2.1.1 Algorithms

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 2006

5. (b) A program is to be written which will update the records in a sequential file and then produce a backup copy.

Describe, using a diagram, the way that this problem can be split into modules to prepare it for coding. **[5]**

May/June 2009

A company specialises in creating websites for customers.

8. (a) As part of the process of designing a site, the company will use diagrams in order to make understanding easier.

Describe two types of diagram that may be used by the company. **[4]**

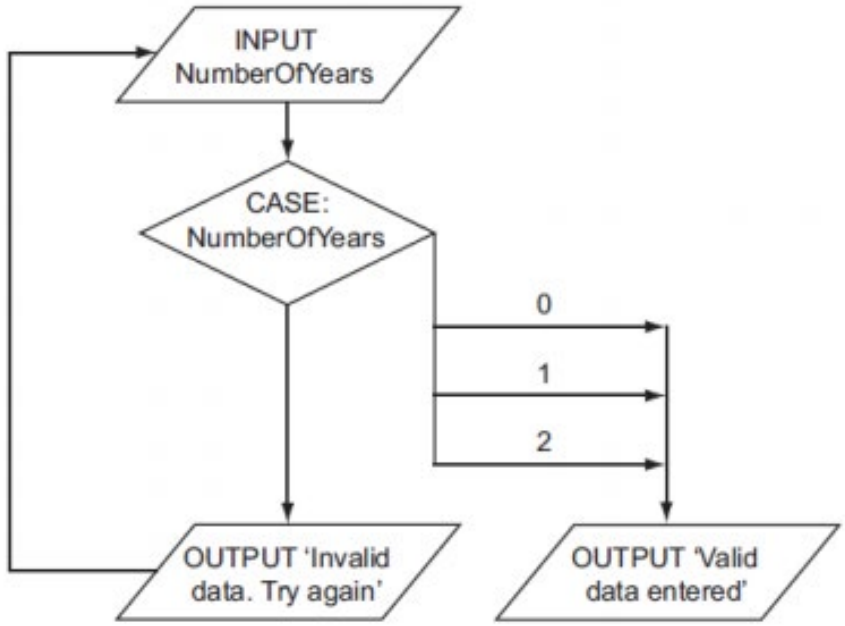
Oct/NOV 2011 P22

2 Ahmed is writing a program to record the data of members of the school football squad.

The input data will be validated. One input is the number of years a member has played for the team. This will be 0, 1 or 2.

The flowchart for the validation of number of years is shown below.

2.1.1 Algorithms

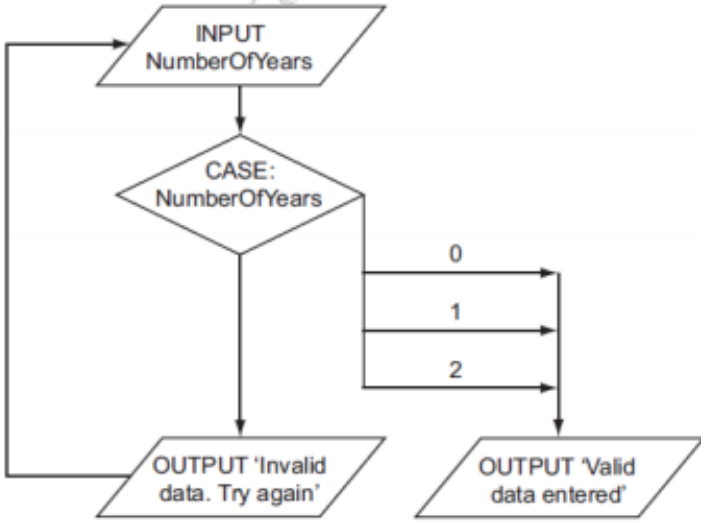


(a) (i) What is the output when the input is 2? **[1]**

(ii) What is the output when the input is 3? **[1]**

(e) Ahmed thinks it will be a good idea to allow only five attempts at getting the input data correct. If it is not a valid entry after five attempts, then a message 'Please check which values are allowed' should be output.

Modify the flowchart to include this additional check.



[5]

2.1.1 Algorithms

Oct/NOV 2012.P21

1 Soni works for a software house which has been asked to design software for a cycle hire company, Super Bikes.

Soni decides on the main tasks:

- collecting the information about new bikes
- entering details of repairs
- entering details of hirer
- entering details of payment

(c) State two reasons for dividing the main task into smaller tasks. **[2]**

May/June 2013. P21/22

1 Meena wants to develop a program to keep a record of her coursework assignments.

She will want to enter, sort and print out data.

She decides to modularise the solution.

(a) State two reasons why using modules is a sensible way for her to proceed.**[2]**

One way of storing her data will be to use a file of records.

Each record will contain at least the following data:

Data	Identifier	Description of input data
subject	Subject	Name of the subject, for example Physics
title	Title	Title of assignment
date set	DateSet	Format DDMMYYYY
hand-in date	HandInDate	Format DDMMYYYY
marked?	IsMarked	Y or N
date returned	DateReturned	Format DDMMYYYY
mark	Mark	Range 0 to 100

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

(b) The data input for HandInDate also needs validating.

It will be entered as DDMMYYYY.

- DD must be less than 32 and greater than 0

- MM must be less than 13 and greater than 0
- YYYY must be greater than 2012 and less than 2015

Draw a flowchart that shows the logic to validate the hand-in date. **[5]**

Oct/Nov 2013.P21

1 The Computing Department has a problem keeping track of its teaching resources. Juan, a student, has been asked to design and program a solution as his computing project. It will be the first large problem he has worked on.

He intends to write one large program that follows the process right through. His teacher tells him to break the problem into smaller parts.

(a) State and justify three of the reasons his teacher could give him for breaking the problem into smaller parts.

