Cambridge International AS & A Level

**Cambridge International Examinations**
Cambridge International Advanced Subsidiary and Advanced Level

**COMPUTER SCIENCE** **9608/33**

Paper 3 Written Paper **May/June 2017**

MARK SCHEME

Maximum Mark: 75

**Published**

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 May/June 2017 series for most Cambridge IGCSE®, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

| Question | Answer | Marks |
|---|---|---|
| 1(a)(i) | `DECLARE Book : LibraryBookRecord` | **1** |
| 1(a)(ii) | `Book.Title ← "Dune"` | **1** |
| 1(b) | `TYPE LibraryBookRecord`<br>`  DECLARE ISBN : INTEGER`<br>`  DECLARE Title : STRING`<br>`  DECLARE Genre : (Fiction, Non-Fiction)` **1**<br>`  DECLARE NumberOfLoans : 1 .. 99` **1**<br>`ENDTYPE`<br>mark for correct declaration and first two fields (**note**: only if attempt at modification) **1** | **3** |
| 1(c)(i) | 6715 | **1** |
| 1(c)(ii) | 8216 | **1** |
| 1(c)(iii) | 88 | **1** |
| 1(c)(iv) | `FALSE` | **1** |
| 1(d)(i) | `Temp2 ← 22` | **1** |
| 1(d)(ii) | `IntPointer ← @Temp1` | **1** |
| 1(d)(iii) | `IntPointer^ ← Temp2` | **1** |

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>2(a)(i)</td><td>Worm</td><td>1</td></tr>
<tr><td>2(a)(ii)</td><td>Phishing</td><td>1</td></tr>
<tr><td>2(a)(iii)</td><td>Malicious software that replicates by inserting a copy of itself (1)<br>into a file of data (1)</td><td>2</td></tr>
<tr><td>2(b)</td><td>Example:<br>No <u>up-to-date</u> anti-virus (or equivalent) software<br>Regular virus scans not performed<br>Operating system not up-to-date<br>Attachments/suspicious links clicked on<br>1 mark for any valid vulnerability</td><td>Max 2</td></tr>
<tr><td>2(c)(i)</td><td>public</td><td>1</td></tr>
<tr><td>2(c)(ii)</td><td>Bob sends his <u>digital certificate</u><br>Digital certificate contains Bob’s public key<br>Successful decryption of certificate using CA’s public key provides legitimacy<br>1 mark for any valid point – max 2</td><td>2</td></tr>
<tr><td>2(c)(iii)</td><td>
<table>
<tr><th>The person performing the action</th><th>What that person does</th><th></th></tr>
<tr><td>Anna</td><td>Requests Bob’s public key.</td><td></td></tr>
<tr><td>Bob</td><td>Sends Anna his public key.</td><td>1</td></tr>
<tr><td>Anna</td><td>Encrypts email with <u>Bob’s public key</u>.</td><td>1</td></tr>
<tr><td>Anna</td><td>Sends the email to Bob.</td><td></td></tr>
<tr><td>Bob</td><td>Decrypts email.<br>Using his private key.</td><td>1<br>1</td></tr>
</table>
</td><td>4</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>3(a)</td><td>X = A.($\overline{B}$ + (B . C))<br>B.C 1<br>$\overline{B}$ + B.C 1<br>A. 1</td><td>3</td></tr>
<tr><td>3(b)</td><td>
<table>
<tr><th>A</th><th>B</th><th>C</th><th>Working Space</th><th>X</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td></td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td></td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td></td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td></td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td></td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td></td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td></td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td></td><td>1</td></tr>
</table>
1 mark first four entries, 1 mark for the last four entries</td><td>2</td></tr>
<tr><td>3(c)(i)</td><td>
<table>
<tr><td></td><td></td><td colspan="4">AB</td></tr>
<tr><td></td><td></td><td>00</td><td>01</td><td>11</td><td>10</td></tr>
<tr><td rowspan="2">C</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
</table>
</td><td>1</td></tr>
<tr><td>3(c)(ii)</td><td>
<table>
<tr><td></td><td></td><td colspan="4">AB</td></tr>
<tr><td></td><td></td><td>00</td><td>01</td><td>11</td><td>10</td></tr>
<tr><td rowspan="2">C</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
</table>
</td><td>2</td></tr>
<tr><td>3(c)(iii)</td><td>X = A.$\overline{B}$ + A.C<br><br>1 1</td><td>2</td></tr>
<tr><td>3(d)</td><td>X = A.($\overline{B}$ + (B . C))<br>X = A.($\overline{B}$ + C ) 1<br>X = A.$\overline{B}$ + A.C 1 (dependent mark – must be correct outcome from previous line)</td><td>2</td></tr>
</table>

