4 A program is to be written to calculate the discount given on purchases.

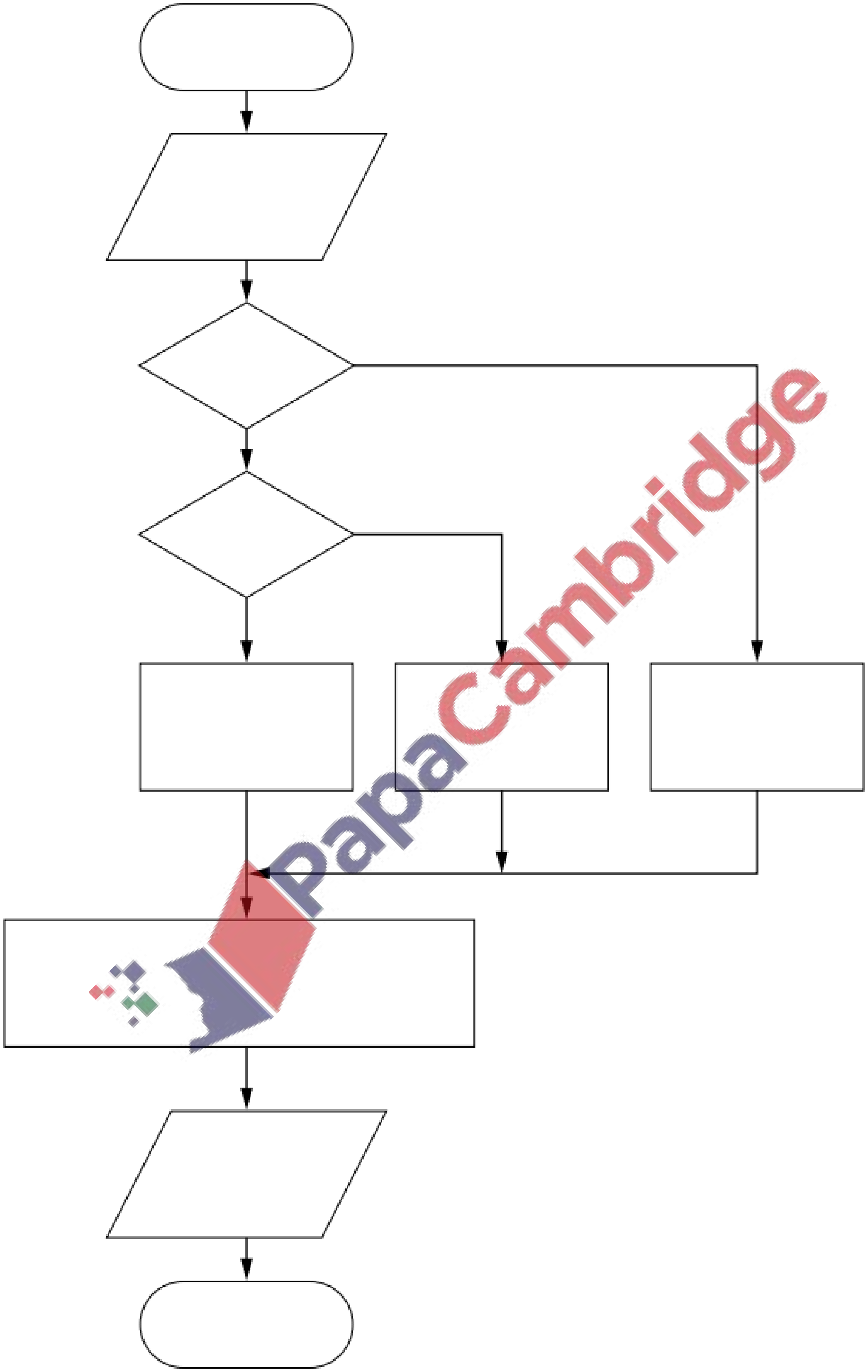
A purchase may qualify for a discount depending on the amount spent. The purchase price (`Purchase`), the discount rate (`DiscountRate`) and amount paid (`Paid`) is calculated as shown in the following pseudocode algorithm.

```
INPUT Purchase

IF Purchase > 1000
   THEN
      DiscountRate ← 0.10
   ELSE
      IF Purchase > 500
         THEN
            DiscountRate ← 0.05
         ELSE
            DiscountRate ← 0
      ENDIF
ENDIF

Paid ← Purchase * (1 - DiscountRate)
OUTPUT Paid
```

The algorithm is also to be documented with a program flowchart.