Juan decides that the design will include the following modules:

- update the resource file when a new teaching resource is purchased
  - input all the data about the resource
  - generate a resource ID for the resource
  - store in the resource file
- update the resource file when a current resource is discarded

(c) Name two features of a high-level programming language that help with this modular approach. **[2]**

(d) Juan realises that he will have to pass data from one module to another.

Explain how this is done. **[2]**

2 (b) Juan needs to design the code for a part of the program that determines where resources are kept. If the resource has:

- an ID less than 1001 it will be kept in Cabinet 1
- an ID between 1001 and 3000 it will be kept in Cabinet 2
  - even numbered IDs in Drawer 1
  - odd numbered IDs in Drawer 2
- an ID between 3001 and 5000 it will be kept in Cabinet 3

Write pseudocode that processes the variable ResourceID and outputs where the resource is kept.

Use nested IF statements. **[6]**

2.1.1 Algorithms

Oct/Nov 2013.P22

1Jemma 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

(c) (ii) Some modules require data values that originate from another module.

Explain the feature of a high-level programming language that enables this. **[2]**

Oct/Nov 2013.P23

1 The Science Department has a problem keeping track of its equipment. Ashvin has been asked to design and program a solution as his Computing Project.

(b) This project was the largest computing problem Ashvin had worked on. His friends wrongly advised him to program a solution as a whole. He decided to break the problem into smaller parts.

State and justify three of the reasons Ashvin gave his friends for breaking the problem into smaller parts. **[6]**

Ashvin decides the design will include the following modules:

- Update the equipment file when new equipment is bought.
  - Generate an equipment ID for an item of science equipment. ‘
  - Input all the data about the piece of equipment.
- Update the equipment file when old equipment is discarded.

(e) The module that stores all the equipment data needs to receive a data value produced in another module.

(i) What data value is this? **[1]**
(ii) Explain how this is done. **[2]**

2 (d) Ashvin needs to design the code for a part of the program that decides where equipment is kept.

- If the item has an ID less than 2000 it will be kept in the Physics Laboratory.
- If the item has an ID between 2001 and 4000 it will be kept in the Biology Laboratory.
- If the item has an ID between 4001 and 8000 it will be kept in the Chemistry Laboratory.
  - If the item has an ID ending in a zero it will have to be kept in a locked cabinet.

2.1.1 Algorithms

Write pseudocode that processes the variable EquipID and outputs where the equipment is kept, using nested IF statements. **[6]**

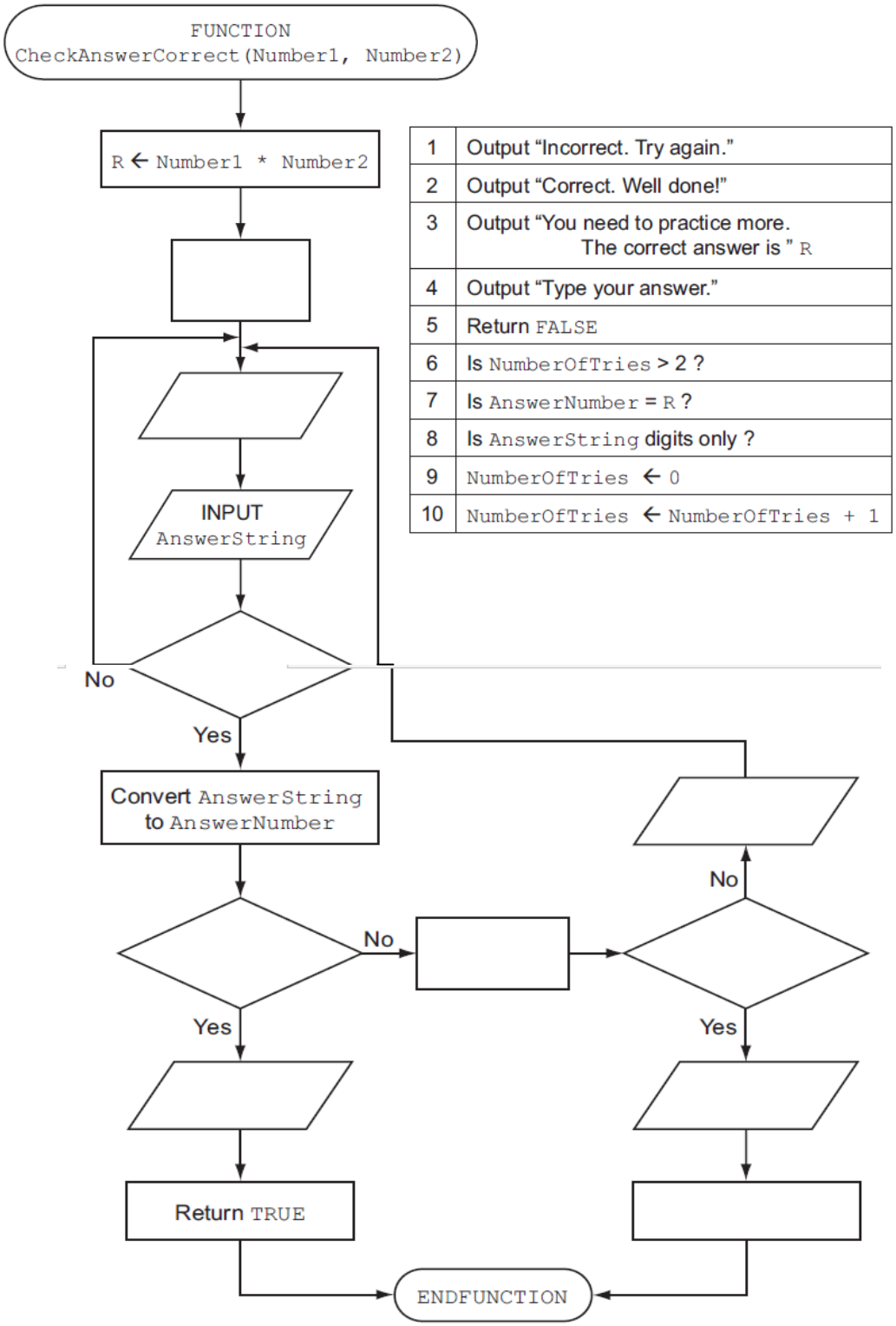
May/June 2014.P21

1 A teacher wants to write a program to help young children learn their multiplication tables.

(d) The function CheckAnswerCorrect gives the child three chances to type in the correct answer. The function returns TRUE if the child typed the correct answer, and FALSE if all three attempts are incorrect.

