Computer Science 2210 (Notes)

Chapter: 2.1 Algorithm design and problem-solving

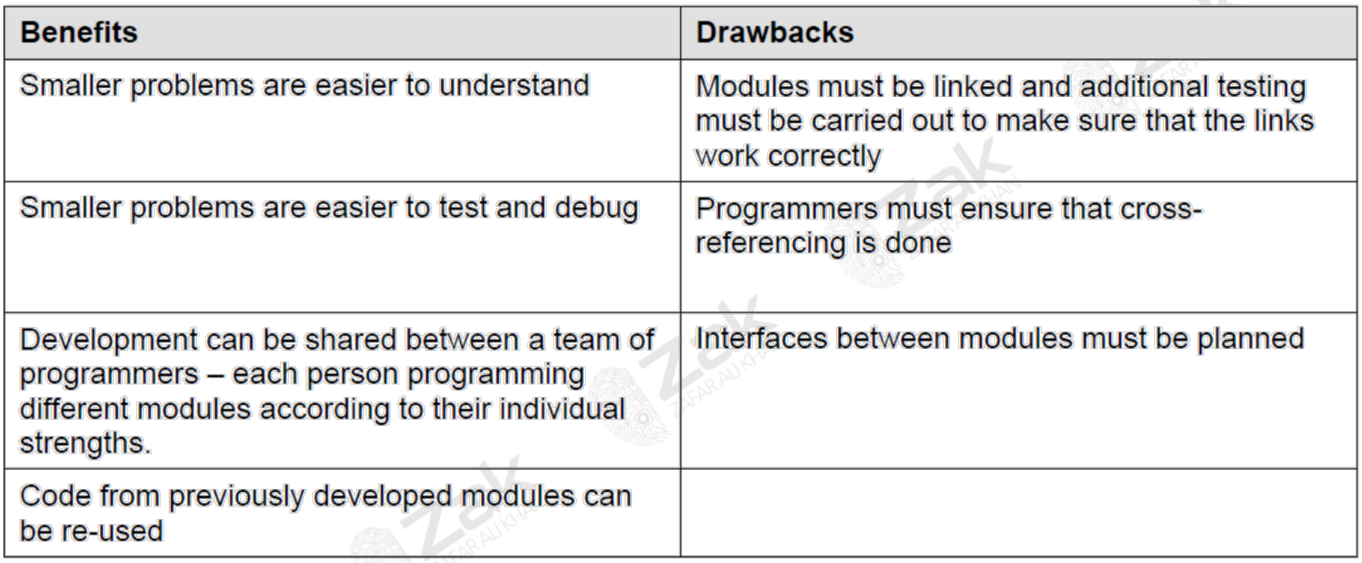
Topic: 2.1.1 Problem-solving and design

Top down/modular design

1. Advantages of designing a solution to a problem by splitting it up into smaller problems.
2. Produce and describe top-down/modular designs using appropriate techniques, including structure diagrams, showing stepwise refinement

Definition: A Top-down design is when a problem is split into smaller sub-problems, which themselves are split into even smaller sub-problems until each is just one element of the final program.

Benefits and drawbacks of modular programs

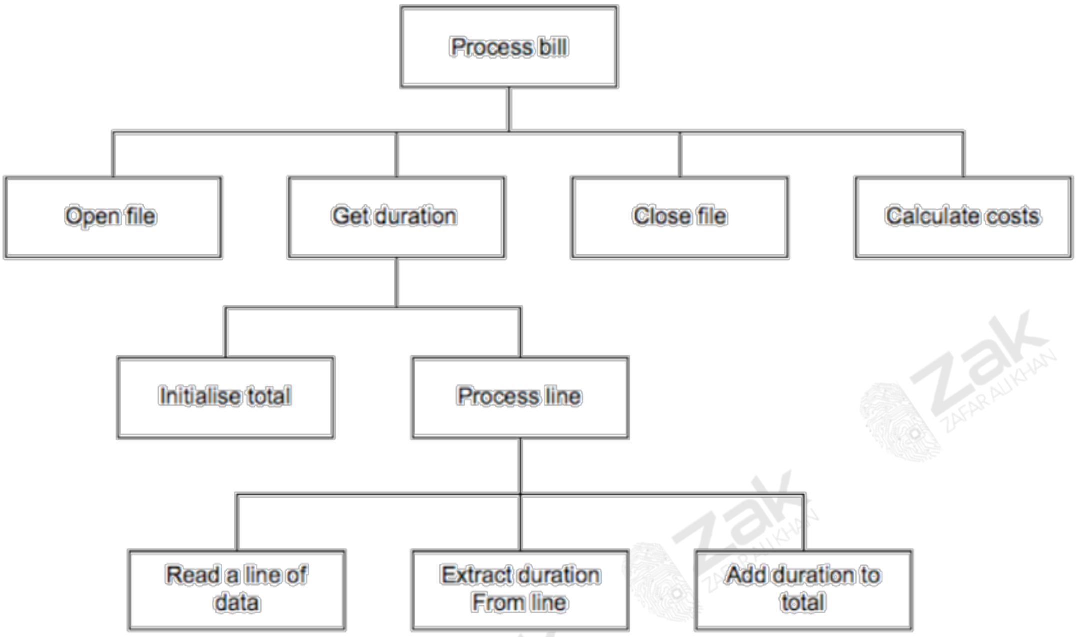


Benefits	Drawbacks
Smaller problems are easier to understand	Modules must be linked and additional testing must be carried out to make sure that the links work correctly
Smaller problems are easier to test and debug	Programmers must ensure that cross-referencing is done
Development can be shared between a team of programmers – each person programming different modules according to their individual strengths.	Interfaces between modules must be planned
Code from previously developed modules can be re-used	

Topic: 2.1.1 Problem-solving and design

Structure diagrams

A structure diagram is a pictorial representation of a modular system.



Stepwise refinement

Stepwise refinement is the process of developing a modular design by splitting a problem into smaller sub-tasks, which themselves are repeatedly split into even smaller sub-tasks until each is just one element of the final program.