Complete the flowchart by:

- filling in the flowchart boxes
- labelling, where appropriate, lines of the flowchart



[6]

1 A programmer wants to write a program to calculate the baggage charge for a passenger's airline flight.

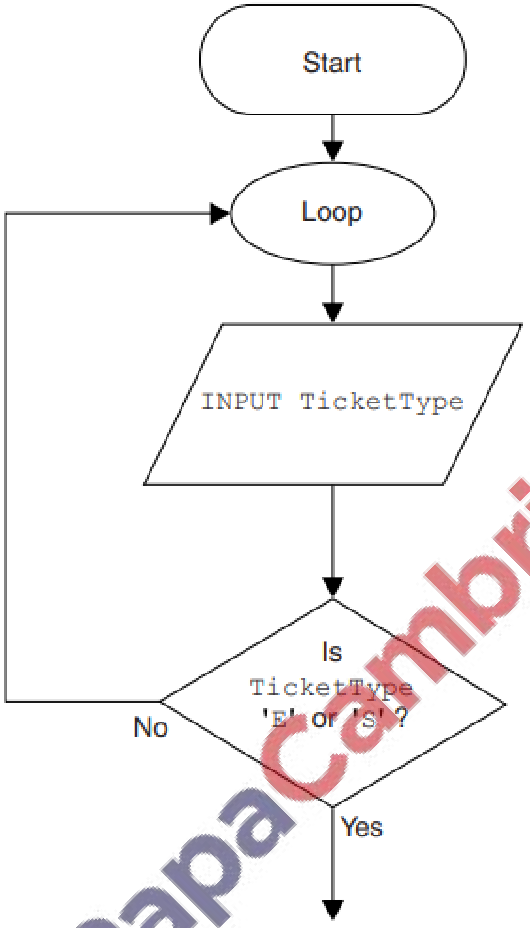
Two types of ticket are available for a flight:

- economy class (coded E)
- standard class (coded S)

Each ticket type has a baggage weight allowance as shown below. The airline makes a charge if the weight exceeds the allowance.

Ticket type	**Baggage allowance (kg)**	**Charge rate per additional kg ($)**
'E'	16	3.50
'S'	20	5.75

(c) The program design is to be amended. The value input by the user for the ticket type is to be validated. Part of the amended flowchart is shown below.



Write **pseudocode** to use a pre-condition loop for this validation.

[3]

5 Toni has a large collection of jazz CDs that are stored in different places. She wants to record where the CDs are stored. She decides to write a program to do this.

The program must store the data in a file, `MyMusic`.

(a) (i) Why is a file needed?

....

....[1]

(ii) `MyMusic` is a text file with the data for each CD as one line of text.

Data for a typical CD are:

Title:	Kind of Green
Artist:	Miles Coltrane
Location:	Rack1-5

The line will be formed by concatenating the three data items.

For the example above, the line stored will be:

```
Kind of GreenMiles ColtraneRack1-5
```

Describe a problem that might occur when organising the data in this way.

....

....

....

....

Describe a possible solution.

....

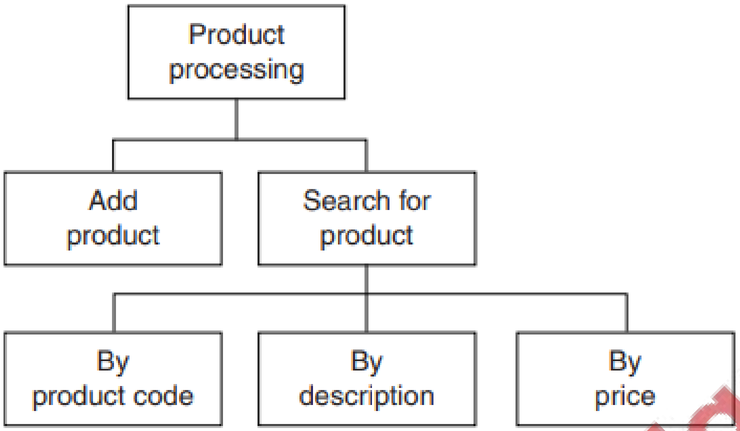
....