Complete the flowchart opposite, using the given statements. Label each blank symbol with the correct statement number.

2.1.1 Algorithms



1	Output "Incorrect. Try again."
2	Output "Correct. Well done!"
3	Output "You need to practice more. The correct answer is " R
4	Output "Type your answer."
5	Return FALSE
6	Is NumberOfTries > 2 ?
7	Is AnswerNumber = R ?
8	Is AnswerString digits only ?
9	NumberOfTries ← 0
10	NumberOfTries ← NumberOfTries + 1

[10]

2.1.1 Algorithms

May/June 2014.P23

4 A puzzle starts with a partially completed grid of digits. A player must fill a 9×9 grid with single digits so that each column, each row, and each of the nine 3×3 sub-grids contain all of the digits from 1 to 9.

Each puzzle has a unique solution.

Below is an example of a puzzle and its solution:

PUZZLE

8		5						7
9			5		4			
4	1			6				
			7			1	6	
1			4		6			3
	5	8			1			
				1			4	9
			2		7			1
2						5		6

SOLUTION

8	6	5	1	9	2	4	3	7
9	3	2	5	7	4	6	1	8
4	1	7	8	6	3	9	5	2
3	2	4	7	8	9	1	6	5
1	7	9	4	5	6	8	2	3
6	5	8	3	2	1	7	9	4
7	8	3	6	1	5	2	4	9
5	9	6	2	4	7	3	8	1
2	4	1	9	3	8	5	7	6

Raul wants to write a program that displays the puzzle and allows the user to enter digits to attempt a solution.

(c) When the user enters a character, the program needs to check it is a digit.

The character is stored in the character variable `Entry`.

Write the Boolean expression required to check that it is a digit. **[2]**

Oct/Nov 2014.P21

2 Ali sets up user IDs and passwords for his program.

When the user types in their user ID, the program looks up the stored password for this user ID.

The stored password is the encrypted version of the user's password.

(a) The program calls the function `PasswordCheck` with the stored password as parameter.

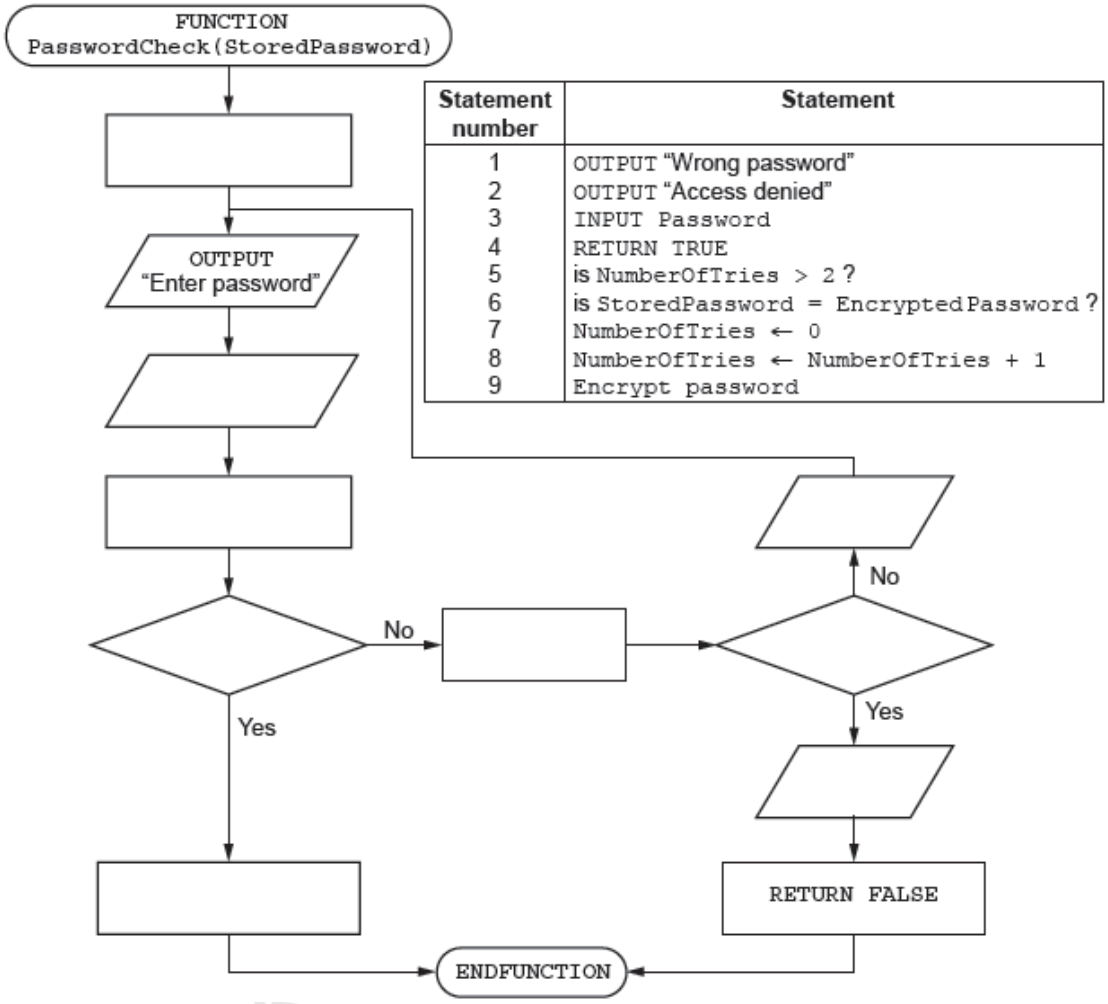
If the user enters the correct password the function returns the value `TRUE`.

