A.** alternative wording	**1**

05	**4**	**Marks are for AO3 (design)** They have left out the RETURN keyword; **A.** corrected line of code `RETURN LEN(xs) + LEN(ys)`	**1**

05	**5**	**Marks are for AO2 (apply)** (The algorithm in answer 5.4) does not need to iterate/loop over the arrays/LEN could run in constant time/Only one addition is needed; so it will run in less time; **R**. it uses less lines of code or similar	**2**

06	**1**	**Marks are for AO2 (apply)** **Max of 4**: The search begins with the first element/4; The element is compared to the value being searched for/15; The search iterates over the elements sequentially; If the element in the array does not match the value being searched for then the iteration continues; The search returns true/the position of the element if the value is found in the array; **A.** a Boolean value/flag is set to True if the value is found; If the value is not found then the search returns False/-1/other appropriate value;	**4**

06	**2**	**Mark is for AO2 (understanding)** It is sorted;	**1**

06	**3**	**Marks are for AO2 (understanding)** The arrays are small (so); The run-time of both algorithms will be very similar// The difference in run-time will be negligible;	**2**

07	**1**	**Marks are for AO2 (apply)** 1 mark for each correct change (allow follow-on). The completed sequence is: 3 **1** **5** 4 2 3 1 **4** **5** 2 3 1 4 **2** **5** **1** **3** 4 2 5 1 3 **2** **4** 5	**5**

08	**1**	**Mark is for AO2 (apply)** Third row only; (It contains an error)	**1**

08	**2**	**Mark is for AO2 (apply)** Second row only; (2)	**1**

08	**3**	**Mark is for AO2 (apply)** Third row only; (True)	**1**