....

....

[4]

4 **(c)** The company maintains a file of product data. Ahmed is to write a program to add a new product and search for a product based on the structure diagram shown:



The program records the following data for each product:

- product code
- product description
- product retail price

The text file PRODUCTS stores each data item on a separate line, as shown below:

File **PRODUCTS**

```
0198
Plums(10kg)
11.50
0202
Onions(20kg)
10.00

0376
Mango chutney(1kg)
02.99

0014
Mango(10kg)
12.75
```

The program uses the variables shown in the identifier table.

Identifier	**Data type**	**Description**
PRODUCTS	TEXT FILE	Storing the code, description and retail price for all current products
PCode	ARRAY[1:1000] OF STRING	Array storing the product codes
PDescription	ARRAY[1:1000] OF STRING	Array storing the product descriptions
PRetailPrice	ARRAY[1:1000] OF REAL	Array storing the product retail prices
i	INTEGER	Array index used by all three arrays

(i) The first operation of the program is to read all the product data held in file `PRODUCTS` and write them into the three 1D arrays.

Complete the pseudocode below.

```
OPEN .......................................................................

i ← 1

WHILE ......................................................................

  READFILE ("PRODUCTS", ................................................)

  READFILE ("PRODUCTS", ................................................)

  READFILE ("PRODUCTS", ................................................)

  ..........................................................................

  ..........................................................................

ENDWHILE

CLOSE "PRODUCTS"

OUTPUT "Product file contents written to arrays"
```

[5]

When Ahmed designed the `PRODUCTS` file, he considered the alternative file structure shown opposite.

It stores one product per line in the text file.

File **PRODUCTS**

```
0198 Plums(10kg)             11.50
0202 Onions(20kg)            10.00

0376 Mango chutney(1kg) 02.99

0014 Mango(10kg)             12.75
```

(ii) State **one** benefit and **one** drawback of this file design.

Benefit ....

....

Drawback ....

....[2]

2 A program displays a menu with choices 1 to 4. The code to display the menu is written as the procedure `DisplayMenu`.

(a) Pseudocode which uses this procedure is:

```
CALL DisplayMenu
REPEAT
   OUTPUT "Enter choice (1..4)"
   INPUT Choice
UNTIL Choice >= 1 AND Choice <= 4
```

(i) Describe what this pseudocode will do.

....

....

....

....[3]

(ii) State why a loop is required.

....

....[1]

(b) The following pseudocode is a revised design.

```
CONSTANT i ← 3
CALL DisplayMenu
NoOfAttempts ← 0
REPEAT
   OUTPUT "Enter choice (1..4)"
   INPUT Choice
   NoOfAttempts ← NoOfAttempts + 1
UNTIL (Choice >= 1 AND Choice <= 4) OR NoOfAttempts = i
```

(i) Give the maximum number of inputs the user could be prompted to make.

.... [1]

(ii) State why this algorithm is an improvement on the one given in **part (a)**.

....

....[1]

5 **(c)** The company wants a program to output the total monthly sales for one of the selected websites.

The programmer codes a function with the following function header:

```
FUNCTION MonthlyWebSiteSales(ThisMonth : INTEGER, ThisSite : CHAR)
                                                     RETURNS INTEGER
```

The function returns the total number of bicycles sold for the given month and website.

The function will use the following:

Identifier	Data type	Description
`ThisMonth`	`INTEGER`	Represents the month number e.g. 4 represents April
`ThisSite`	`CHAR`	Coded as: • X for website X • Y for Website Y

(i) Give the number of parameters of this function. ....[1]

(ii) Some of the following function calls may be invalid.

Mark each call with:

- a tick (✓), for a valid call
- a cross (✗), for an invalid call

For any function calls which are invalid, explain why.

Function call	Tick (✓) / cross (✗)	Explanation (if invalid)
`MonthlyWebSiteSales(1, "Y")`		
`MonthlyWebSiteSales(11, 'X', 'Y')`		
`MonthlyWebSiteSales(12, 'X')`		

[3]

3 A program is to simulate the operation of a particular type of logic gate.

- The gate has two inputs (TRUE or FALSE) which are entered by the user.
- The program will display the output (TRUE or FALSE) from the gate.