Each time the password is entered incorrectly, the message "Wrong password" is output.

If the user enters an incorrect password 3 times, the user is told that access is denied, and the function returns the value `FALSE`.

2.1.1 Algorithms

Complete the flowchart using the given statements. Ensure that only statement numbers are present on the flowchart.



Statement number	Statement
1	OUTPUT "Wrong password"
2	OUTPUT "Access denied"
3	INPUT Password
4	RETURN TRUE
5	is NumberOfTries > 2 ?
6	is StoredPassword = EncryptedPassword ?
7	NumberOfTries ← 0
8	NumberOfTries ← NumberOfTries + 1
9	Encrypt password

[9]

3 Ahmed runs his own business. He lays floor tiles in rooms for customers. Ahmed wants a program that calculates how many tiles he needs when he inputs the measurements of the length and width of the room he is working on.

(b) The width of a room will measure at least 100 cm and less than 1000 cm.

The program must validate the width input.

Write a logic expression that is `TRUE` when the width stored in the variable `RoomWidth` is within the expected range. **[3]**

2.1.1 Algorithms

(c) Ali is going to write the program for Ahmed. The size of one floor tile is 30 cm × 30 cm. Ali knows that the following calculations are required:

```
TilesForWidth ← RoomWidth DIV 30

TilesForLength ← RoomLength DIV 30

TilesRequired ← TilesForWidth * TilesForLength
```

(i) If the room width measures 100 cm, give the value stored in `TilesForWidth`. **[1]**

If the room measurements are not exact multiples of 30 cm, the number of tiles must be rounded up so that Ahmed has enough tiles.

Ali knows that he can use the `MOD` operator to test for this.

(ii) Write a logic expression that is `TRUE` when the room width is not an exact multiple of 30 cm. **[1]**

May/June 2015.P21/P22

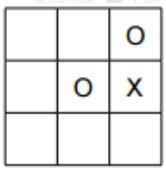
3 A board game is designed for two players, O and X.

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:



Ali wants to write a program to play the game.

(c) Ali uses the top-down design approach for his overall program solution.

His design is as follows:

```
01 GameEnd ← FALSE
02 CurrentPlayer ← 'O'
03 CALL DisplayGrid()
04
05 REPEAT
```

```
06 CALL PlayerTakesTurn(CurrentPlayer)
07 CALL DisplayGrid()
08 IF HasPlayerWon() = TRUE
09 THEN
10 GameEnd ← TRUE
11 OUTPUT "Player", CurrentPlayer, "has won"
12 ELSE
13 IF GridFull() = TRUE
14 THEN
15 GameEnd ← TRUE
16 OUTPUT "Draw"
17 ELSE
18 CALL SwapPlayer(CurrentPlayer)
19 ENDIF
20 ENDIF
21 UNTIL GameEnd = TRUE
```

(i) Identify one feature in the above pseudocode which indicates that top-down design has been used. **[1]**

(ii) State one benefit of top-down design. **[1]**

May/June 2015.P23

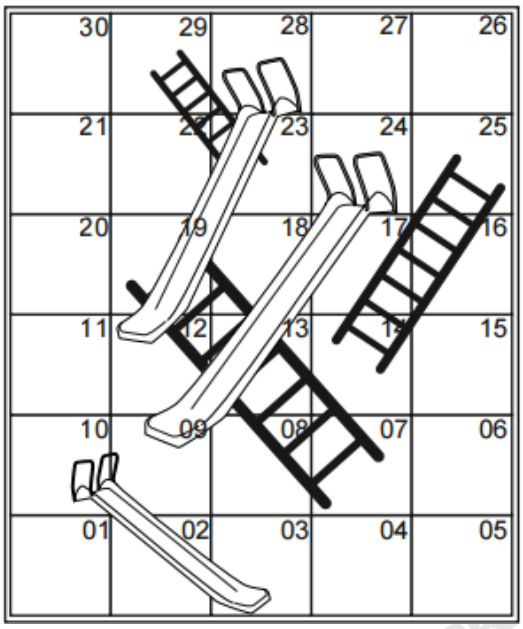
2 Ali has designed a board game.

The board consists of numbered squares.

Slides and ladders connect some of the squares.

An example layout of a board is:

2.1.1 Algorithms



The object of the game is for a player to move from the bottom square (Square 01) to the top square (Square 30).

A roll of a 6-sided die determines how many squares the player moves.

The table below shows example moves. These use the board layout above, with the player currently on Square 04.

Number rolled by die	Square moved to	Consequence	Final position
2	06	–	06
3	07	The base of a ladder, so climb to top of ladder	19
6	10	The top of a slide, so move down the slide	03

Ali plans to write a program to test different board designs. Each board design will have different numbers, positions and lengths for the slides and ladders.

The program is to test each board design to check that it is possible to complete the game in a reasonable number of moves.

For each board design, the program will simulate playing the game 1000 times and report the average number of moves.

