2.4.3 Testing Strategies

May/June 2004

3. (c) A program has been written using a top-down technique.

The individual modules in the program have been fully tested and there are no errors in any of them.

Explain why the program may fail to run or may produce incorrect results, despite the testing that has been done. **[2]**

May/June 2007

```
6. D=1
INPUT X, E
B=E
C=E
    FOR I = 1 TO (X-1)
        INPUT A
        IF A>B THEN B = A
            ELSE IF A < C THEN C = A
            END IF
        END IF
        D = D + 1
        E = E + A
    NEXT
F = E/D
OUTPUT B, C, F
END
```

(a) State the output values of B, C and F for the following input test data

4, 6, 3, 7, 0 **[3]**

(b) Give three other different sets of test data, explaining what condition each is meant to test. **[3]**

2.4.3 Testing Strategies

May/June 2011. P21/P22

1 Ahmed, a designer, stores the following details of each job that he does in a file.

- job ID (a whole number between 1 and 1000 inclusive)
- job description
- price (greater than $10 and not more than $5000)
- expected completion date
- paid (yes/no)

(f) The code for the validation will have to be tested.

State four items of data you would use to test the JobID validation.

State the reasons for using that test data.

	JobID value	Reason
Test 1		
Test 2		
Test 3		
Test 4		

[8]

May/June 2011. P23

3 Kris has written a program that will work out the wages for her staff. The main steps for each employee are: to work out the hours worked, work out the total earnings, work out tax and finally print out how much will be taken home.

(d) For each employee the hours worked module collects data for five days. If they do not work on a particular day a zero is entered. Each person can work up to 9 hours a day for up to 5 days a week. The hours are added up; no-one may work more than 40 hours.

Write **five** sets of test data which test the module for different inputs/outcomes.

For each set of test data, give a reason for your choice.

2.4.3 Testing Strategies

	Day 1	Day 2	Day 3	Day 4	Day 5
Test 1					
Reason					
Test 2					
Reason					
Test 3					
Reason					
Test 4					
Reason					
Test 5					
Reason					

[10]

Oct/NOV 2011. P21

2 Nathan is designing a software solution for stock control in a mobile phone shop. He has a colleague, called Andre, who will help him write the program. Nathan decides to modularise the solution.

(g) One type of test data is invalid data.

(i) Name the other two types. **[2]**

(ii) Andre has written the StockOrdering module, which now needs testing.

- The StockID is a whole number between 1000 and 9999
- The ReOrderLevel is between 10% and 20%

Give six different items of test data, other than invalid data, which thoroughly test the two rules given above. Give a reason for each choice. **[6]**

2.4.3 Testing Strategies

StockID	ReOrderLevel	Reason
50		Invalid data for StockID
	21%	Invalid data for ReOrderLevel

Oct/NOV 2011 P22

1 (g) One type of test data is invalid data.

(ii) Nathan has written the PhoneSales module, which now needs testing.

- ContractLength, the number of months of the contract, can be only 12, 18 or 24
- FreeTexts, the number of free text messages per month must be in the range 0 to 600

Give six different items of test data other than invalid data which thoroughly test the two rules given above. Give a reason for each choice.

ContractLength	FreeTexts	Reason
20		Invalid data for ContractLength
	1000	Invalid data for FreeTexts

[6]

2.4.3 Testing Strategies

Oct/NOV 2011 P23

1 Nathan is designing a software solution for stock control in a computer shop. He has a colleague, called Andre, who will help him write the program. Nathan decides to modularise the solution.

(g) One type of test data is invalid data.

(i) Name the other two types. **[2]**

(ii) Andre has written the Sales module, which now needs testing.

- InvoiceNumber has the format yy-nnnn e.g. 11-0035 is the 35th invoice of the year 2011
- PromotionCode can be 'gold', 'silver' or 'bronze' only

Give six different items of test data, other than invalid data, which thoroughly test the two rules given above. Give a reason for each choice.

