**CAMBRIDGE INTERNATIONAL EXAMINATIONS**
International General Certificate of Secondary Education

# MARK SCHEME for the October/November 2013 series

## 0420 COMPUTER STUDIES

**0420/31**        Paper 3, maximum raw mark 60

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge will not enter into discussions about these mark schemes.

Cambridge is publishing the mark schemes for the October/November 2013 series for most IGCSE, GCE Advanced Level and Advanced Subsidiary Level components and some Ordinary Level components.

1   (a)  One per reason, max 2 marks:
  –   simple project
  –   not a complex problem with many tasks
  –   few inter task dependencies
  –   timescale is more obvious
  –   Gantt charts are easier to understand/PERT charts can be confusing
  –   production of PERT charts can be expensive
  –   PERT charts can be complicated to manage

[2]

  (b)  Up to **three** points from
   (i)  Questionnaire for parents
    –   asks standard questions
    –   so results can be analysed easily/quickly
    –    no need for analyst to be present
    –   parents can complete at a time convenient to them
    –    (parents) can remain anonymous
    –   incentives to return questionnaire
    –   inexpensive/more efficient method for  <u>large groups</u> (of parents)

     Up to **three** points from
     interviewing for school office staff
    –   allows questions to be tailored to the individual
    –   can ask supplementary questions
    –   can ask for clarification
    –   staff may tell you things that you haven't identified as issues

     **Max 4 overall**, but up to 3 for each part

[4]

   (ii)  Any **one** from
    –   observation
    –   document search

[1]

**(c) (i)** max **three** marks for design, **one** mark for trip details, max **two** marks for
screen design
max **two** marks for explanation about user friendliness must relate to candidate's
design

| Screen | Explanation |
|--------|-------------|
| – inclusion/entry of trip details: Destination Activities Date(s Cost | – use of a suitable font to aid readability <br> – use of drop down boxes for ease of selection |
| – appropriate title | – to clearly identify the screen |
| – use of suitable picture(s) /logo/video clip | – to make the screen more attractive/understandable /identifiable |
| – navigation to consent form/ student details/parent details/payment | – clear links through the process |

[5]

**(ii)** max **three** marks for design, max **two** marks for explanation about user
friendliness must relate to candidate's design

| Screen | Explanation |
|--------|-------------|
| – entry of student details Name Class | – clear prompts/instructions for user (to improve accuracy of form filling) |
| – entry of parent details: Name Address Telephone no | – highlighting any errors to show the parent where there are problems with the information entered |
| – Consent | – explanation of why data entry method used |
| – use of submit/reset | – to clarify actions |
| – email | |

[5]
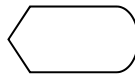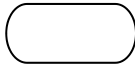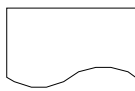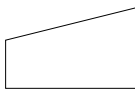
(d) up to **three** points from:
  – use of login/passwords(keyed or biometric)/smart cards
  – use of firewall
  – anti-virus software
  – anti-hacking/pharming/phishing/spyware software

[3]

(e) (i) One mark per symbol **and** description

|  Hard disk (database) |  Connector (a link to or from another part of diagram) |  Input/Output operation |
|---|---|---|
|  Visual Display Unit (Monitor) |  Terminator (Start and end of the flow diagram) |  Data processing operation |
|  Document output (printed hard copy) |  Manual input (e.g. keyboard) |  Communication line (e.g. telephone line) |

[4]

(ii) One mark for per process, max 3:
  – Storing consent details
  – Updating parents' payments
  – Transferring payments to school (trip) spreadsheet
  – <u>Sending</u> email confirmation of payments
  – <u>Sending</u> email reminder for <u>final payments</u>

One mark per input, max 2
  – Logon Details
  – Information for consent form/list of details
  – Payment details/debit or credit card details

One mark per data store, max 2
  – Pupil/Parent/Consent details
  – Trip details
  – Spreadsheet/payment details
  – Database/data store/storage
    *(only if none of the above are given)*

One mark per output, max 2
  – trip/consent details (screen not paper)
  – Confirmation of payment (email)
  – Request for final payment (email)

[6]

**(f)** up to **eight** marks from:
Prompt to enter amount  to pay
Input amount to pay
*(Note only one mark if the first two statements are reversed)*
Check new total would not be greater than total price of trip
Output error message if greater than
Exit routine
Update total amount
Calculate amount outstanding
Output appropriate message
Output amount outstanding

Sample algorithms

```
print "Enter amount to pay"                    (1)
input AmountToPay                              (1)
Total = Total + AmountToPay                    (1)
if Total > Price                               (1)
   then
   print "Payment is too large"                (1)
   exit                                        (1)
AmountOutstanding = Price - Total              (1)
print "Amount still outstanding"               (1)
print AmountOutstanding                        (1)


print "Enter amount to pay"                    (1)
input AmountToPay                              (1)
if AmountToPay + Total > Price                 (1)
   then
   print "Payment is too large"                (1)
   exit                                        (1)
Total = Total + AmountToPay                    (1)
AmountOutstanding = Price - Total              (1)
print "Amount still outstanding"               (1)
print AmountOutstanding                        (1)
```
[8]


**(g)** **One** mark per value suitable for amount paid, **one** mark per reason
(**the following two numbers are examples**)
–    100 – this checks that system  processes appropriate inputs
–    200 – rejected payment too large, error message produced

–    160 – boundary/extreme data that is doesn't produce an error message
–    161 – boundary data that produces an error message

[4]

**(h)** **one** mark for the choice, **three** marks for reasons only
(MUST match up with choice)
**existing software/off the shelf** chosen for the following reasons e.g.:
– software readily/immediately available
– already fully tested and de-bugged
– usually less expensive to buy
– technical support online/by phone
– usually compatible with other existing software
– large user base mutual support

**bespoke software/tailor made** chosen for the following reasons e.g.:
– can be tailored to meet the exact requirements for the school trip booking
– in contact with the actual programmers if there is a problem
– software can develop as it is used
– doesn't contain unwanted features                                                                 [4]


**(i)** One mark per point up to a maximum of **three**

Test data max 2
– Use of test data sets
– normal test data has known outcomes
– extreme test data checks validation rules
– abnormal test data should produce error messages
– mention of normal, abnormal, and extreme data
*(all 3 with no valid expansions, one mark if above not given)*

Types of testing max 2
– Black box
– White box
– System
– Alpha
– Beta
– User, etc.

Other
– testing that the school trip payment system meets the requirements of the school
– volume testing                                                                                                  [3]

**(j)** One mark per item, one mark per reason must match item, e.g.

program listing/coding
–   so alterations can be made
programming language(s) used
–   so the application can be recompiled
System flowchart
–   overview of whole system
flowchart/algorithm/pseudocode
–   so that the logic can be altered if required
input/output formats
hardware requirements
–   for testing any alterations/to know what is needed to run the system
software requirements
–   for testing any alterations/to know what is needed to run the system
known 'bugs' in the system
–   so they can be removed
list of variables used (and their meaning/description)
–   so that alterations to code don't spawn new errors
file structures/data model
–   - so that files and file sizes can be checked
Test plan/sample runs (with results and actual test data used)
–   - so that test can be re-run when alterations are made
validation rules
–   so that any extra inputs can be subject to similar validation                    [8]

**(k)** one mark for each point related to the school scenario, e.g.

–   discuss with school office staff/parents whether or not new system works
–   discuss with school office staff/parents whether or not system was easy to use
–   consider if objectives of new system have been met
–   look at results from tests
–   look at printouts, etc. to see if system produced the correct outcomes

[3]