CAMBRIDGE INTERNATIONAL EXAMINATIONS

Cambridge International Advanced Level

MARK SCHEME for the October/November 2015 series

9608 COMPUTER SCIENCE

9608/43 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.

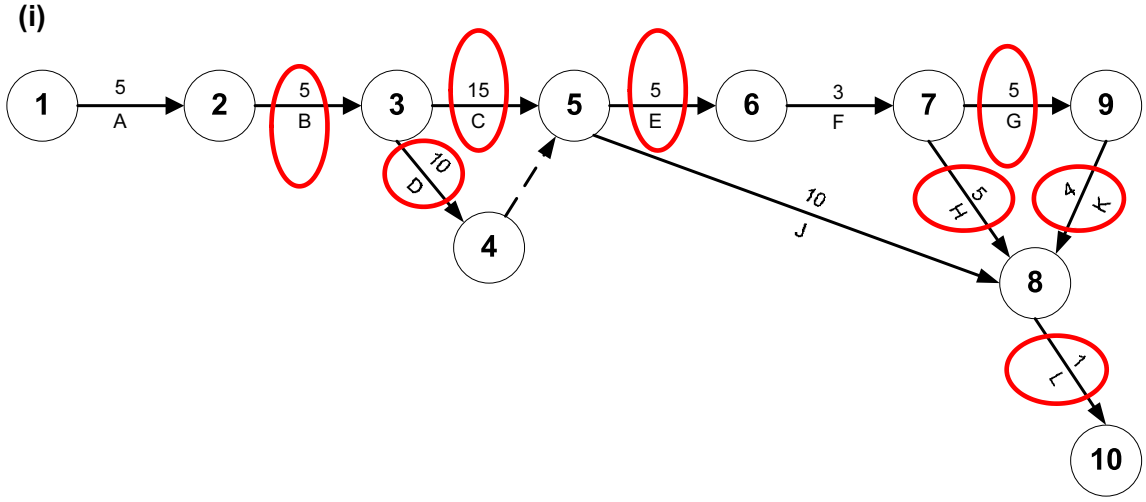
Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

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.

CAMBRIDGE
International Examinations

1 (a) (i)



[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]

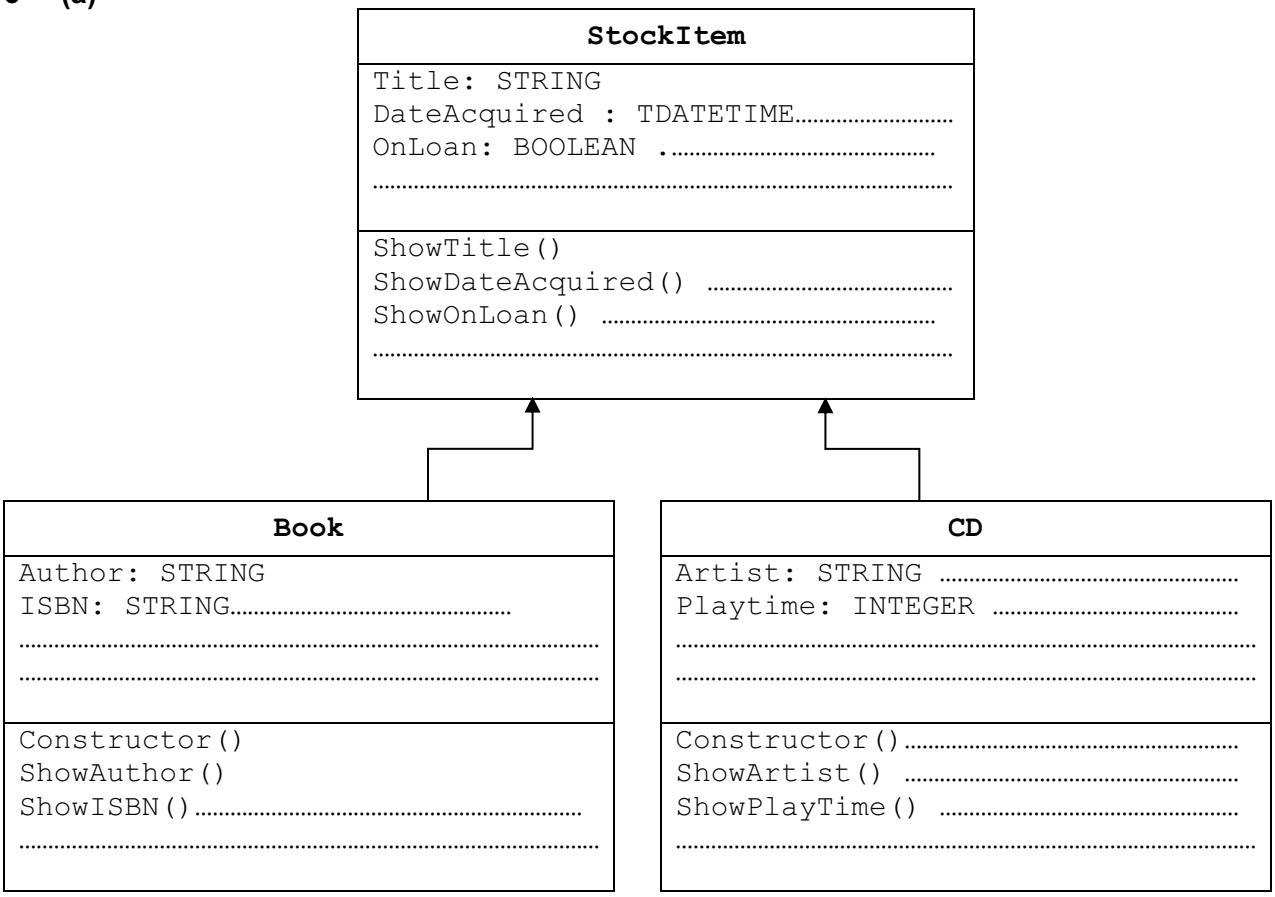
(e)
```
grandparent(W, Z)
IF
parent(W,X)
AND parent(X,Z).
```
[2]

