Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

COMPUTER SCIENCE	**9608/04**
Paper 4 Further Problem-solving and Programming Skills	**For Examination from 2015**
SPECIMEN MARK SCHEME	
	2 hours

MAXIMUM MARK: 75

This document consists of **9** printed pages and **1** blank page.

CAMBRIDGE
International Examinations

1 **(a)** *Mark as follows:*

```
High ← 63                                                        [1]
X = 0                                                            [1]
High ← Middle – 1                                                [1]
```

One mark for each correct line

(b) **(i)** ordered / in order [1]

(ii) 6 [1]

(iii) 0 [1]
item not present in array [1]
non zero [1]
position of the item in the array [1]

(c) **(i)** e.g. in Python:

```
def BinarySearch(Low, High):                                     [1]
    global Found
    if Low>High:                                           )
        return                                             )     [1]
    Middle=int((High+Low)/2)
    if SearchData[Middle] == SearchItem:                   )
        Found = Middle                                     )     [1]
    elif SearchData[Middle] < SearchItem:                  )
        BinarySearch(Middle + 1, High)                     )     [1]
    elif SearchData[Middle] > SearchItem:                  )
        BinarySearch(Low, Middle - 1)                      )     [1]
    return
```

(d) `BinarySearch(1,63)` [1]

[Total: 15]

2 (a)

	0	1	2	3	4	5	6	7	8
Conditions	Group 1 tests	Y	Y	Y	Y	N	N	N	N
	Group 2 tests	Y	Y	N	N	Y	Y	N	N
	Group 3 tests	Y	N	Y	N	Y	N	Y	N
Actions	Accepted	Y							
	Repair		Y	Y					
	Rejected				Y	Y	Y	Y	Y

correct column 1	[1]
correct columns 2 and 3	[1]
correct column 4	[1]
correct columns 5–8	[1]

(b)

	0	1	2	3	4	5			
Conditions	Group 1 tests	Y	Y	Y	Y	N			
	Group 2 tests	Y	Y	N	N	–			
	Group 3 tests	Y	N	Y	N	–			
Actions	Accepted	Y							
	Repair		Y	Y					
	Rejected				Y	Y			

correct column 1	[1]
correct column 2	[1]
correct column 3	[1]
correct column 4	[1]
correct column 5	[1]

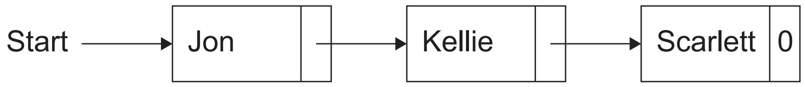
(c) e.g. in Python:

```
def Reject():
    if ((G1Tests() == True and G2Tests() == False and
G3Tests() == False)or G1Tests() == False):
        return True
```

correct function header	[1]
correct if statement	[1]
correct return statement	[1]

[Total: 12]

3 (a)



Mark as follows:

Three correct items [1]

Indication of correct order with start and termination [1]

(b)

```
Type ListNode
   Pointer as Integer
   Name As String
EndType
```

Mark as follows:

Record structure definition [1]

Pointer field definition [1]

Node data definition [1]

(c)

```
Dim NameList[1..50] As ListNode
```

Mark as follows:

Appropriate size of array [1]

Use of user defined record type [1]

(d) (i)

HeadPointer
0

FreePointer
1

NameList

	Name	Pointer
[1]		2
[2]		3
[3]		4
[4]		5
: :		
[49]		50
[50]		0

Mark as follows:

HeadPointer [1]

FreePointer [1]

Pointers[1] – [49] [1]

Pointer[50] [1]

(ii)

```
FOR Index ← 1 TO 49
   NameList[Index].Pointer ← Index + 1
ENDFOR
NameList[50].Pointer ← 0
HeadPointer ← 0
FreePointer ← 1
```

Mark as follows:

Correct FOR loop [1]
Correct setting of Pointer[50], HeadPointer and FreePointer [1]