The program uses the following identifiers in the pseudocode below:

Identifier	Data type	Description
`InA`	`BOOLEAN`	Input signal
`InB`	`BOOLEAN`	Input signal
`OutZ`	`BOOLEAN`	Output signal

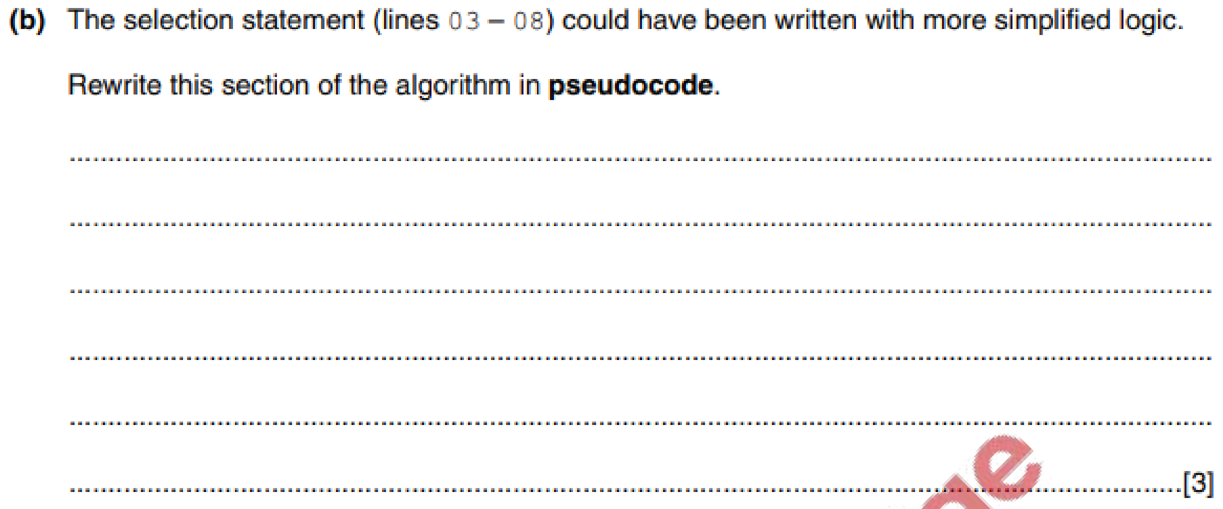
```
INPUT InA
INPUT InB
IF (InA = FALSE AND InB = FALSE) OR (InA = FALSE AND InB = TRUE)
                                    OR (InA = TRUE AND InB = FALSE)
  THEN
    OutZ ← TRUE
  ELSE
    OutZ ← FALSE
ENDIF
OUTPUT OutZ
```

(a) The programmer chooses the following four test cases.

Show the output (`OutZ`) expected for each test case.

Test case	Input		Output
	`InA`	`InB`	`OutZ`
1	`TRUE`	`TRUE`	
2	`TRUE`	`FALSE`	
3	`FALSE`	`TRUE`	
4	`FALSE`	`FALSE`	

[4]



(b) The selection statement (lines `03` – `08`) could have been written with more simplified logic.

Rewrite this section of the algorithm in **pseudocode**.

....

....

....

....

....

....[3]

Data:

Person 1 – 7

Person 2 – 110

Person 3 – 47

Person 4 – 74

Person 5 – 20

Person 6 – 10

268

Amend this pseudocode so that the output is the six salesperson numbers, their total and the final total.

```
FOR SalesPerson ← 1 TO 6
     FOR MonthNumber ← 1 TO 12
          TotalSales ← TotalSales + Sales[SalesPerson, MonthNumber]
     NEXT MonthNumber
NEXT SalesPerson
OUTPUT TotalSales
```

1 A programmer wants to write a program to calculate the baggage charge for a passenger's airline flight.

Two types of ticket are available for a flight:

- economy class (coded E)
- standard class (coded S)

Each ticket type has a baggage weight allowance as shown below. The airline makes a charge if the weight exceeds the allowance.

Ticket type	Baggage allowance (kg)	Charge rate per additional kg ($)
'E'	16	3.50
'S'	20	5.75

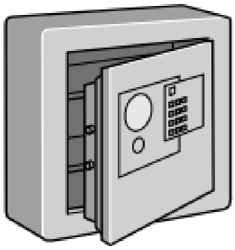
(b) The programmer needs data to test the flowchart.