Topic: 2.1.1 Problem-solving and design

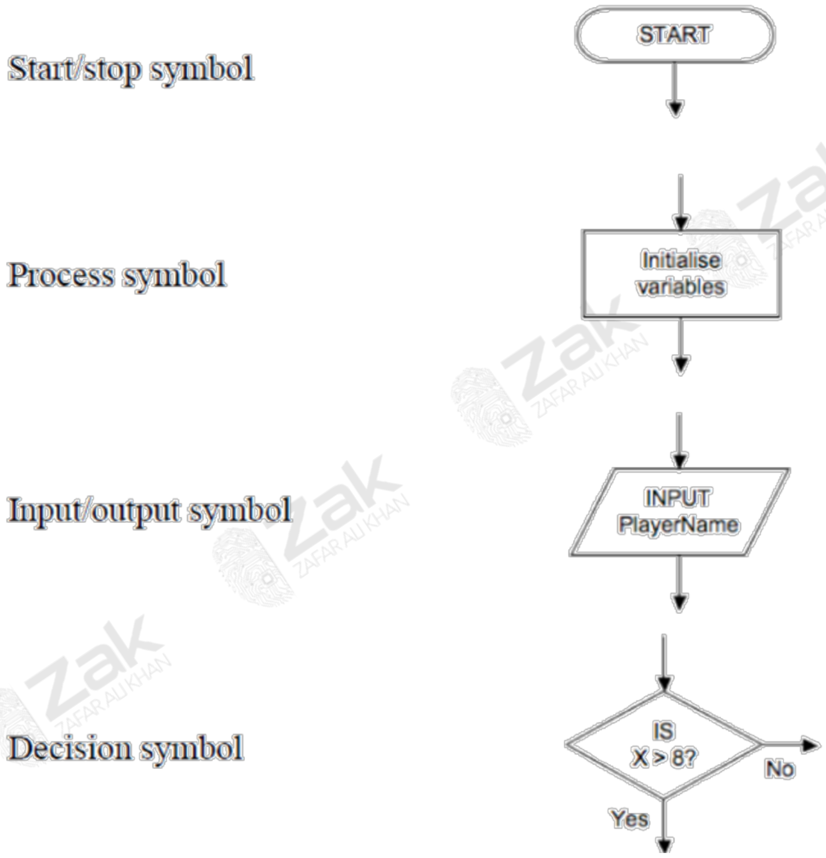
1. produce an algorithm for a given problem either in pseudocode form or the flowchart form
2. understand algorithms presented in the form of program flowcharts and pseudocode

What is an Algorithm?

An algorithm is a sequence of steps, which perform a specific task. In computing, algorithms are usually represented as a program flowchart, or in pseudocode.

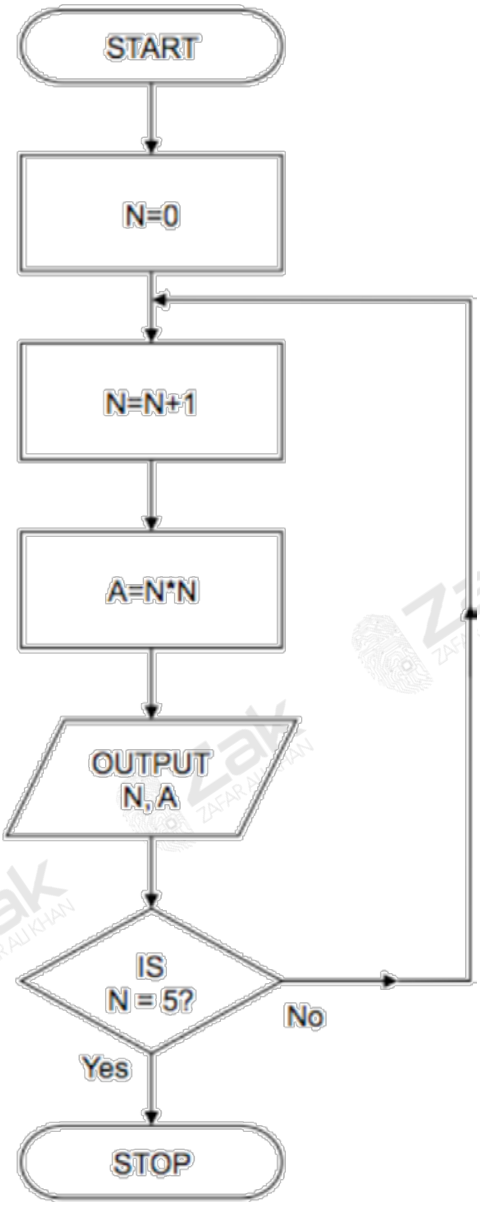
What is a program flowchart?

A program flowchart is a pictorial representation of an algorithm. Program flowcharts consist of special symbols:



Topic: 2.1.1 Problem-solving and design

Here is an example of a flowchart to output the first five squared numbers:



Topic: 2.1.1 Problem-solving and design

What is Pseudocode?

Pseudo-code is a simplified form of programming code that uses common programming terminologies, but does not use the strict syntax rules of a programming language.

An example of a pseudocode algorithm:

```
{
BEGIN
      INPUT CardNumber
      REPEAT
            INPUT PIN
            IF PIN is wrong for this CardNumber THEN
                  OUTPUT “Wrong Pin”
            END IF
      UNTIL Pin is correct
      INPUT Amount
      IF there are enough funds THEN
            Dispense Cash
            Update customer’s balance
      ELSE
            OUTPUT “Sorry, insufficient funds”
      END IF
END
}
```

Topic: 2.1.1 Problem-solving and design

Algorithms should be evaluated using the following criteria:

1. Efficiency
2. Correctness
3. Appropriateness

Efficiency

An algorithm's efficiency can be judged in terms of:

- **Speed**: How quick the algorithm produces the required output.
- **Memory requirements**: How many lines of code the algorithm contains.

Correctness

Although an algorithm is expected to produce the correct outputs, correctness can still be measured in terms of:

- **Accuracy** : How many decimal places produce output with greater accuracy (e.g. more decimal places)
- **Range**: Will the algorithm work with the complete range of inputs? Or can it only deal with positive numbers, whole numbers, numbers below 1 million, etc.
- **Reliability**: Will the algorithm always produce correct output within the range that it is designed to work? Or are there values which it will not accept
(e.g. zero).

Appropriateness

Appropriateness can be measured in terms of:

- **Length**: If the problem is simple then a short algorithm would normally be expected.
- **Speed**: if the output needs to be generated quickly, then the algorithm must be able to generate output quick enough.
- **Memory requirements**: An algorithm controlling a washing machine must not require a lot of memory!

Topic: 2.1.1 Problem-solving and design

Flow Charts

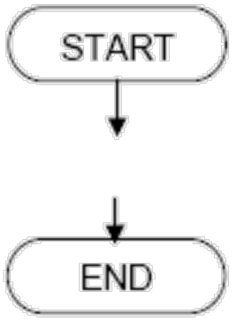
This section covers the use of flow diagrams (charts) in the production of algorithms. This topic must not be confused with “Systems flowcharts” as those are different and these are covered in a different section (Systems analysis).