InvoiceNumber	PromotionCode	Reason
130092		Invalid data for InvoiceNumber
	glod	Invalid data for PromotionCode

[6]

2.4.3 Testing Strategies

May/June 2013. P21/22

2 The data for each record is validated as it is entered.

(c) (i) Meena uses three items of data to test this logic. In the table below enter 'normal' or 'borderline' in the empty cells.

HandInDate	Type of Data
31122014	.....
30142015	invalid
16062013	.....

(ii) State the reason why this invalid HandInDate is not a good test of the validation rules. **[1]**

(iii) State three hand-in dates that provide a better test to show that invalid data does not get entered. **[3]**

Oct/Nov 2013. P22

1 Jemma is designing a program that will work out the end of year bonuses for her employees. The main steps are:

- input employee's data
- calculate the bonus
- calculate deductions
  - tax
  - optional contribution to charity
- print out the bonus

(d) Jemma is designing a range validation check for the input of an employee's pay. The pay range depends on the employee's job type, which may be P(part-time), F(full-time) or C(commission only).

- A part-time employee will earn between $100 and $10000 a year.
- A full-time employee will earn between $5000 and $50000 a year.
- A commission only employee will earn between $0 and $80000 a year.

Complete the table showing five more rows of test data. Give a different reason for each, describing what is being tested.

2.4.3 Testing Strategies

Job type	Pay	Reason
F	25000	Normal data – within pay range for full-time

[5]

Oct/Nov 2014.P21/P23

1 Rema surveys the students in her class to find out which is the most popular sport.

She draws a tally chart:

1	Cricket	\\\\\\\
2	Football	\\
3	Tennis	\\\
4	Swimming	\\\\\\

Rema plans to collect sport data from students in the whole school. She designs a program to:

- input the number of the sport a student likes best (1, 2, 3 or 4)
- repeatedly ask for input until the input is 0 (zero)
- keep a count of each choice
- on completion of data entry, print out the results as a tally chart (as shown above)

Rema's first attempt is the following pseudocode:

```
Cricket ← 0
Football ← 0
Tennis ← 0
Swimming ← 0
REPEAT
    INPUT Choice
    CASE Choice OF
        1: Cricket ← Cricket + 1
```

2.4.3 Testing Strategies

```
            2: Football ← Football + 1
            3: Tennis ← Tennis + 1
            4: Swimming ← Swimming + 1
    ENDCASE
UNTIL Choice = 0
OUTPUT "Cricket ", Cricket
OUTPUT "Football ", Football
OUTPUT "Tennis ", Tennis
OUTPUT "Swimming ", Swimming
```

Her friend Aisha suggests that the pseudocode could be improved by:

- using a one-dimensional array, Tally, instead of four variables to store the counts
- modularising the design. The main program should just consist of three procedure calls:
  `InitialiseArrayCounts`
  `InputStudentChoices`
  `OutputTallyChart`

(d) Rema wants to test each module before she tests the whole program.

The first module she is going to test is the `OutputTally` procedure using different parameter values.

Give **three** different types of test data with an example of each.

Justify your choices.

Type of test data	Example test data	Justification

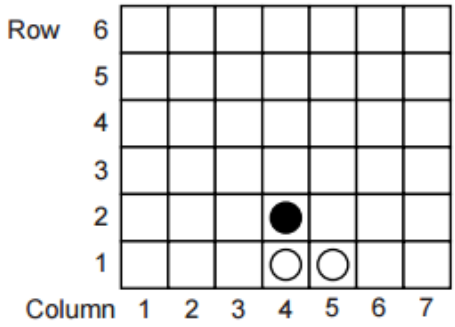
[9]

2.4.3 Testing Strategies

Oct/Nov 2014.P22

3 A game is played by two players. Player A uses white tokens (◯). Player B uses black tokens (●). The players take turns dropping tokens into a vertical grid. The tokens fall straight down and occupy the next available space in the chosen column. The aim of the game is to connect four of one's own colour tokens. This must be done in a vertical, horizontal or diagonal line.

Here is one example after Player A has had 2 turns and Player B has had 1 turn:



Nathan wants to write a program to allow two users to play the game on the computer.