(e) (i)

```
PROCEDURE AddItem(NewItem)
//
  NameList[FreePointer].Name ← NewItem
  CurrentPointer ← HeadPointer                                    [1]
//
  REPEAT
    IF NameList[CurrentPointer].Name < NewItem                    [1]
      THEN
        PreviousPointer ← CurrentPointer
        CurrentPointer ← NameList[CurrentPointer].Pointer         [1]
    ENDIF
  UNTIL NameList[CurrentPointer].Name > NewItem
//
  IF CurrentPointer = HeadPointer
    THEN
      NameList[FreePointer].Pointer ← HeadPointer
      HeadPointer ← FreePointer
    ELSE
      NameList[FreePointer].Pointer
                        ← NameList[PreviousPointer].Pointer
      NameList[PreviousPointer] ← FreePointer
  ENDIF
  FreePointer ← NameList[FreePointer].Pointer
ENDPROCEDURE
```

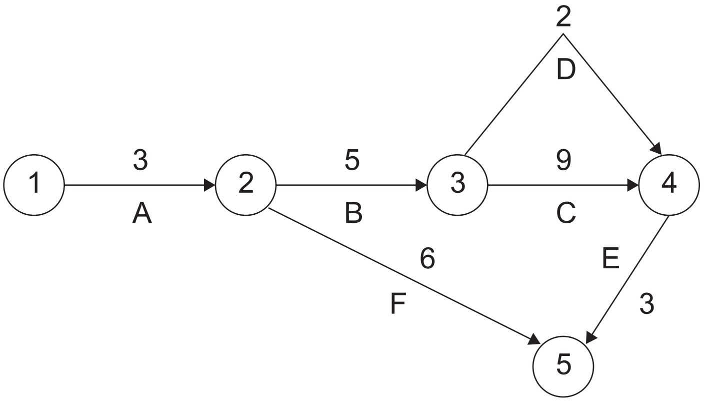
(ii) New item placed in node at head of Free List [1]

(iii) Loop that repeats until position of new item located [1]
Records current pointer and then updates current pointer [1]

(iv) Check to see whether new item is first in linked list [1]
If first item then place item at head of list [1]
If not first item then adjust pointers to place it in correct position in list [1]

[Total: 22]

4 (a)



1 mark for each correctly labelled activity – max 4 marks [max 4]

(b) (i) 1 – 2 – 3 – 4 – 5 [1]

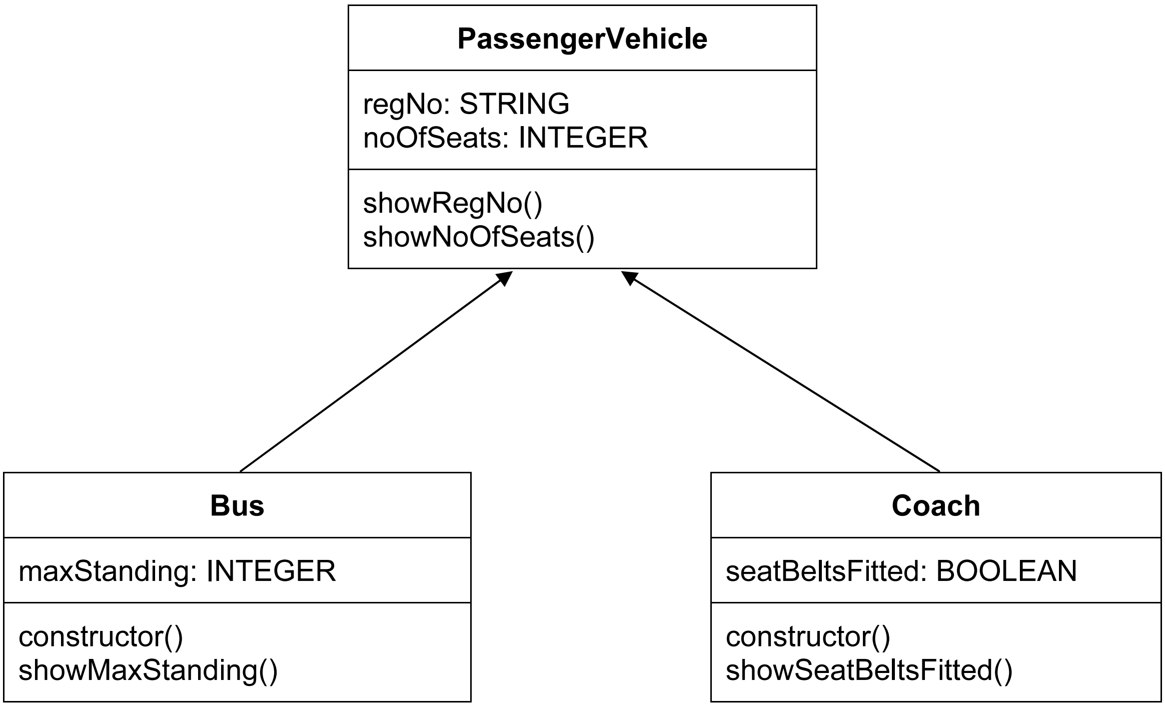
(ii) 20 [1]