This section primarily covers four areas:

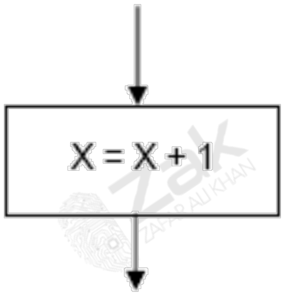
1. Common flow chart symbols
2. Writing flowcharts to solve problems
3. Dry running of flowcharts to determine its function and outputs
4. Exercises to test the above concepts
5. Topical past paper questions

Common flowchart symbols

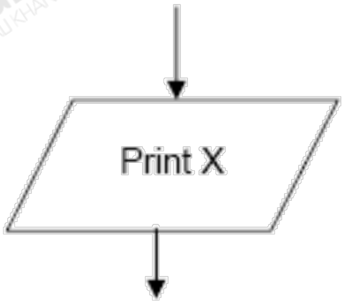
1.1 The start and end box:



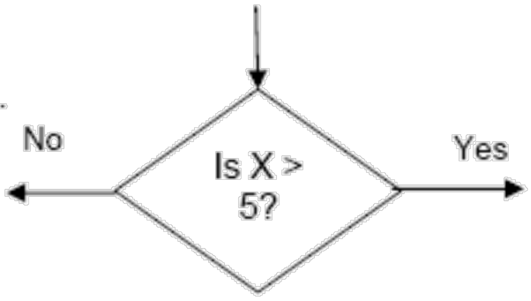
1.2 The process box:



1.3 Input/Output box:



1.4 Decision/query box:



Topic: 2.1.1 Problem-solving and design

Understand and use assignment statements

What is an Assignment?

An assignment is an instruction in a program that places a value into a specified variable.
Some typical assignments are:

```
TheLength ← 20.5
TheUsersName$ ←  “Charlie”
TheArea ← TheWidth * TheLength
TotalCost ← LabelledCost + 15
Counter ← Counter + 1
```

Note that the last example is a common method used to increment the value of a variable. It could be read as:

(The new value of “Counter” is its existing value plus one)

Topic: 2.1.1 Problem-solving and design

Arithmetic operators including operators for integer

Addition, subtraction and multiplication

Operator	Meaning/purpose	Example	Result
+	Addition	`Result=3+5`	8
–	Subtraction	`Result=3–7`	–4
*	Multiplication	`Result=3*7`	21

Powers

Operator	Meaning/purpose	Example	Result
^	Power	`Result=3^2`	9

Division

A result of a division such as 17 ÷ 4 can be expressed either as a real (4.25) or as two integers (4 remainder 1).

The integer method, in most programming languages, uses the operators DIV and MOD.

Operator	Meaning/purpose	Example	Result
/	Division	`Result=14/5`	2.8

DIV	Integer Division – returns the result of a division after ignoring the decimal portion of all numbers	`14 DIV 5`
		`7.2 DIV 3.9 (= 7 DIV 3)`
			2
			2

MOD	Remainder – returns the remainder of the division of two integers	`14 MOD 5`
		`18 MOD 6`
		`3 MOD 7`
			4
			0
			3

Topic: 2.1.1 Problem-solving and design

Relational operators, eg. =, <, <=, >, >= and <>

Relational operators are used in the format: [Expression] [Operator] [Expression] and will always return a Boolean (True or False) value.

Relational operators are typically used with the "IF" selection and also within conditional loops (REPEAT-UNTIL or WHILE-WEND).

In the following table, the variables "a" and "name$" have the following assignments:

```
a←3+5
name$← "JAMES"
```

Operator	Meaning/purpose	Example	Result

=	Equal – tests if two expressions are identical	IF a=8
		IF name$="JANET"
		IF a=14
			True
			False
			False

<	Less Than – tests if the first expression is less than the second	IF a<8
		IF name$<"JANET"
		IF a<14
			False
			True
			True

>	Greater Than – tests if the first expression is greater than the first	IF a>8
		IF name$>"JANET"
		IF a>14
			False
			False
			False

<>	Not Equal – tests if two expressions are different	IF a<>8
		IF name$<>"JANET"
		IF name$<>"JAMES"
		IF a<>14
			False
			True
			False
			True

<=	Less Than or Equal	IF a<=8
		IF name$<="JANET"
		IF a<=14
			True
			True
			True

>=	Greater Than or Equal	IF a>=8
		IF name$>="JANET"
		IF a>=14
			True
			False
			False

Boolean operators AND, OR and NOT

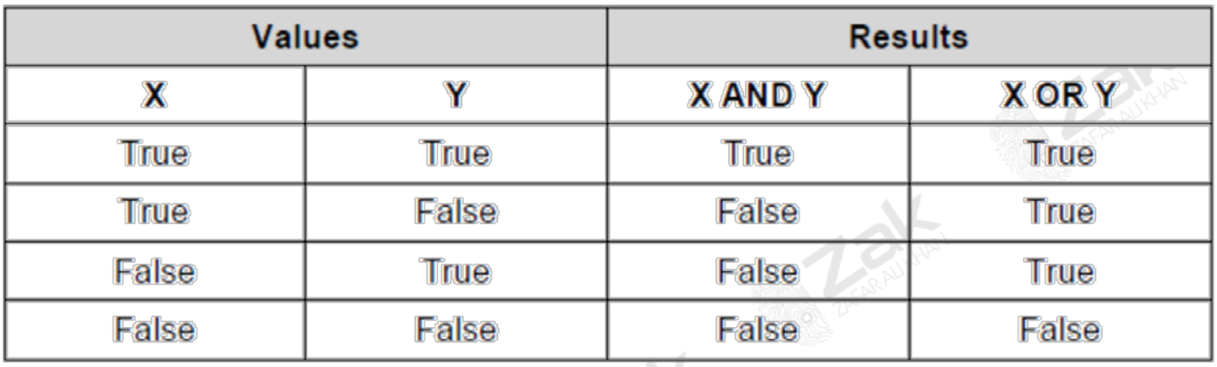
AND & OR

The AND & OR operators always return a Boolean result and are used in the format:

[Boolean] [Operator] [Boolean]

The following 'truth' table summarizes the result of the Boolean operations:

Values Results



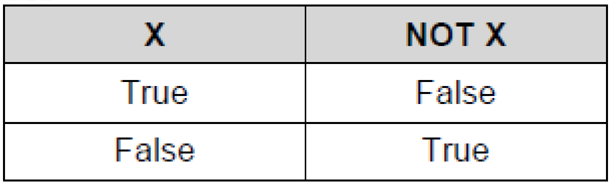
Values		Results	
X	Y	X AND Y	X OR Y
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

NOT

The NOT operator reverses the result of the Boolean expression and is used in the format:

NOT [Boolean]

The following truth table summarizes the NOT operation:



X	NOT X
True	False
False	True

Examples of Boolean 'logic'

Consider the following algorithm, which is used to monitor a printer and display its output via an LCD display in the front panel:

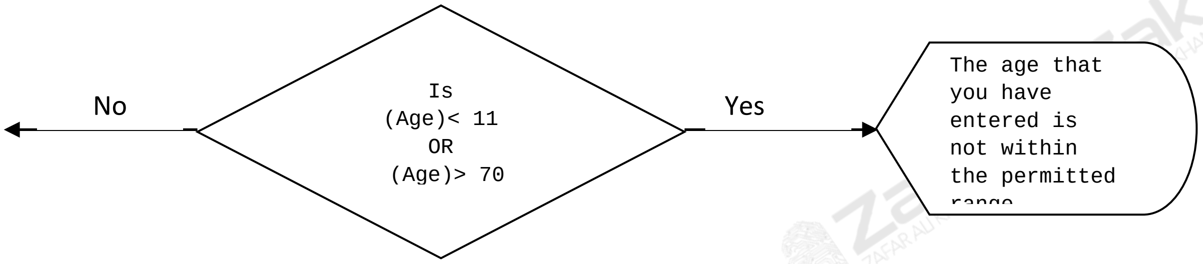
```
IF NOT(PaperTrayEmpty) AND (FilesWaiting > 0) THEN
      OUTPUT "PRINTING…"
ELSE
      OUTPUT "PLEASE ADD PAPER"
END IF
```

Topic: 2.1.1 Problem-solving and design

Validation

Validation checks ensure that data entered into the computer is sensible. Data is checked in accordance with a set of rules. The computer's software can validate data while it is being entered into the computer. The main purpose of data validation is to spot an error. This can be done quickly and easily as the process is automated.

Range check. A mathematics exam is out of 100. A simple validation rule that the computer can apply to any data that is input is that the mark must be between 0 and 100 inclusive. Consequently, a mark of 101 would be rejected by this check as being outside the acceptable range. Code can be added to check that a particular control has a value between an allowed maximum and minimum:

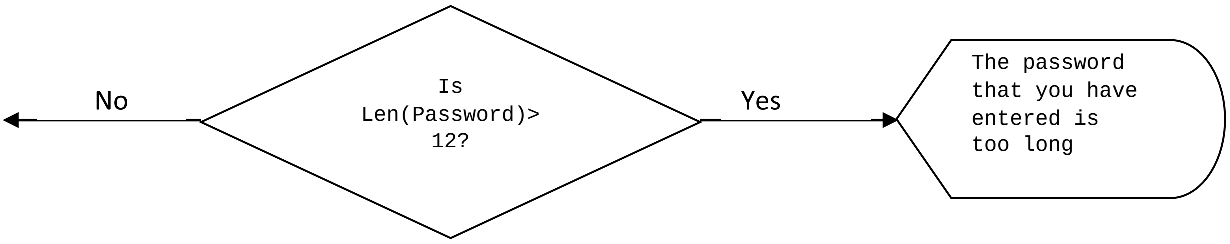


```
If Age < 11 OR Age > 70 Then
      Print (“The age that you have entered is not within the permitted range”)
End
```

Character check. A person's name will consist of letters of the alphabet and sometimes a hyphen or apostrophe. This rule can be applied to input of a person's name so that dav2d will immediately be rejected as unacceptable.

Format check. A particular application is set up to accept a national insurance number. Each person has a unique national insurance number, but they all have the same format of characters, 2 letters followed by 6 digits followed by a single letter. If the computer knows this rule then it knows what the format of a NI number is and would reject ABC12345Z because it is in the wrong format, it breaks the rule.

Length check. A NI number has 9 characters, if more or fewer than 9 characters are keyed in then the data cannot be accurate. Code can be added to check that a particular control has a value between an allowed maximum and minimum:383

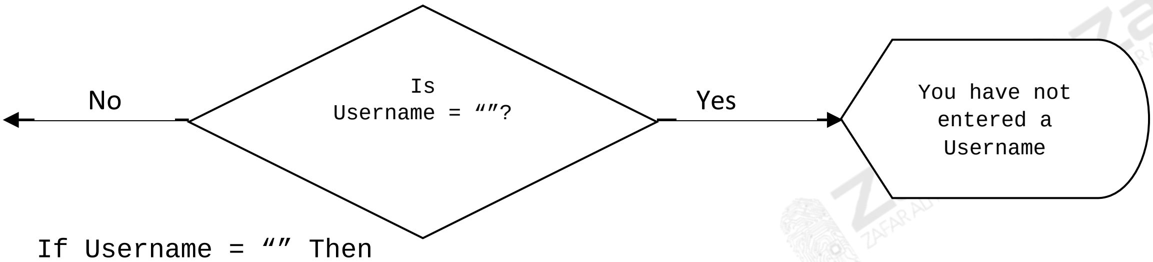


```
If Len(Password) > 12 Then
      Print (“The password that you have entered is too long”)
End
```

Existence check. A bar code is read at a supermarket check-out till. The code is sent to the main computer which will search for that code on the stock file. As the stock file contains details of all items held in stock, if it is not there then the item cannot exist, which it obviously does, therefore the code must have been wrongly read.

Presence check

Code can be added to check that a particular control has not been left empty or un-checked:



```
If Username = “” Then
      Print (“You have not entered a Username”)
End
```

Topic: 2.1.1 Problem-solving and design

Writing flowcharts to solve problems

The following five problems are also covered in section 3.2 where the algorithms are constructed using pseudocode. Candidates may choose to answer questions using either flowcharts or pseudocode but a working knowledge of both techniques is well advised.