The program will display a simplified version of the above grid which is redrawn after every turn.

(c) To drop a token into the grid, the player enters the chosen column number.

The function `ColumnNumberValid` has a parameter (`x`) which is the chosen column number. The function returns:

- `TRUE` if `x` is between 1 and 7 inclusive and there is still space in the column
- `FALSE` otherwise

(ii) Nathan wants to test the validation of the parameter, `x`, by this function.

Give **three** different types of test data with an example of each.

Justify your choices

2.4.3 Testing Strategies

Type of test data	Example test data	Justification

[9]

May/June 2015.P21/P22

3 A board game is designed for two players, O and X.

At the beginning, all cells of a 3 x 3 grid are empty.

The players take turns in placing their marker in an empty cell of the grid; player O always starts.

The game ends when one player completes a row, column or diagonal or the grid is full.

Here is one example after three turns:

		O
	O	X

Ali wants to write a program to play the game.

(b) Ali decides to validate the player input.

The input is valid if:

- the row and column numbers are within the range 1 to 3 inclusive
- the cell is empty

Ali chooses a sequence of **six** pairs of integer values to simulate player input. The test starts with an empty grid.

(i) Show the contents of the grid after the input of each pair of integer values. Circle whether the input is valid or invalid. If the input is invalid state the reason.

2.4.3 Testing Strategies

Row	Column	Grid content	Reason (if invalid)
2	2		valid / invalid
0	1		valid / invalid
1	1		valid / invalid
1	4		valid / invalid
4	1		valid / invalid
2	0		valid / invalid
2	2		valid / invalid

[6]

(h) When Ali has tested all individual modules he plans to do further testing.

Give **two** types of testing Ali should do. **[2]**

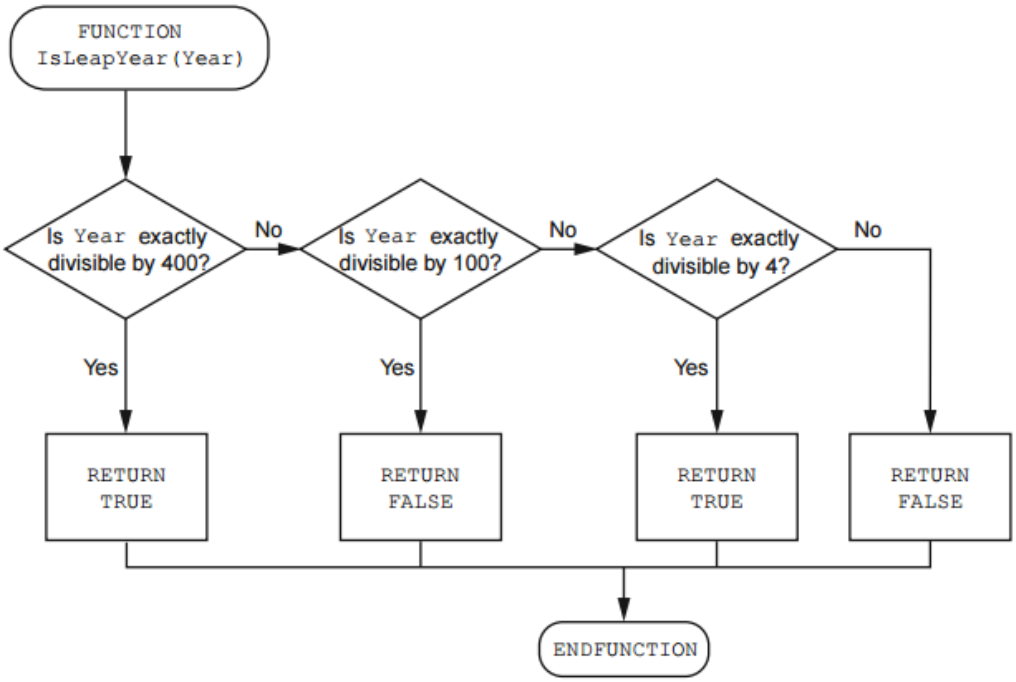
May/June 2015.P23

4 A leap year is a year with special numerical properties.

Ahmed is planning to write a function to check whether a year is a leap year.

