MARK SCHEME for the May/June 2015 series

9608 COMPUTER SCIENCE

9608/42	Paper 4 (Written Paper), maximum raw mark 75

1
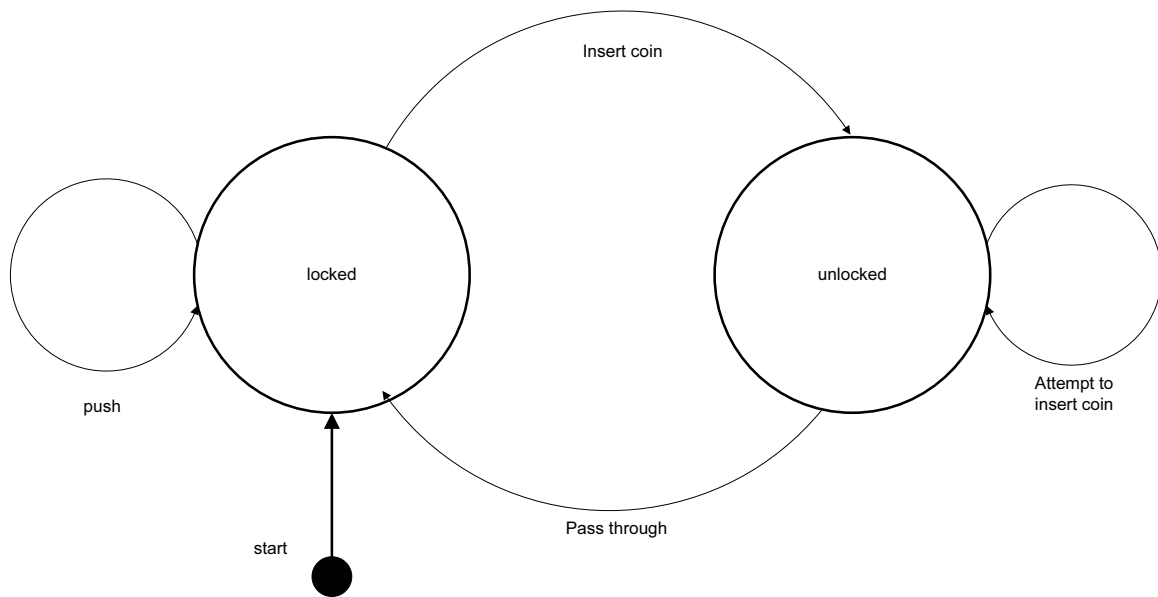
Mark as follows:
1 mark for both states correct
1 mark for each further label **[5]**

2 (a) `capital_city(santiago).`
`city_in_country(santiago, chile).`
`country_in_continent(chile,south_america).`
`city_visited(santiago).`

accept in any order **[4]**

(b) `ThisCity =`
manchester
london **[2]**

(c) `countries_visited(ThisCountry)`
`IF`
city_visited(ThisCity) 1
AND 1
city_in_country(ThisCity, ThisCountry) 2 **[4]**

© Cambridge International Examinations 2015

3 (a)

Conditions	goods totalling more than $20	Y	Y	Y	Y	N	N	N	N
	goods totalling more than $100	Y	Y	N	N	Y	Y	N	N
	have discount card	Y	N	Y	N	Y	N	Y	N
Actions	No discount				X	X	X	X	X
	5% discount		X	X					
	10% discount	X							
		1 mark	1 mark	1 mark	1 mark				

[4]

(b)

Conditions	goods totalling more than $20	Y	Y	Y	Y	N			
	goods totalling more than $100	Y	Y	N	N	-			
	have discount card	Y	N	Y	N	-			
Actions	No discount				X	X			
	5% discount		X	X					
	10% discount	X							

1 mark per column

[5]