(a) Ali starts the high level design of his program using pseudocode.

```
01 CALL InitialiseArray() // blank board
02 CALL InputBoardDesign() // add slides and ladders data
```

```
03 TotalMoves ← .....
04 FOR Game ← .....
05     // play next game and update TotalMoves
06     TotalMoves ← TotalMoves + NumberOfMovesInThisGame()
07  .....
08 AverageMovesPerGame ← .....
09 OUTPUT AverageMovesPerGame
```

(ii) Identify one feature in the pseudocode above which indicates that top-down design has been used. **[1]**

(iii) State one benefit of top-down design. **[1]**

Computer Science (9608)

May/June 2015.P23

1 Horses are entered for a horse race. A horse may have to carry a penalty weight in addition to the rider. This weight is added to the saddle. The penalty weight (if any) depends on the number of wins the horse has achieved in previous races.

The penalty weight is calculated as follows:

Number of previous wins	Penalty weight (kg)
0	0
1 or 2	4
Over 2	8

A program is to be written from the following structured English design.

1 INPUT name of horse

2 INPUT number of previous wins

3 CALCULATE penalty weight

4 STORE penalty weight

5 OUTPUT name of horse, penalty weight

2.1.1 Algorithms

(a) Complete the identifier table showing the variables needed to code the program.

Identifier	Data type	Description

[3]

(b) Line 3 in the algorithm above does not give the detail about how the race penalty weight is calculated; this step in the algorithm must be expressed in more detail.

(i) The algorithm above currently has five stages. One technique for program design is to further break down, where required, any stage to a level of detail from which the program code can be written. Name this technique. **[1]**

(ii) Write pseudocode for the given structured English design. **[5]**

Oct/Nov 2015. P21/P23

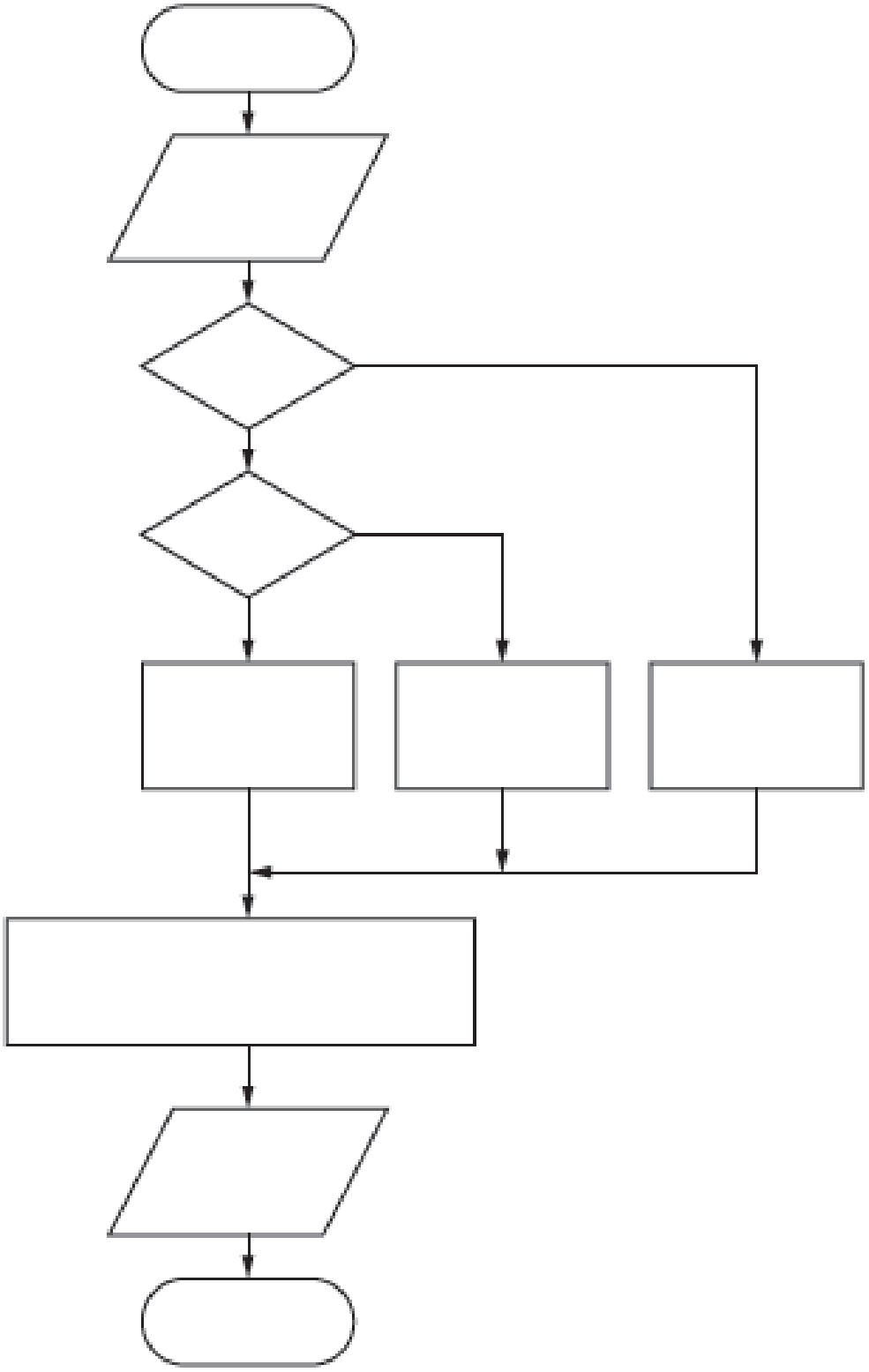
4 A program is to be written to calculate the discount given on purchases.