Complete the table of test data below to show **five** tests.

`TicketType`	`BaggageWeight`	Explanation	Expected output
E	15	....	
		....	
		....	
		....	
		....	

[5]

1 A user can lock a safety deposit box by inputting a 4-digit code. The user can unlock the box with the same 4-digit code.



There is a keypad on the door of the safety deposit box. The following diagram shows the keys on the keypad.

1	2	3
4	5	6
7	8	9
R	0	Enter

Initially, the safety deposit box door is open and the user has not set a code.

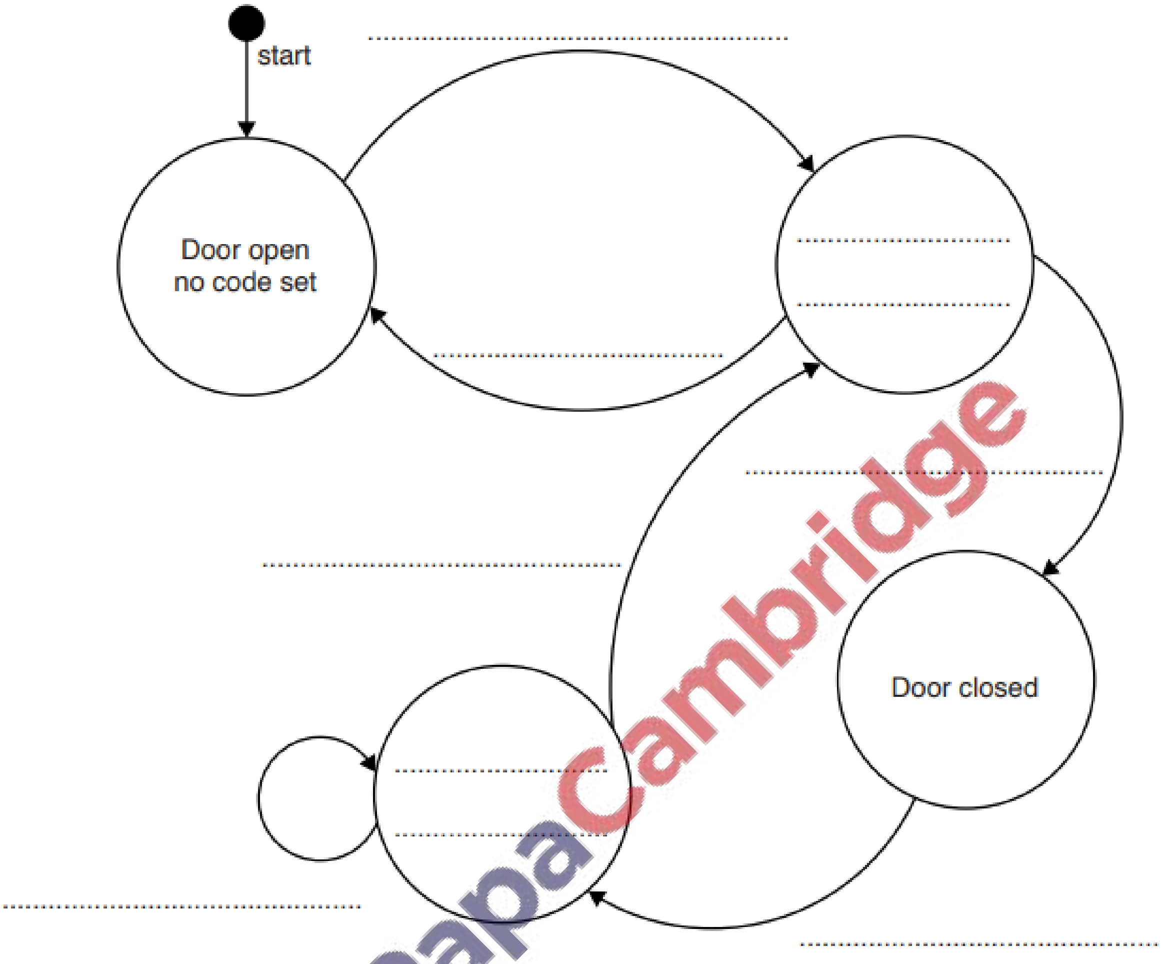
The operation of the safety deposit box is as follows:

A) To set a new code the door must be open. The user chooses a 4-digit code and sets it by pressing the numerical keys on the keypad, followed by the Enter key. Until the user clears this code, it remains the same. (See point E below)
B) The user can only close the door if the user has set a code.
C) To lock the door, the user closes the door, enters the set code and presses the Enter key.
D) To unlock the door, the user enters the set code. The door then opens automatically.
E) The user clears the code by opening the door and pressing the R key, followed by the Enter key. The user can then set a new code. (See point A above)

The following state transition table shows the transition from one state to another of the safety deposit box:

Current state	**Event**	**Next state**
Door open, no code set	4-digit code entered	Door open, code set
Door open, code set	R entered	Door open, no code set
Door open, code set	Close door	Door closed
Door closed	Set code entered	Door locked
Door locked	Set code entered	Door open, code set
Door locked	R entered	Door locked

(a) Complete the state-transition diagram.



[7]

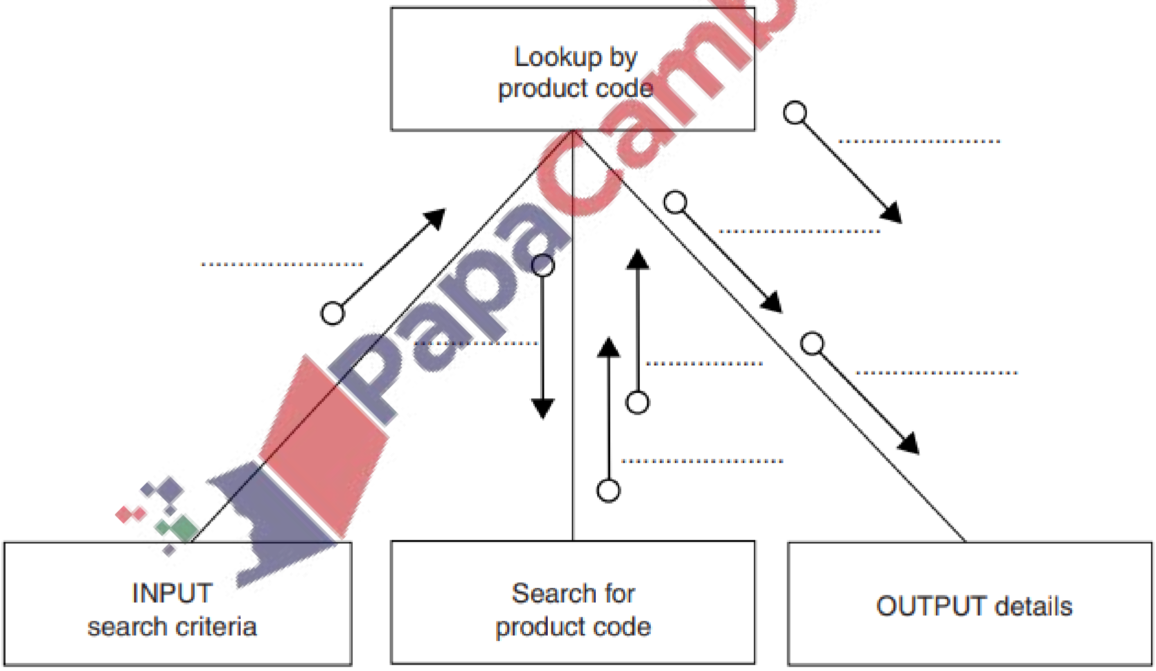
4 **(d)** To code the 'Search by product code' procedure, Ahmed draws a structure chart showing the different stages.

The procedure uses the variables shown in the identifier table.

Identifier	Data type	Description
`SearchCode`	`STRING`	Product code input by the user
`ThisIndex`	`INTEGER`	Array index position for the corresponding product
`ThisDescription`	`STRING`	Product description found
`ThisRetailPrice`	`REAL`	Product retail price found

You can assume that before the procedure is run, all the product data is read from file `PRODUCTS` and then stored in three 1D arrays as described in **part (c)(i)**.

Label the structure chart to show the input(s) and output(s).



[4]