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(a)</td><td>Example:<br>Speed of access<br>Just used as a look-up file<br>No need for any serial or sequential processing<br>1 mark for any valid point</td><td>1</td></tr>
<tr><td>4(b)(i)</td><td>
<table>
<tr><th>CustomerID</th><th>RecordKey</th></tr>
<tr><td>802139</td><td>2139</td></tr>
<tr><td>700004</td><td>4</td></tr>
<tr><td>689998</td><td>89998</td></tr>
<tr><td>102139</td><td>2139</td></tr>
</table>
</td><td>1</td></tr>
<tr><td>4(b)(ii)</td><td>Minimum value: 0 1<br>Maximum value: 99999 1</td><td>2</td></tr>
<tr><td>4(b)(iii)</td><td>
<pre>
PROCEDURE InsertRecord(CustomerID : INTEGER)
   RecordKey ← CustomerID MOD 100000
   Success ← FALSE
   // Find position for new record and insert it
   REPEAT
      IF record at position RecordKey is empty
         THEN
            Insert new record at position RecordKey
            Success ← TRUE
         ELSE
            IF RecordKey = 99999
               THEN
                  RecordKey ← 0
               ELSE
                  RecordKey ← RecordKey + 1
            ENDIF
      ENDIF
   UNTIL Success = TRUE
ENDPROCEDURE
</pre>
</td><td>4</td></tr>
<tr><td>4(c)(i)</td><td>For security<br>If file is hacked then encrypted PIN cannot be used<br>Only encrypted PINs are transmitted and compared<br>1 mark for any valid point</td><td>Max 2</td></tr>
<tr><td>4(c)(ii)</td><td>1. Customer ID is read from card<br>2. Customer enters PIN<br>3. Customer PIN is encrypted<br>4. Customer ID is hashed<br>5. Customer record is located in file<br>6. PIN is checked against PIN in record<br>7. If match then transaction can proceed</td><td>3</td></tr>
</table>

| Question | Answer | Marks |
|---|---|---|
| 5(a)(i) | Packet:<br>Both web page and web page request are split into packets 1<br>Each packet is sent individually from device to device 1 | **2** |
| 5(a)(ii) | Router:<br>Transmit packets<br>Contain connections to many other routers<br>When packets arrive at router, router decides where next to send packet<br>1 mark for any valid point | **Max 2** |
| 5(a)(iii) | TCP/IP:<br>Is the protocol 1<br>Rules for communication between web server and browser 1 | **2** |
| 5(b)(i) | **Two** from:<br>Picture and sound not synchronised 1<br>Interruptions // video not continuous 1<br>Can be degraded by other competing traffic 1 | **Max 2** |
| 5(b)(ii) | Dedicated communications channel between the two communicating devices 1<br>Established prior to start of communication // removal of links at end of communication 1 | **2** |
| 5(b)(iii) | In packet switching, packets can take different routes and may not arrive in order<br>Will arrive in order (only one route)<br>As packets can take many different routes / share paths with others can be delayed<br>Dedicated circuit has full bandwidth<br>No loss of synch<br>1 mark for any valid point | **Max 3** |

| Question | Answer | Marks |
|---|---|---|
| 6(a)(i) | Control system | **1** |
| 6(a)(ii) | Use of actuators means that the system is controlling | **1** |
| 6(b) | System wastes processor time checking for values that are not changing 1<br>Some sensor input needs to be acted upon immediately 1 | **2** |
| 6(c)(i) | Interrupts need to be disabled so that the process of dealing with an interrupt is itself not interrupted | **1** |
| 6(c)(ii) | After handling the interrupt interrupts need to be enabled so that further interrupts can be dealt with | **1** |
| 6(c)(iii) | Content of registers 1<br>Placed on stack 1 | **2** |
| 6(c)(iv) | Changing sensor value dealt with as soon as it happens 1<br>Processor needs to check sensor only when an interrupt occurs 1 | **2** |
| 6(c)(v) | AND #B0000001000000000 // AND #&0200 // AND #512<br>Op code 1<br>Operand 1 | **2** |