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

alternative:

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

3 (a)



[max. 7]

(b) (i) *Mark as follows:*
Class header
Methods
Properties

Pascal

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

Python

```
class StockItem :
    def __int__(self) :
        self.__Title = ""
        self.__DateAquired = ""
        self.__OnLoan = False

    def ShowTitle() :
        pass
    def ShowDateAcquired() :
        pass
    def ShowOnLoan() :
        pass
```

VB.NET

```
Class StockItem
    Public Sub ShowTitle()
    End Sub
    Public Sub ShowDateAquired()
    End Sub
    Public Sub ShowOnLoan()
    End Sub
    Private Title As String
    Private DateAquired As Date
End Class
```

[3]

(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
```

[3]

(iii) Pascal

```
NewBook := Book.Create;                                  1
NewBook.Title := 'Computers';
NewBook.Author := 'A.Nyone';
NewBook.ISBN := '099111';                                1
NewBook.DateAcquired := '12/11/2001';
NewBook.OnLoan := FALSE                                  1
```

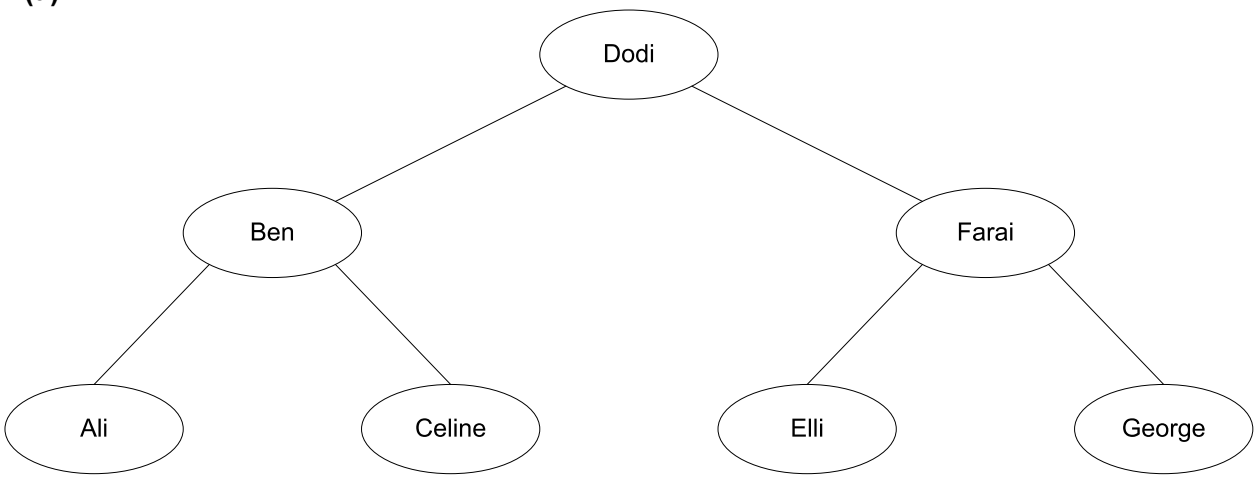
Python

```
NewBook = Book()                                         1
NewBook.Title = "Computers"
NewBook.Author = "A.Nyone"
NewBook.ISBN = "099111"                                  1
NewBook.DateAcquired = "12/11/2001"
NewBook.OnLoan = False                                   1
```

VB.NET

```
Dim NewBook As Book = New Book()                         1
NewBook.Title = "Computers"
NewBook.Author = "A.Nyone"
NewBook.ISBN = "099111"                                  1
NewBook.DateAcquired = #12/11/2001#
NewBook.OnLoan = False                                   1        [3]
```

4 (a)



[4]

(b)

RootPointer: 1

FreePointer: 8

Tree

	Name	LeftPointer	RightPointer
[1]	Dodi	5	2
[2]	Farai	3	4
[3]	Elli	0	0
[4]	George	0	0
[5]	Ben	7	6
[6]	Celine	0	0
[7]	Ali	0	0
[8]		9	0
[9]		10	0
[10]		0	0

[7]

(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
```

[5]

(ii) A procedure that calls itself // is defined in terms of itself
Line number: 04/09 [2]

(iii) `TraverseTree(RootPointer)` [1]

5 (a)

`MembershipFile`

Address	MemberID	other member data
0	0	
1	1001	
2	7002	
3	0	
4	0	
5	3005	
6	0	
7	0	
8	0	
:	:	
:	:	
96	4096	
97	0	
98	2098	
99	0	

1001 and 7002 and 3005 1
4096 and 2098 1 [2]

(b) (i)
```
10 // generate record address
20 NewAddress ← Hash(NewMember.MemberID)
30 // move pointer to the disk address for the record
40 SEEK NewAddress
50 PUTRECORD "MembershipFile", NewMember
```
[4]

(ii)
```
01 TRY
02    OPENFILE "MembershipFile" FOR RANDOM
03 EXCEPT
04    OUTPUT "File does not exist"
05 ENDTRY
```
[2]

(iii) collisions/synonyms
The previous record will be overwritten [2]

(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 [2]

(v)
```
41 GETRECORD "MembershipFile", CurrentRecord
42 WHILE CurrentRecord.MemberID <> 0
43    NewAddress ← NewAdress + 1
44    IF NewAddress > 99 THEN NewAddress ← 0
45    SEEK NewAddress
46    GETRECORD "MembershipFile", CurrentRecord
47 ENDWHILE
```
[max. 4]