He starts by drawing a flowchart.

2.4.3 Testing Strategies



(b) Ahmed wants to carry out white box testing of the function.

Give **four** integers which thoroughly test the function. For each one, give the expected return value and justify your choice.

2.4.3 Testing Strategies

	Year	Expected return value	Justification
1			....
2			....
3			....
4			....

[4]

(c) When Ahmed has tested the function, he plans to use it in a program.

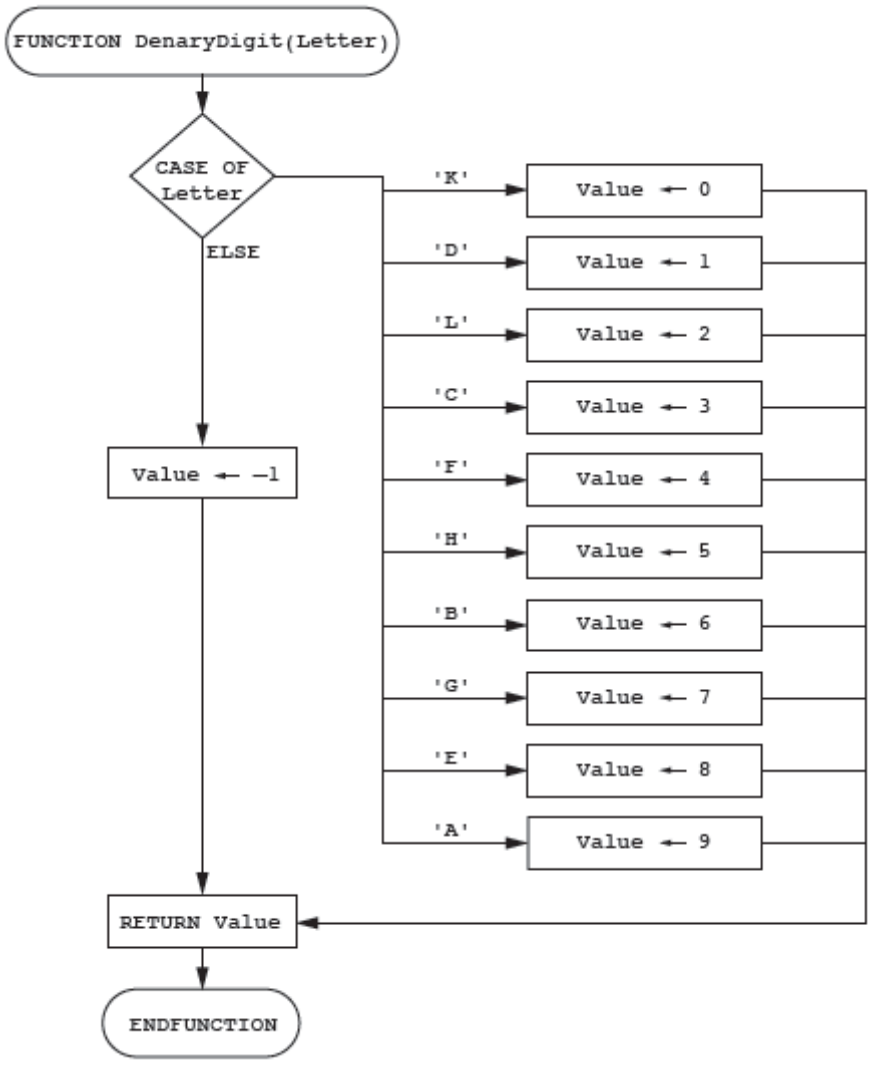
Give **two** types of testing that Ahmed could do with the completed program. **[2]**

Oct/Nov 2015.P21/P23

2 Alia received a number coded as a sequence of letters. She wants to write a program to change this sequence of letters back to the original number. She knows that each digit of the original number was replaced by exactly one letter. The letters used are shown in the flowchart below.
She has drawn the flowchart as part of the design for her solution.