2.1 Example 1

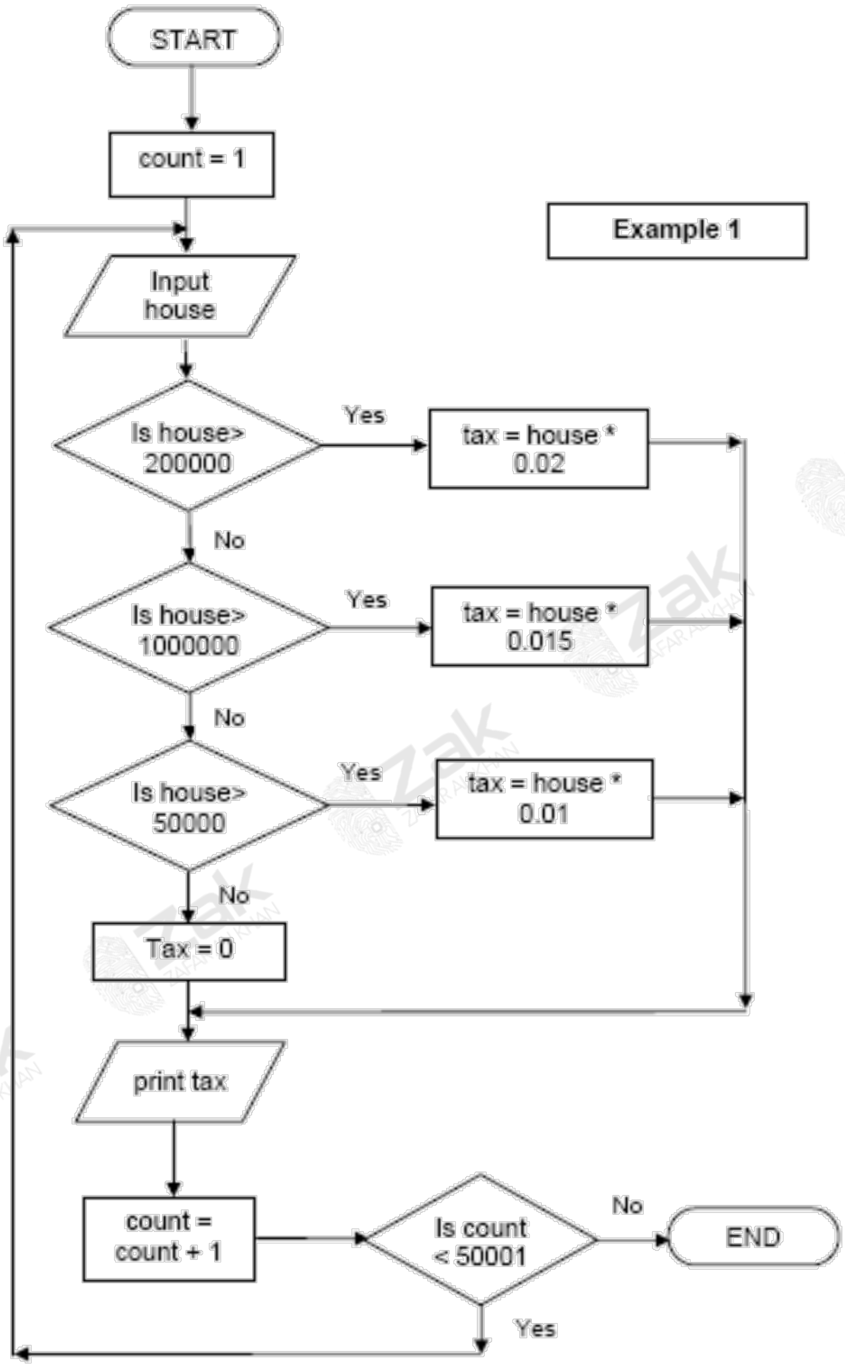
A town contains 5000 houses. Each house owner must pay tax based on the value of the house. Houses over $200 000 pay 2% of their value in tax, houses over $100 000 pay 1.5% of their value in tax and houses over $50 000 pay 1% of their value in tax. All others pay no tax. Write an algorithm to solve this problem in the form of a flowchart.

2.2 Example 2

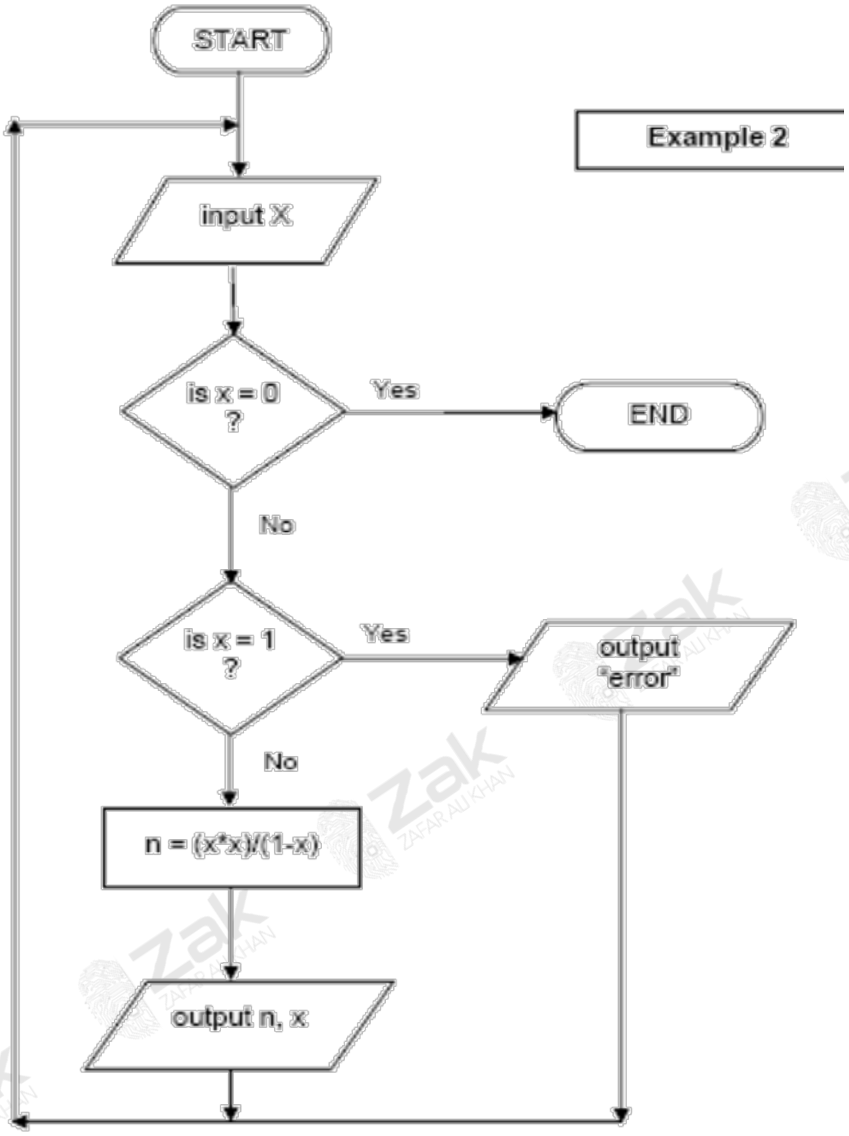
The following formula is used to calculate n: n = (x * x)/(1 – x). The value x = 0 is used to stop the algorithm. The calculation is repeated using values of x until the value x = 0 is input. There is also a need to check for error conditions. The values of n and x should be output. Write an algorithm to show this repeated calculation in the form of a flowchart.