A purchase may qualify for a discount depending on the amount spent. The purchase price (`Purchase`), the discount rate (`DiscountRate`) and amount paid (`Paid`) is calculated as shown in the following pseudocode algorithm.

```
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

2.1.1 Algorithms



[6]

2.1.1 Algorithms

5 A driver buys a new car.
The value of the car reduces each year by a percentage of its current value.
The percentage reduction is:

- in the first year, 40%
- in each following year, 20%

The driver writes a program to predict the value of the car in future years.
The program requirements are:

- enter the cost of the new car (to nearest $)
- calculate and output the value of the car at the end of each year
- the program will end when either the car is nine years old, or when the value is less than$1000

(a) Study the incomplete pseudocode which follows in **part (b)** and fill in the identifier table.

Identifier	Data type	Description

[3]

Oct/Nov 2015. P22

3 Regular customers at a supermarket use a rewards card at the point-of-sale.
Points are calculated from every transaction and added to the points total stored on the card.
One reward point is given for every $1 spent.
When the points total exceeds 500, the customer can either:

- pay the full amount due and increase their points total
- get $1 deducted from the amount due in exchange for 500 reward points

The new points total and amount to be paid is printed on the receipt.
A program is to be written with the following specification:

- read the points total from the card
- process the amount spent
- output the amount to be paid and the new points total

A user-defined function `CalculatePoints` has already been coded to calculate the new points earned from the amount spent.
Study the following pseudocode:

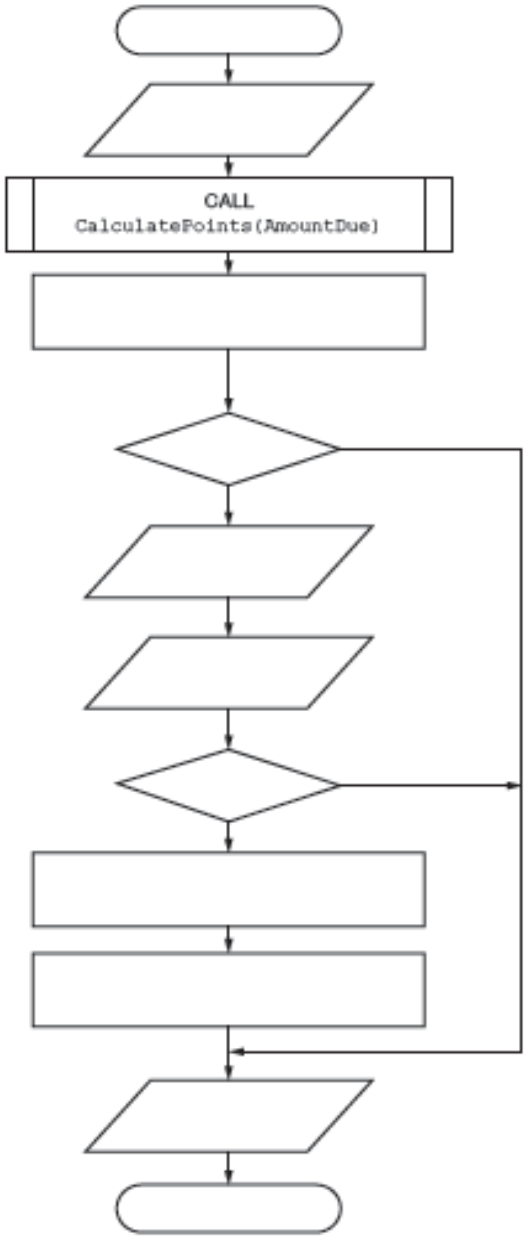
```
INPUT AmountDue