The function `DenaryDigit(Letter)` returns an integer.

2.4.3 Testing Strategies



(a) (ii) Alia needs to test this function before using it in her program.
Complete the table of test data.

Letter	Expected result	Type of data (normal, borderline or invalid)
'1'		
'X'		
'G'		

[3]

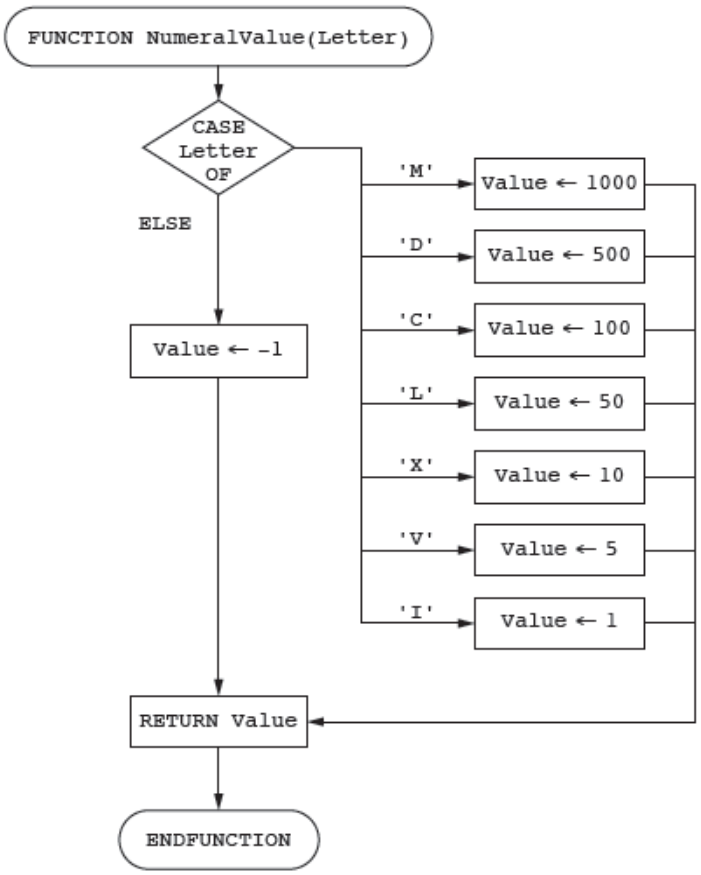
2.4.3 Testing Strategies

Oct/Nov 2015.P22

2 At the end of movies, when credits are listed, the year of production is often shown in Roman numerals.
Roman numbers are formed by combining letters together and adding their value. The letters used and their values are:

```
M: 1000
D: 500
C: 100
L: 50
X: 10
V: 5
I: 1
```

For example, `MMXV` is 1000 + 1000 + 10 + 5 = 2015.
Ali wants to write a program to convert a year written in Roman numerals to denary.
He has drawn the flowchart as part of the design for his solution.



(a) (ii) Ali needs to test this function before using it in his program.
Complete the table of test data.

2.4.3 Testing Strategies

Letter	Expected result	Type of data (normal, borderline or invalid)
'D'		
'V'		
'I'		
'Y'		

[3]

(d) The order of letters in Roman numbers is significant. Letters are placed from left to right, in order of value, starting with the largest.
However, a sequence of four identical letters (such as IIII) is shortened as follows.
If a letter to the left is of lower value than the letter to its right, the left letter's value becomes negative.
Examples are shown in the following table:

Roman number	Shortened Roman number	Interpretation	Denary
IIII	IV	-1 + 5	4
VIIII	IX	-1 + 10	9
XXXX	XL	-10 + 50	40
LXXXX	XC	-10 + 100	90
CCCC	CD	-100 + 500	400
DCCCC	CM	-100 + 1000	900

Ali has to amend his pseudocode to include the conversion of these shortened Roman numbers.
Ali has chosen some test data.
Complete the table.