Topic: 2.1.1 Problem-solving and design

Answers to the Examples:



Topic: 2.1.1 Problem-solving and design



Topic: 2.1.1 Problem-solving and design

Dry running of flowcharts

Dry running of flowcharts is basically a technique to:

- Determine the output for a known set of data to check it carries out the task correctly.
- Check the logic of the algorithm.
- Determine the function of the algorithm.
- When dry running a flowchart it is advisable to draw up a trace table showing how variables change their values at each stage in the algorithm.

  The advantages of doing this are:

  - If you make a mistake, it is easier to back track to where the error occurred rather than starting from the beginning again
  - There is less chance of an error being made
  - Encourages a more logical approach

The following three examples show all stages in the dry running for the given set of input data:

Example 1

This algorithm inputs 3 numbers, each number goes through successive division by 10 until its value is less than 1. An output is produced which contains the number input and a value generated by the flowchart processing.

Data to be used: X = 85, 3190, -40

Example 2

This algorithm inputs 5 values and outputs how many input numbers were negative and how many were positive.

Data to be used: N = 1, -5, 2, -8, -7

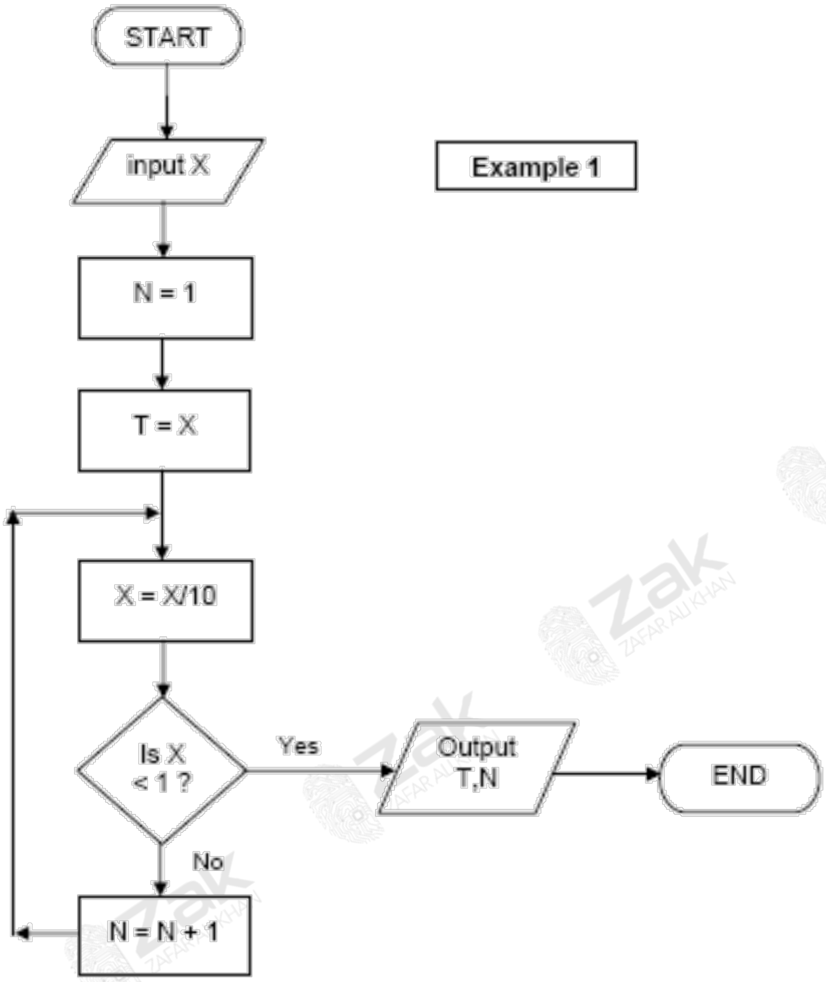
Example 3

This algorithm inputs the number of hours of sunshine recorded each day for a week (7 days). The output is the highest value for hours of sunshine and the average (mean) value for the numbers of hours of sunshine per day.

Data to be used: `hours = 9.0, 7.8, 1.2, 4.5, 10.0, 6.4, 3.1`

Topic: 2.1.1 Problem-solving and design

Answers to the Examples:

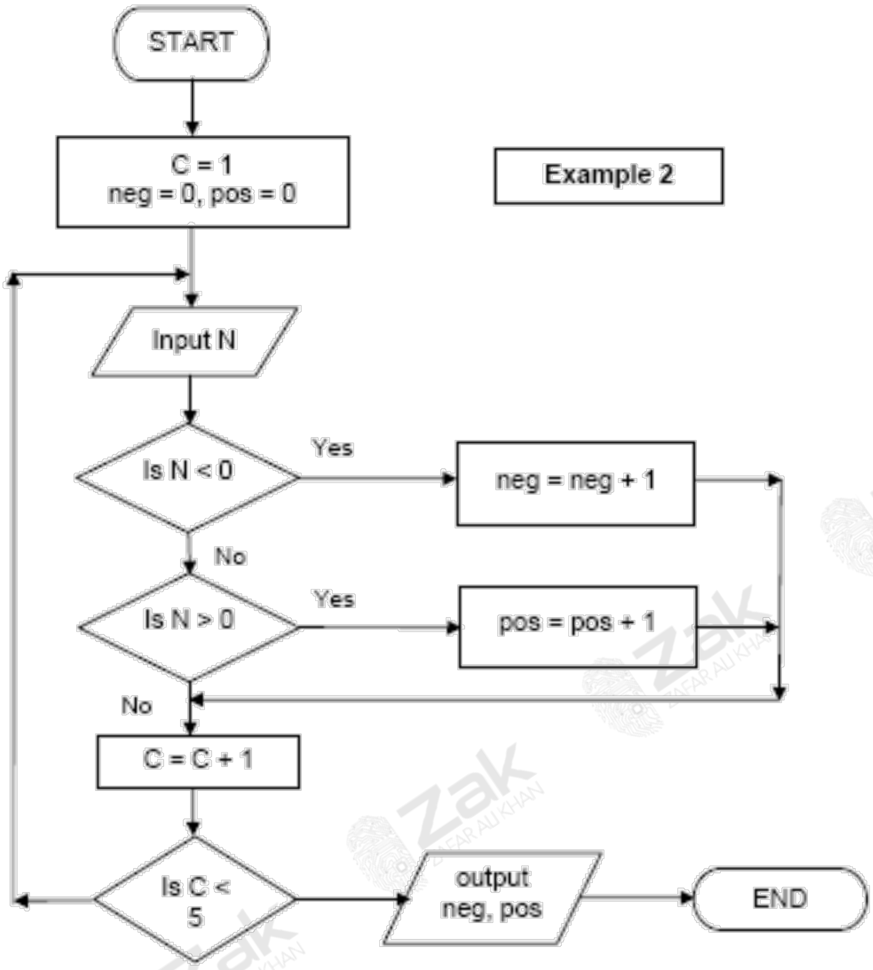


Trace Table

X	N	T	Output T, N
85 8.5 0.85	1 2	85	85, 2
3190 319 31.9 3.19 0.319	1 2 3 4	3190	3190, 4
-40 -4	1	-40	-40, 1

Topic: 2.1.1 Problem-solving and design

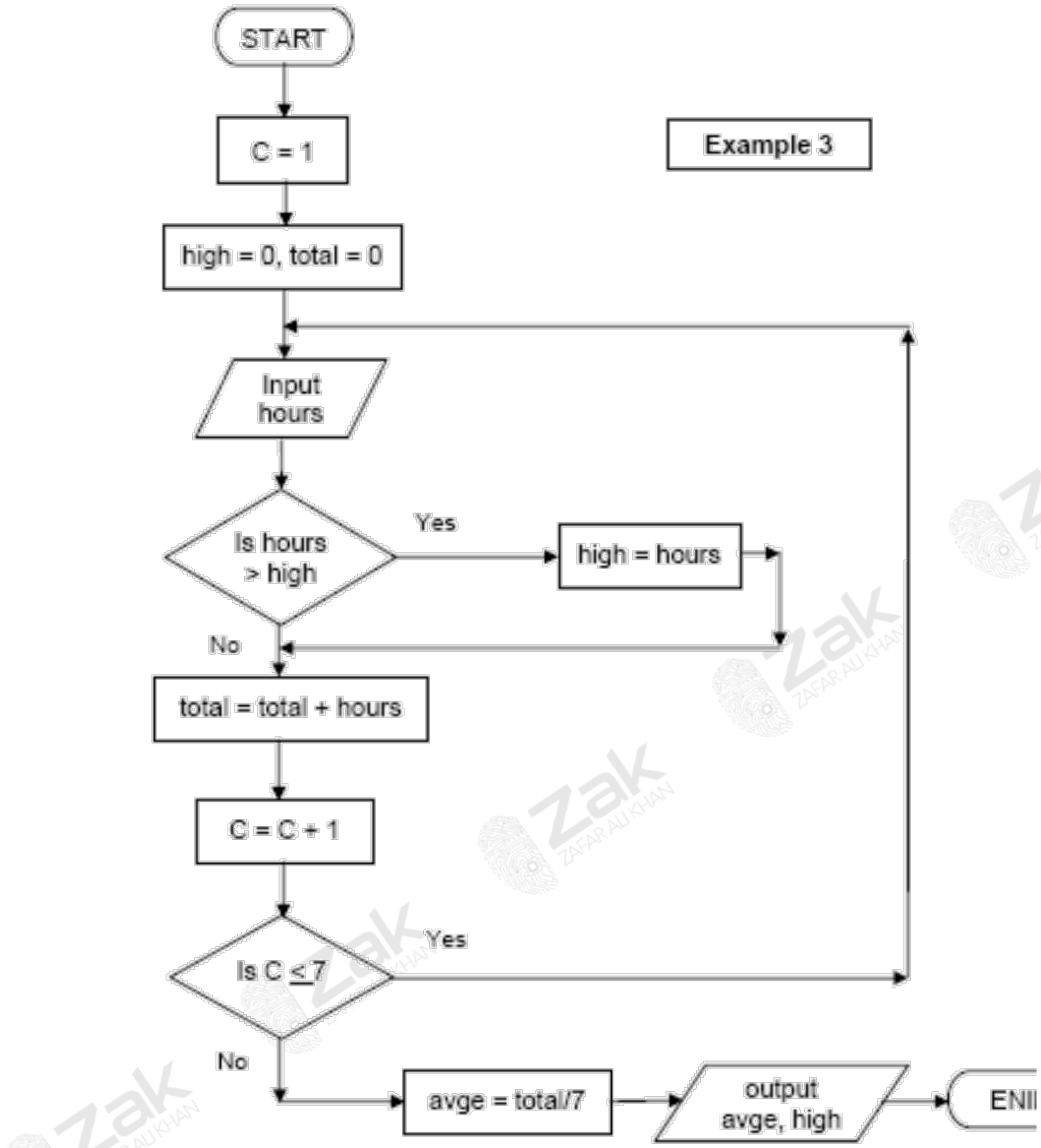
Example 2



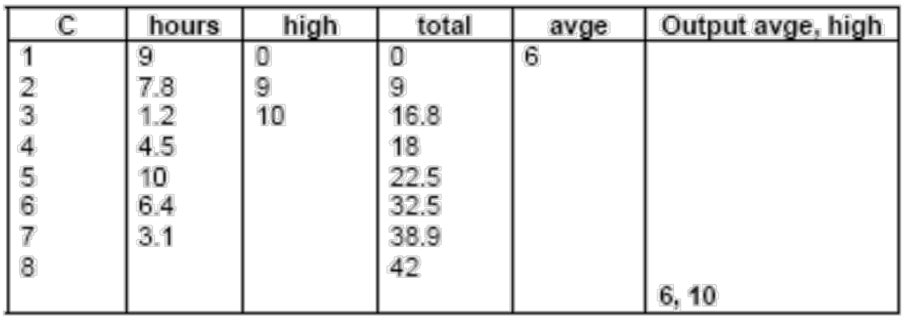
Trace Table

C	N	neg	pos	Output neg, pos
1	1	0	0	
2	-5	1	1	
3	2	2	2	
4	-8	3		
5	-7			
6				3, 2

Topic: 2.1.1 Problem-solving and design



Trace Table



C	hours	high	total	avge	Output avge, high
1	9	0	0	6	
2	7.8	9	9		
3	1.2	10	16.8		
4	4.5		18		
5	10		22.5		
6	6.4		32.5		
7	3.1		38.9		
8			42		
					6, 10

Topic: 2.1.1 Problem-solving and design

Questions 1 to 7 are problems which require an algorithm to be written in the form of a flowchart. Questions 8 to 10 require a trace table to be written and find the expected output for the given set of data. The answers to these questions can be found in booklet 6.

