MARK SCHEME for the October/November 2015 series

9608 COMPUTER SCIENCE

9608/41  Paper 4 (Written Paper), maximum raw mark 75

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 2015 series for most Cambridge IGCSE®, Cambridge International A and AS Level components and some Cambridge O Level components.

© IGCSE is the registered trademark of Cambridge International Examinations.
1 (a) (i) A F B C E G D J 5 5 5 5 3 [max. 7]

(ii) 1 – 2 – 3 – 5 – 6 – 7 – 9 – 8 – 10
1–5 scores 1
6–10 scores 1 [2]

(iii) 43 weeks [1]

(b) (i) week number 25 [1]

(ii) week number 32 [1]

(c) To see what activities can be done in parallel // show dependencies
To record changes to project timings [max. 1]
2 (a) parent(philippe, meena).
   parent(gina, meena). [2]

(b) ahmed, aisha, raul [2]

(c) father(F, ahmed). [1]

(d) mother(X, Y)
   IF
   female(X) AND parent(X, Y). [2]

(e) grandparent(W, Z)
   IF
   parent(W, X)

(f) grandfather(G, K)
   IF
   male(G) AND
   grandparent(G, K).

   alternative:

   father(G, X) AND
   parent(X, K). [2]
3 (a)

```
StockItem
Title: STRING
DateAcquired : TDATETIME
OnLoan: BOOLEAN
...

ShowTitle()
ShowDateAcquired()
ShowOnLoan()
...

Book
Author: STRING
ISBN: STRING
...

Constructor()
ShowAuthor()
ShowISBN()
...

CD
Artist: STRING
Playtime: INTEGER
...

Constructor()
ShowArtist()
ShowPlayTime()
...
```

[max. 7]
(b) (i) Mark as follows:

Class header
Methods
Properties

**Pascal**

```pascal
StockItem = CLASS
    PUBLIC
        Procedure ShowTitle();
        Procedure ShowDateAcquired();
        Procedure ShowOnLoan();
    PRIVATE
        Title : STRING;
        DateAcquired : TDateTime;
        OnLoan : Boolean;
    END;
```

**Python**

```python
class StockItem :
    def __init__(self) :
        self.__Title = ""
        self.__DateAcquired = ""
        self.__OnLoan = False

    def ShowTitle() :
        pass

    def ShowDateAcquired() :
        pass

    def ShowOnLoan() :
        pass
```

**VB.NET**

```vbnet
Class StockItem
    Public Sub ShowTitle()
    End Sub
    Public Sub ShowDateAcquired()
    End Sub
    Public Sub ShowOnLoan()
    End Sub
    Private Title As String
    Private DateAcquired As Date
End Class
```

© Cambridge International Examinations 2015
(ii) **Mark as follows:**

Class header and showing superclass

Methods

Properties

**Pascal**

TYPE Book = CLASS (StockItem)

PUBLIC
  Procedure ShowAuthor();
  Procedure ShowISBN();

PRIVATE
  Author : STRING;
  ISBN : STRING;

END;

**Python**

class Book(StockItem) :
  def __init__(self) :
    self.__Author = ""
    self.__ISBN = ""
  def ShowAuthor() :
    pass
  def ShowISBN() :
    pass

**VB.NET**

Class Book : Inherits StockItem
  Public Sub ShowAuthor()
  End Sub
  Public Sub ShowISBN()
  End Sub
  Private Author As String
  Private ISBN As String ' reject integer

End Class

© Cambridge International Examinations 2015
(iii) Pascal

```pascal
NewBook := Book.Create;
NewBook.Title := 'Computers';
NewBook.Author := 'A.Nyone';
NewBook.ISBN := '099111';
NewBook.DateAcquired := '12/11/2001';
NewBook.OnLoan := FALSE
```

**Python**

```python
NewBook = Book()
NewBook.Title = "Computers"
NewBook.Author = "A.Nyone"
NewBook.ISBN = "099111"
NewBook.DateAcquired = "12/11/2001"
NewBook.OnLoan = False
```

**VB.NET**

```vbnet
Dim NewBook As Book = New Book()
NewBook.Title = "Computers"
NewBook.Author = "A.Nyone"
NewBook.ISBN = "099111"
NewBook.DateAcquired = #12/11/2001#
NewBook.OnLoan = False
```

© Cambridge International Examinations 2015
4  (a)  

```
  Dodi  
  /     
 Ben    Farai  
 /  
 Ali  Celine  Elli  George  
```

(b)  

<table>
<thead>
<tr>
<th>RootPointer</th>
<th>Name</th>
<th>LeftPointer</th>
<th>RightPointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dodi</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Farai</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Elli</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>George</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>Ben</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>Celine</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>Ali</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

© Cambridge International Examinations 2015
(c) (i) 01 PROCEDURE TraverseTree(BYVALUE Root : INTEGER)
02   IF Tree[Root].LeftPointer < > 0
03     THEN
04       TraverseTree(Tree[Root].LeftPointer)
05     ENDIF
06   OUTPUT Tree[Root].Name
07   IF Tree[Root].RightPointer < > 0
08     THEN
09       TraverseTree(Tree[Root].RightPointer)
10   ENDIF
11 ENDPROCEDURE

(ii) A procedure that calls itself // is defined in terms of itself
Line number: 04/09

(iii) TraverseTree(RootPointer)

5 (a)

<table>
<thead>
<tr>
<th>Address</th>
<th>MemberID</th>
<th>other member data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1001</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>7002</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3005</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>4096</td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>98</td>
<td>2098</td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

1001 and 7002 and 3005 1
4096 and 2098 1
(b)  
(i)  
\[\begin{align*}
10 & \quad \text{// generate record address} \\
20 & \quad \text{NewAddress} \leftarrow \text{Hash(NewMember.MemberID)} \\
30 & \quad \text{// move pointer to the disk address for the record} \\
40 & \quad \text{SEEK NewAddress} \\
50 & \quad \text{PUTRECORD} "\text{MembershipFile}" , \text{NewMember} \\
\end{align*}\] 

(ii)  
\[\begin{align*}
01 & \quad \text{TRY} \\
02 & \quad \text{OPENFILE} "\text{MembershipFile}" \ \text{FOR RANDOM} \\
03 & \quad \text{EXCEPT} \\
04 & \quad \text{OUTPUT} "\text{File does not exist}" \\
05 & \quad \text{ENDTRY} \\
\end{align*}\] 

(iii) collisions/synonyms  
The previous record will be overwritten 

(iv) Create an overflow area  
The 'home' record has a pointer to others with the same key  
OR  
Store the overflow record at the next available address in sequence  
OR  
Re-design the hash function ....  
to generate a wider range of indexes // to create fewer collisions 

(v)  
\[\begin{align*}
41 & \quad \text{GETRECORD} "\text{MembershipFile}" , \text{CurrentRecord} \\
42 & \quad \text{WHILE} \ \text{CurrentRecord.MemberID} <> 0 \\
44 & \quad \text{IF} \ \text{NewAddress} > 99 \ \text{THEN} \ \text{NewAddress} \leftarrow 0 \\
45 & \quad \text{SEEK} \ \text{NewAddress} \\
46 & \quad \text{GETRECORD} "\text{MembershipFile}" , \text{CurrentRecord} \\
47 & \quad \text{ENDWHILE} \\
\end{align*}\] 

[max. 4]