RomanNumber	Expected result	Reason for choice
"MDCLXVI"	1666	Each letter used once in descending order
"CCC"	300	Multiple letters (but not 4 identical letters)
"IIII"		
"IV"		
"XIV"		
"XY"		

[4]

Computer Science (9608)

May/June 2015.P21/P22

1 A marathon runner records their time for a race in hours, minutes and seconds.

An algorithm is shown below in structured English.

INPUT race time as hours, minutes and seconds

CALCULATE race time in seconds

STORE race time in seconds

OUTPUT race time in seconds

(c) The program code will be tested using white-box testing.

(i) Explain what is meant by white-box testing. **[2]**

(ii) Complete the table heading.
Complete Test Number 1.
Add the data for Test Number 2 and Test Number 3.

	Input values				Output	
Test number	Race hours	Race minutes	Race seconds	.....	Total time (seconds)	Message
1	3	4	13	11053	11053	
2				11053		
3				11053		

[6]

Oct/Nov 2015.P21/P23

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

2.4.3 Testing Strategies

```
01 INPUT InA
02 INPUT InB
03 IF (InA = FALSE AND InB = FALSE) OR (InA = FALSE AND InB = TRUE)
                                       OR (InA = TRUE AND InB = FALSE)
04     THEN
05         OutZ ← TRUE
06     ELSE
07         OutZ ← FALSE
08 ENDIF
09 OUTPUT OutZ
```

(a) The programmer chooses the following four test cases.
Show the output (`OutZ`) expected for each test case.

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

[4]

Oct/Nov 2015.P22

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

- The gate has two inputs (0 or 1) which are entered by the user.
- The program will display the output (0 or 1) from the gate.

The program uses the following identifiers in the pseudocode below:

Identifier	Data type	Description
P	INTEGER	Input signal
Q	INTEGER	Input signal
X	INTEGER	Output signal

```
01 INPUT P
02 INPUT Q
03 IF (P = 1 AND Q = 0) OR (P = 0 AND Q = 1) OR (P = 0 AND Q = 0)
04 THEN
05 X ← 0
06 ELSE
07 X ← 1
08 ENDIF
09 OUTPUT X
```

2.4.3 Testing Strategies

(a) The programmer chooses the following four test cases.
Show the output (X) for each test case.

Test case	Input		Output
	P	Q	X
1	1	1	
2	1	0	
3	0	1	
4	0	0	

[4]

May/ June 2016. P23

6 A string-handling function has been developed.
For the built-in functions list, refer to the **Appendix** on the last page.
The pseudocode for this function is shown below.

```
FUNCTION SF(ThisString : STRING) RETURNS STRING
   DECLARE x : CHAR
   DECLARE NewString : STRING
   DECLARE Flag : BOOLEAN
   DECLARE m, n : INTEGER
   Flag ← TRUE
   NewString ← ""
   m ← LENGTH(ThisString)
   FOR n ← 1 TO m
       IF Flag = TRUE
          THEN
             x ← UCASE(MID(ThisString, n, 1))
             Flag ← FALSE
          ELSE
             x ← LCASE(MID(ThisString, n, 1))
       ENDIF
       NewString NewString & x
      IF x = " "
         THEN
           Flag ← TRUE
      ENDIF
   ENDFOR
   RETURN NewString
ENDFUNCTION
```

(b) Test data must be designed for the function SF.
(i) State what happens when the function is called with an empty string. **[1]**
(ii) The function should be thoroughly tested.
Give **three** examples of non-empty strings that may be used.
In each case explain why the test string has been chosen. **[3]**

2.4.3 Testing Strategies

Oct/Nov 2016. P21/P23

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.