(c) (i) 8 [1]

(ii) 17 [1]

[Total: 8]

5 **(a)**



Mark as follows:

noOfSeats declaration and associated show method in PassengerVehicle	[1]
inheritance arrows	[1]
constructor method in Coach	[1]
seatBeltsFitted declaration and associated show method in Coach	[1]

(b) e.g. in Python:

```
class PassengerVehicle():
    def __init__(self, regNo, noOfSeats):
        # Sets all the initial values
        self.__ regNo = regNo
        self.__ noOfSeats = noOfSeats

    def showRegNo(self):
        print("Registration No: ",self.__regNo)

    def showNoOfSeats (self):
        print("No of seats: ",self.__noOfSeats)
```

Mark as follows:

data declarations	[1]
use of __ in identifiers to give "private" attribute	[1]
use of 'self' parameter	[1]
showRegNo function	[1]
showNoOfSeats function	[1]

e.g. in Visual Basic:

```
MustInherit Class PassengerVehicle
        Protected regNo As String
        Protected noOfSeats As Integer

        Public Sub showRegNo()
            Console.WriteLine(regNo)
        End Sub

        Public Sub showNoOfSeats()
            Console.WriteLine(noOfSeats)
        End Sub

    End Class
```

Mark as follows:

MustInherit	[1]
data declarations	[1]
protected	[1]
showRegNo function	[1]
showNoOfSeats function	[1]

(c) e.g. in Python:

```
class Bus(PassengerVehicle):
    def __init__(self, regNo,
 noOfSeats, maxStanding):
        super().__init__(regNo, noOfSeats)
        self.__maxStanding = maxStanding

    def showMaxStanding (self):
        print("No of standing passengers: ", self.__maxStanding)
```

Mark as follows:

inheritance	[1]
__init__ function header	[1]
use of __init__ from superclass	[1]
initialisations in __init__ function	[1]
showMaxStanding function	[1]

e.g. in Visual Basic:

```
Class Bus
    Inherits PassengerVehicle
    Private maxStanding As Integer
    Public Sub New(ByVal regNoValue As String, ByVal
        noOfSeatsValue As Integer, ByVal
        maxStandingValue As Integer)
        regNo = regNoValue
        noOfSeats= noOfSeatsValue
        maxStanding = maxStandingValue
    End Sub
    Public Sub ShowMaxStanding ()
        Console.WriteLine(maxStanding)
    End Sub
End Class
```

Mark as follows:

inheritance [1]
private [1]
Public Sub New header [1]
Initialisations in Sub New [1]
ShowMaxStanding function [1]

(d) **(i)** e.g. in Python:

```
pv1 = Bus("NBR 123", 51,10)
```
[1]

e.g. in Visual Basic:

```
Dim pv1 As Bus = New Bus("NBR 123", 51, 10)
```
[1]

(ii) e.g. in Python:

```
pv1.showRegNo()
pv1.showNoOfSeats()
pv1.showMaxStanding()
```
[1]
[1]
[1]

e.g. in Visual Basic

```
pv1.showRegNo()
pv1.showNoOfSeats()
pv1.showMaxStanding()
```
[1]
[1]
[1]

[Total: 18]

BLANK PAGE