© Cambridge International Examinations 2015

(c) Example Pascal

```
    FUNCTION Discount(GoodsTotal: INTEGER; HasDiscountCard: BOOLEAN) :
INTEGER;

        BEGIN
(1)            IF GoodsTotal > 20
(1)            THEN
(2)              IF GoodsTotal > 100
(2)                  THEN
(3)                    IF HasDiscountCard = TRUE
(3)                        THEN
(3)                            Discount := 10
(3)                        ELSE
(3)                            Discount := 5
(2)                    ELSE
(4)                        IF HasDiscountCard = TRUE
(4)                        THEN
(4)                            Discount := 5
(4)                        ELSE
(4)                            Discount := 0
(1)                  ELSE
(1)                    Discount := 0;
        END;
```
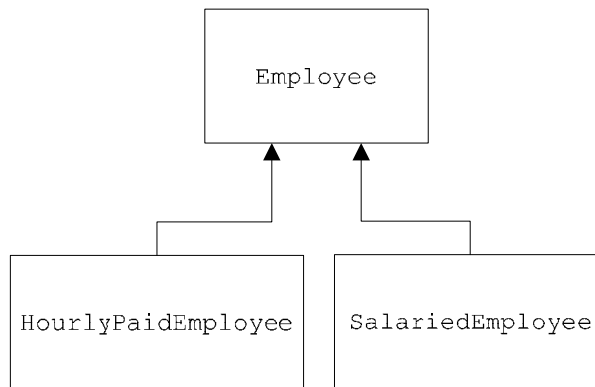
Example Python

```
    def Discount(GoodsTotal, HasDiscountCard) :

(1)    if GoodsTotal > 20:
(2)        if GoodsTotal > 100:
(3)            if HasDiscountCard == True:
(3)                return 10
(3)            else:
(3)                return 5
(2)        else:
(4)            if HasDiscountCard == TRUE:
(4)                return 5
(4)            else:
(4)                return 0
(1)    else:
(1)        return 0
```
                                                                            [6]

4 (a)

Employee

HourlyPaidEmployee   SalariedEmployee

[3]

(b) Example Pascal

```
Type
Employee = CLASS
        PUBLIC
            procedure SetEmployeeName
            Procedure SetEmployeeID
            Procedure CalculatePay
        PRIVATE
            EmployeeName : STRING
            EmployeeID : STRING
            AmountPaidThisMonth : Currency
    END;
```

Mark as follows:

Class header	(1 mark)
`PUBLIC` and `PRIVATE` used correctly	(1 mark)
`EmployeeName` **+** `EmployeeID`	(1 mark)
`AmountPaidThisMonth`	(1 mark)
Methods x 3	(1 mark)

Example VB

```
Class Employee
    Private EmployeeName As String
    Private EmployeeID As String
    Private AmountPaidThisMonth As Decimal
Public Sub SetEmployeeName()
End Sub
Public Sub SetEmployeeID()
End Sub
Public Sub CalculatePay()
End Sub
```

Example Python

```
Class Employee():
    def __init__(self):
        self.__EmployeeName = ""
        self.__EmployeeID = ""
        self.__AmountPaidThisMonth = 0
    def SetEmployeeName(self, Name):
        self.__EmployeeName = Name
    def SetEmployeeID(self, ID):
        self.__EmployeeID = ID
    def SetAmountPaidThisMonth(self, Paid):
        self.__AmountPaidThisMonth = Paid
```
[max 5]

(c) (i)
```
HoursWorked                              1
HourlyPayRate                            1
SetHoursWorked                           1
CalculatePay : Override                  1 + 1
SetPayRate                               1              [max 4]
```

(ii)
```
AnnualSalary                             1
SetSalary                                1
CalculatePay : Override                  1              [max 2]
```

(d) Polymorphism **[1]**

5 (a) (i)
```
FOR ThisPointer ← 2 TO 10
    // use a temporary variable to store item which is to
    // be inserted into its correct location
    Temp ← NameList[ThisPointer]
    Pointer ← ThisPointer – 1

    WHILE (NameList[Pointer] > Temp) AND (Pointer > 0)
       // move list item to next location
       NameList[Pointer + 1] ← NameList[Pointer]
       Pointer ← Pointer - 1
    ENDWHILE

    // insert value of Temp in correct location
    NameList[Pointer + 1] Temp←
ENDFOR
```
1 mark for each gap filled correctly **[7]**