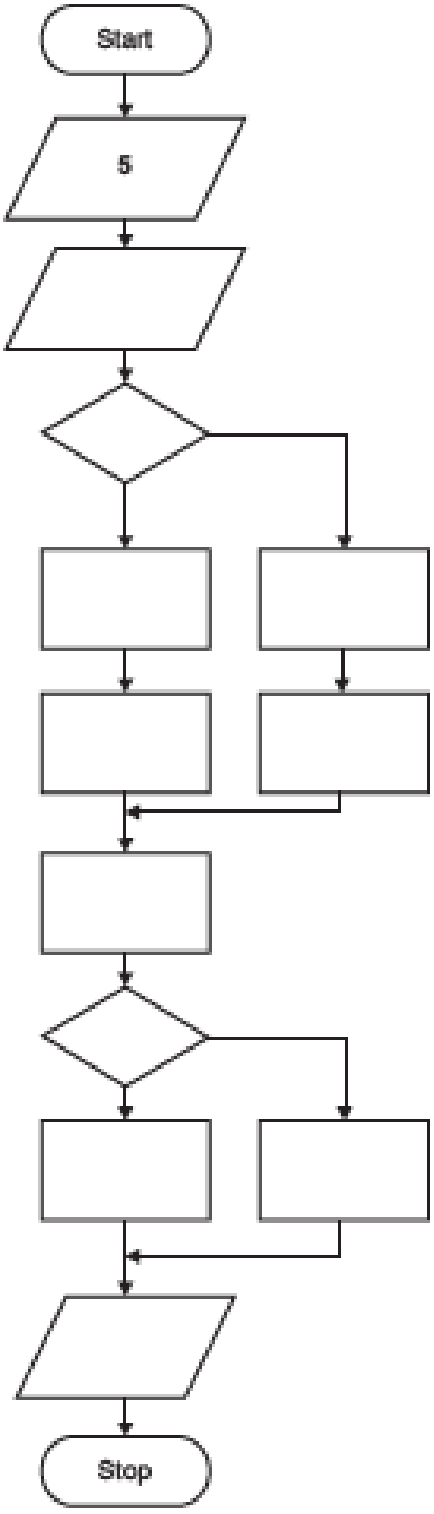
Ticket type	Baggage allowance (kg)	Charge rate per additional kg ($)
'E'	16	3.50
'S'	20	5.75

(a) A program flowchart will document the program. The flowchart will contain the following statements:

Statement number	Statement
1	`Charge ← 0`
2	`INPUT BaggageWeight`
3	`Charge ← ExcessWeight * ChargeRate`
4	`Is ExcessWeight > 0 ?`
5	`INPUT TicketType`
6	`ExcessWeight ← BaggageWeight - BaggageAllowance`
7	`BaggageAllowance ← 16`
8	`ChargeRate ← 3.5`
9	`OUTPUT Charge`
10	`ChargeRate ← 5.75`
11	`BaggageAllowance ← 20`
12	`Is TicketType = 'E' ?`

2.4.3 Testing Strategies

Complete the flowchart by putting the appropriate **statement number** in each flowchart symbol. Statement 5 has been done for you.



[6]

(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]

Oct/Nov 2016. P22

1 A number of players take part in a competition. The competition consists of a number of games.
Each game is between two players. The outcome of a game is that each player is awarded a grade (A, B, C or D). Each grade has an associated number of points as shown in the table below.

Grade	Points
A	0
B	1
C	3
D	5

The points total for all players is recorded. After each game is completed, the total number of points for each player is updated.

2.4.3 Testing Strategies

For example:

- before the game between Ryan and Karina, Ryan's total is 5 points and Karina's total is 3 points
- the result of the game between Ryan and Karina is: Ryan achieved grade B, Karina achieved grade D
- the players' points totals are updated to: Ryan has 6 and Karina has 8

When a player's points total reaches 12 or higher, that player is removed from the competition.
A programmer will write a program to update the player total after a game.
The program will output:

- the player's updated points total
- the message 'ELIMINATED' if the player is removed from the competition.