NewPoints← CalculatePoints(AmountDue)

PointsTotal← PointsTotal + NewPoints
IF PointsTotal> 500
THEN
OUTPUT "Exchange points?"
INPUT Response
IF Response = "YES"
```

2.1.1 Algorithms

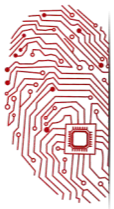
```
THEN
PointsTotal←  PointsTotal - 500
AmountDue←  AmountDue - 1
ENDIF
ENDIF
OUTPUT AmountDue, PointsTotal
```

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]



2.1.1 Algorithms

May/ June 2016. P21/P22

1 The items in the table below are individual statements in a generic programming language.
For the built-in functions list, refer to the **Appendix** on page 18.
(a) (i) Show what type of programming construct each statement represents.
Complete the table by putting a tick (✓) in the appropriate column for each item.

Item	Statement	Selection	Iteration	Assignment
1	`MyScore = 65`			
2	`FOR IndexVal = 0 TO 99`			
3	`MyArray[3] = MID(MyString,3,2)`			
4	`IF MyScore >= 70 THEN`			
5	`ENDWHILE`			
6	`ELSE Message = "Error"`			

[6]

(ii) State the purpose of each statement in the table in **part (a)(i)**.
Do **not** use mathematical symbols in your descriptions.

Item	Purpose of statement
1	....
2	....
3	....
4	....
5	....
6	....

[6]

2.1.1 Algorithms

2 A team is designing a software system to monitor temperature in a process. To do this, the system needs to sample the temperature repeatedly. If the temperature exceeds a given threshold value,an alarm will sound.

The system is to be software-based. It will include a subroutine, SampleTemp, which samples thetemperature and sets the alarm state to either ON or OFF.

The initial design stage will produce a prototype of SampleTemp with a user interface.
The structured English for this is:

1. IF the temperature does not exceed threshold value, SET alarm state to OFF
2. INPUT threshold value (to two decimal places)
3. INPUT sensor value (a whole number in the range 0 to 100)
4. MULTIPLY sensor value by conversion factor 1.135 to give temperature
5. IF temperature exceeds threshold value SET alarm state to ON
6. IF temperature exceeds threshold value OUTPUT message “Temperature Alarm”
7. IF temperature does not exceed threshold value OUTPUT message “Temperature OK”

(a) The procedure needs four variables. Complete the identifier table below for these variables.

Identifier	Data type	Description
AlarmState		....
SensorValue		....
ThresholdValue		....
Temperature		....

[4]

(b) Write the **pseudocode** equivalent of the structured English. Use the identifiers from the tablein **part (a)**. **[6]**

May/June 2016. P23

1 The items in the table below are statements from a program in a generic programming language.
For the built-in functions list, refer to the **Appendix** on the last page.
(a) (i) Show what type of programming construct each statement represents.
Complete the table by putting a tick (✓) in the appropriate column for each item.

2.1.1 Algorithms

Item	Statement	Selection	Iteration	Assignment
1	`WHILE DegF > 37.5`			
2	`MyName = "Gordon"`			
3	`DegF = INT(DegF)`			
4	`ENDIF`			
5	`CASE OF MyFavourite`			
6	`UNTIL x = 5`			

[6]

(ii) State the purpose of each statement in the table in **part (a)(i)**.
Do **not** use mathematical symbols in your descriptions.

Item	Purpose of statement
1	....
2	....
3	....
4	....
5	....
6	....

[6]

2.1.1 Algorithms

2 The engine management system of a car includes an energy-saving facility. When certain conditions are met, this facility will automatically stop the engine.
The system is to be software-based. It will include a subroutine, `EnergySaver`, which repeatedly takes data from sensors in the car. The subroutine decides whether or not to set the `EngineStop` value.
The table of identifiers used by this subroutine is shown below.

(a) Complete the identifier table below by stating the data types.

Identifier	Data type	Description
`Accelerator`	.....	Accelerator pedal position Values: 0 to 100 in steps of 1 Meaning: 0: none (not pressed) 100: maximum (fully pressed)
`EngineTemp`	.....	Engine temperature in °C (-50 to +150 stored to 1 decimal place)
`NormalTemp`	.....	Normal engine temperature in °C Whole number; typical value 90
`Speed`	.....	Road speed of car (in km/hr) Values: 0 to 200 in steps of 1
`EngineStop`	.....	Value used to signal engine must be stopped Possible values: TRUE: stop engine FALSE: run engine

[5]

The condition for stopping the engine is that all three of the following are true:

- Accelerator is not pressed
- Engine temperature is normal or above
- Car speed is zero

The initial design stage will produce a prototype of `EnergySaver`, with a user interface.
The structured English for this is:

1. INPUT value for accelerator pedal position
2. INPUT value for engine temperature
3. INPUT value for normal engine temperature
4. INPUT value for car speed
5. EVALUATE engine stopping condition

6. IF stopping condition satisfied SET engine stop value to TRUE
7. IF stopping condition not satisfied SET engine stop value to FALSE
8. OUTPUT message indicating engine stop value

(b) Write the **pseudocode** equivalent of the structured English. Use the identifiers from the table in **part (a)**. For the built-in functions list, refer to the **Appendix** on the last page. **[6]**

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.

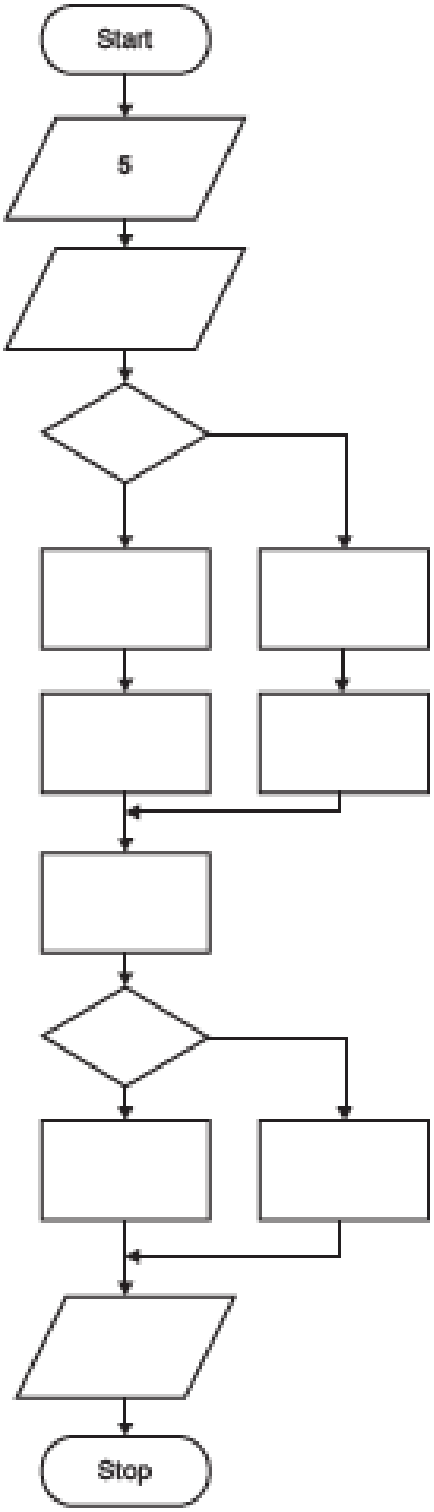
Ticket type	Baggage allowance (kg)	Charge rate per additional kg ($)
'E'	16	3.50
'S'	20	5.75

(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' ?`

2.1.1 Algorithms

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



[6]

2.1.1 Algorithms

2 A sensing device sends bit values to a computer along data channels.

- Channel 1 transmits a sequence of binary values from a sensor
- Channel 2 transmits at regular intervals to indicate whether the sensor is switched on or off:
- 0 indicates switched off
- 1 indicates switched on

A program tests the bits received from the sensing device.
A program reads the signal from Channel 2 after every six values from Channel 1.
A built-in function `READ(<ChannelNumber>)` reads a value from the specified channel.
Pseudocode for the program is as follows:

```
01 BitCount 0
02 Status2 READ(2)
03 WHILE Status2 = 1
04
05    FOR ReadingCount 1 TO 6
06          ThisBit READ(1)
07          IF ThisBit = 1
08                THEN
09                      BitCount BitCount + 1
10          ENDIF
11          IF BitCount = 5
12                THEN
13                      OUTPUT "Error - Investigate"
14                      BitCount ← 0
15          ENDIF
16    ENDFOR
17
18    Status2 READ(2)
19 ENDWHILE
```

(b) Identify the following constructs in the given program, using line numbers.
For multi-line constructs give the first line number only.

Construct	Line number
Assignment	
Selection	
Iteration	

[3]

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.

2.1.1 Algorithms

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.
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:

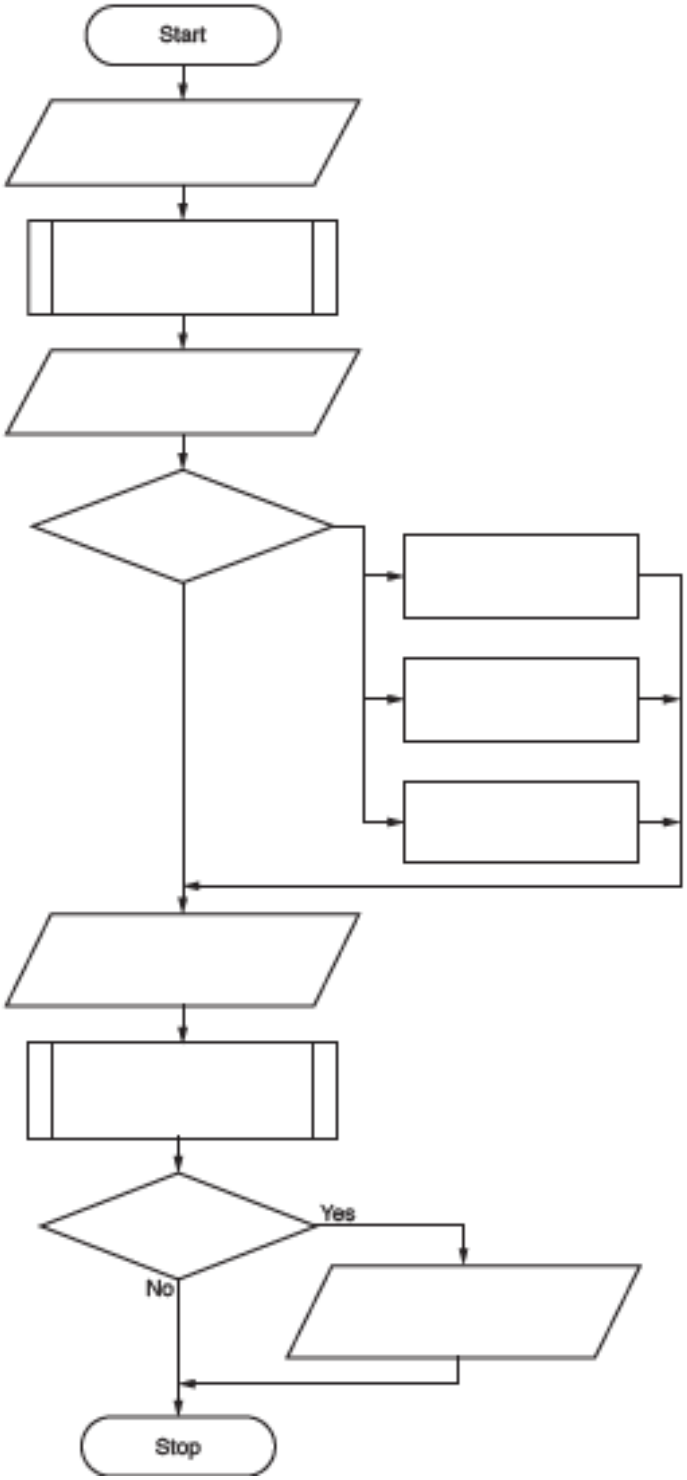
Identifier	Data type	Description
`PlayerName`	`STRING`	Name of the player
`PlayerGameGrade`	`CHAR`	Game grade for the player
`PointsTotal`	`INTEGER`	Current player points
`SavePlayerTotal`	procedure	Procedure has parameters `PlayerName` and `PointsTotal` and saves the updated player total
`ReadPlayerTotal`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

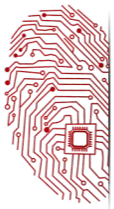
(a) Complete the following program flowchart by:

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

[9]

2.1.1 Algorithms





2.1.1 Algorithms

May/June 2018. P21

1 (a) A program stores data about hospital patients.
Give a suitable **identifier name** for each of the data items.

Description of data item	Suitable identifier name
The temperature of the patient	
The temperature of the room	
The patient identification number	
The name of the nurse taking the measurement	

[4]

3 In a chemical factory, a procedure, `CheckSensor()` is required to allow an operator to monitor the temperature in different locations.
In the factory:

- the temperature is measured by 10 sensors, each at a different location
- each sensor has a unique ID (1 to 10).

The procedure `CheckSensor()` will compare the measured temperature against each of two constant values, `LowTemp` and `HighTemp`. It will perform the following actions depending on the result of the comparison.

Measured temperature	Action
below `LowTemp`	Output `"Cold"`
from `LowTemp` to `HighTemp`	Output `"Normal"`
above `HighTemp`	Call procedure `Alarm()`

A library function, `GetTemp()`, returns the temperature value from a given sensor.
The structured English representing the algorithm for the procedure `CheckSensor()` is as follows:
1. Prompt for the input of a sensor ID.
2. Input a sensor ID.
3. If the sensor ID is invalid, repeat from step 1.
4. Call the `GetTemp()` function with the sensor ID as the parameter, to obtain the relevant temperature.
5. Compare the temperature against the two constant values and take the appropriate action.

Draw a program flowchart on the next page to represent the algorithm for procedure `CheckSensor()`.
Variable declarations are not required in program flowcharts. **[8]**