(ii)
The outer loop (FOR loop) is executed 9 times (1 mark)
it is not dependant on the dataset (1 mark)

The Inner loop (WHILE loop) is not entered (1 mark)
as the condition is already false at the first encounter (1 mark) **[max 3]**

(b) (i)
outer loop is executed 9 times (1 mark)
inner loop is executed 9 times (for each iteration of the outer loop) (1 mark)
not dependant on the dataset (1 mark) **[max 2]**

(ii)
```
NumberOfItems ← 10
REPEAT
    NoMoreSwaps   ← TRUE

    FOR Pointer ← 1 TO NumberOfItems – 1
        IF NameList[Pointer] > NameList[Pointer + 1]
            THEN
                NoMoreSwaps ← FALSE
                Temp ← NameList[Pointer]
                NameList[Pointer] ← NameList[Pointer + 1]
                NameList[Pointer + 1] ← Temp
        ENDIF
    ENDFOR
    NumberOfItems ← NumberOfItems - 1
UNTIL NoMoreSwaps = TRUE
```
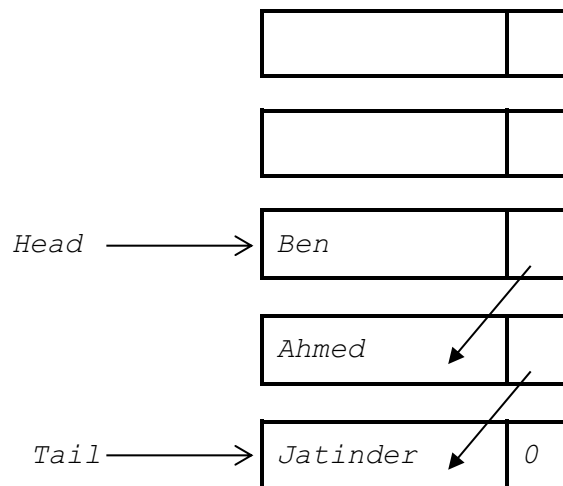
Mark as follows:
- change outer loop to a REPEAT/WHILE loop   (1 mark)
- FOR loop has variable used for final value   (1 mark)
- Initialise Boolean variable to TRUE   (1 mark)
- set Boolean variable to FALSE in correct place   (1 mark)
- number of items to consider on each pass decrements   (1 mark)
- Correct stopping condition for REPEAT loop   (1 mark)   **[max 5]**

6 (a)

1 mark for Head and Tail pointers
1 mark for 3 correct items – linked as shown
1 mark for correct order with null pointer in last nod   **[3]**

© Cambridge International Examinations 2015

(b) (i)

Queue

HeadPointer   Name   Pointer

	Name	Pointer
[1]		2
[2]		3
[3]		4
[4]		5
[5]		6
[6]		7
[7]		8
[8]		9
[9]		10
[10]		0

HeadPointer

```
0
```

TailPointer

```
0
```

FreePointer

```
1
```

Mark as follows:

HeadPointer =0 & TailPointer = 0
*FreePointer **assigned a value***
*Pointers[1] **to** [9] **links the nodes together***
Pointer[10] = 'Null' **[4]**

(ii)
```
PROCEDURE RemoveName()
    // Report error if Queue is empty
    IF HeadPointer = 0
       THEN
           Error
       ELSE
           OUTPUT Queue[HeadPointer].Name
           // current node is head of queue
           CurrentPointer ← HeadPointer
           // update head pointer
           HeadPointer ← Queue[CurrentPointer].Pointer
           //if only one element in queue,then update tail pointer
           IF HeadPointer = 0
              THEN
                  TailPointer ← 0
           ENDIF
              // link released node to free list
              Queue[CurrentPointer].Pointer ← FreePointer
           FreePointer ← CurrentPointer
    ENDIF
ENDPROCEDURE
```
[max 6]

© Cambridge International Examinations 2015