The programmer designs the identifier table below:

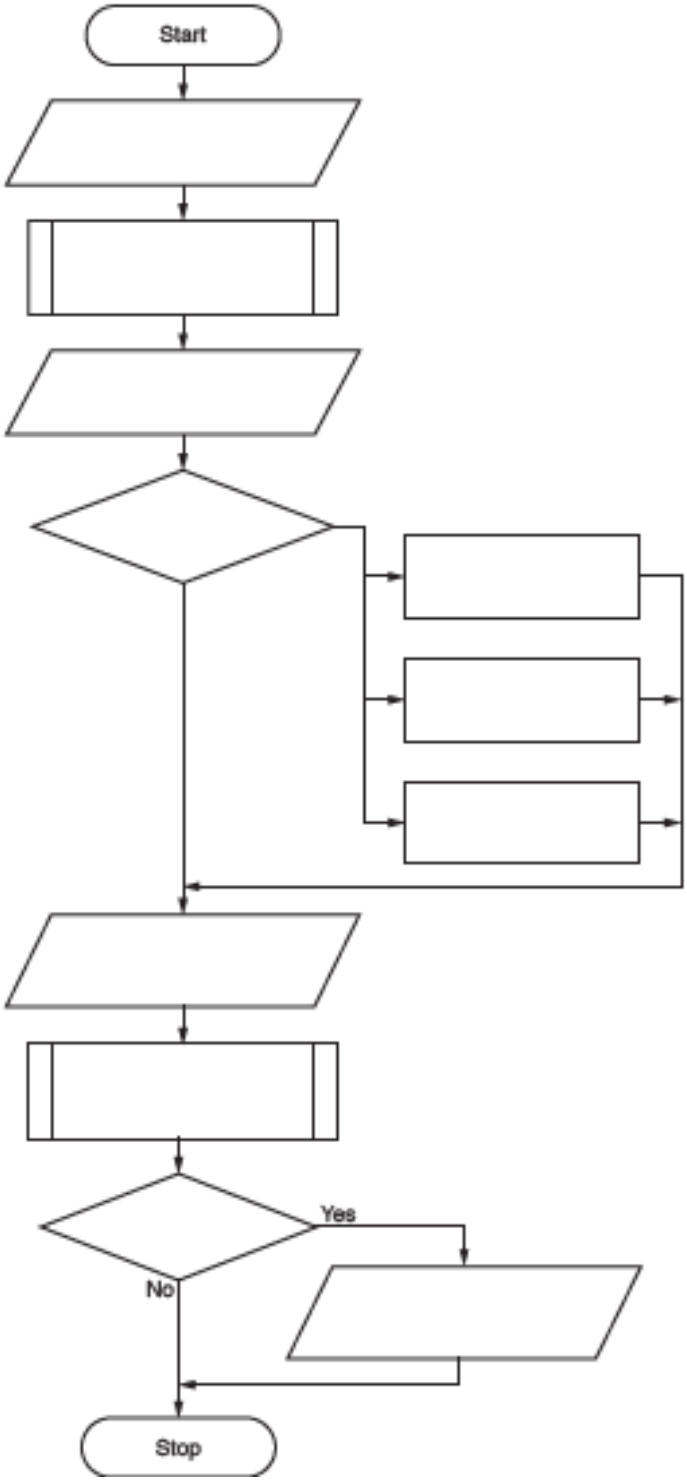
Identifier	Data type	Description
`PlayerName`	`STRING`	Name of the player
`PlayerGameGrade`	`CHAR`	Game grade for the player
`PointsTotal`	`INTEGER`	Current player points
`SavePlayerTotal`	procedure	Procedure has parameters `PlayerName` and `PointsTotal` and saves the updated player total
`ReadPlayerTotal`	function	Function has a parameter `PlayerName` and returns the current total for that player

(a) Complete the following program flowchart by:

- filling in the boxes, using pseudocode where appropriate
- labelling the lines of the flowchart, where necessary.

[9]

2.4.3 Testing Strategies



2.4.3 Testing Strategies

(b) Test data is to be produced to test the flowchart.
Complete the table of test data below to show **five** tests that should be used to test different paths through the flowchart.

Test data		Expected results	
`PointsTotal`	`PlayerGameGrade`	Updated `PointsTotal`	Output

[5]