1. **Regis lives in Brazil and often travels to USA, Europe and Japan. He wants to be able to convert Brazilian Reais into US dollars, European euros and Japanese yen.**

   The conversion formula is: Currency value = number of Reais X conversion rate

   For example, if Regis is going to USA and wants to take 1000 Reais (and the exchange rate is 0.48) then he would input USA, 1000 and 0.48 and the output would be: 480 US dollars.
   Write an algorithm, using a flowchart, which inputs the country he is visiting, the exchange rate and the amount in Brazilian Reais he is taking. The output will be value in foreign currency and the name of the currency.

2. **As part of an experiment, a school measured the heights (in meters) of all its 500 students.**

   Write an algorithm, using a flowchart, which inputs the heights of all 500 students and outputs the height of the tallest person and the shortest person in the school.

3. **A geography class decide to measure daily temperatures and hours of sunshine per day over a 12 month period (365 days)**

   Write an algorithm, using a flowchart, which inputs the temperatures and hours of sunshine for all 365 days, and finally outputs the average (mean) temperature for the year and the average (mean) number of hours per day over the year.

4. **A small shop sells 280 different items. Each item is identified by a 3 – digit code. All items that start with a zero (0) are cards, all items that start with a one (1) are sweets, all items that start with a two (2) are stationery and all items that start with a three (3) are toys.**

   Write an algorithm, using a flowchart, which inputs the 3 – digit code for all 280 items and outputs the number of cards, sweets, stationery and toys.

5. A company are carrying out a survey by observing traffic at a road junction. Each time a car, bus or lorry passed by the road junction it was noted down.
   10 000 vehicles were counted during the survey.
   Write an algorithm, using an algorithm, which:
   inputs all 10000 responses outputs the number of cars, buses and lorries that passed by the junction during the survey outputs the number of vehicles that weren't cars, buses or lorries during the survey.

Topic: 2.1.1 Problem-solving and design

6. Speed cameras read the time a vehicle passes a point (A) on the road and then reads the time it passes a second point (B) on the same road (points A and B are 100 metres apart).
The speed of the vehicle is calculated using:

$$\text{speed} = \frac{100}{(\text{time at point B} - \text{time at point A})} \qquad \text{(metres/sec)}$$

The maximum allowed speed is 100 kilometres per hour. 500 vehicles were monitored using these cameras over a 1 hour period.

Write an algorithm, using a flowchart, which:

- inputs the start time and end time for the 500 vehicles that were monitored
- calculate the speed for each vehicle using the formula above
- outputs the speed for each vehicle and also a message if the speed exceeded 100 km/hour
- output the highest speed of all the 500 vehicles monitored

7. There are ten stations on a railway line:

1 ------ 2 ------ 3 ------ 4 ------ 5 ------ 6 ------ 7 ------ 8 ------ 9 ------ 10

The train travels in both directions (i.e. from 1 to 10 and then from 10 to 1). The fare between each station is $2.
A passenger inputs the number of the station at the start of his journey and the number of the destination station and the fare is calculated (e.g if a passenger gets on a station 3 and his destination is station 9 his fare will be $12). The calculation must take into account the direction of the train (e.g. a passenger getting on at station 7 and getting off at station 1 will also pay $12 and not a negative value!!).
A discount of 10% is given if 3 or more passengers are travelling together.

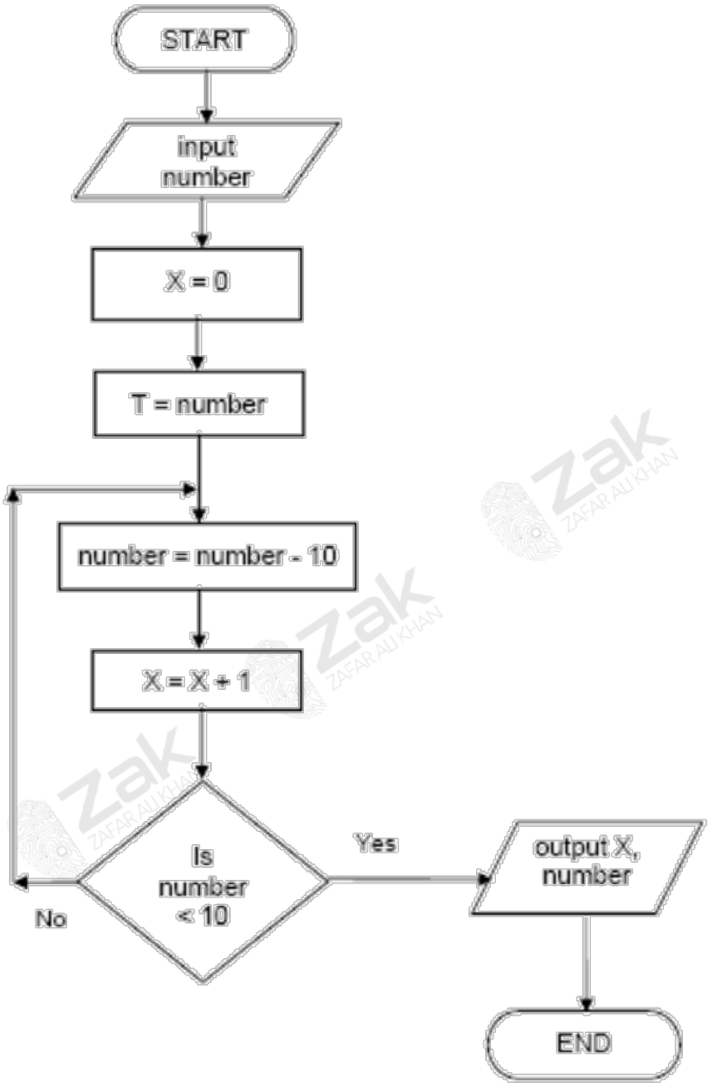
Write an algorithm, using a flowchart, which:

- inputs the number of passengers travelling
- inputs the station number of the starting point and the station number of the destination
- calculates the total fare taking into account the direction of travel
- calculates any discount due
- outputs the cost of the tickets and prints the tickets

Topic: 2.1.1 Problem-solving and design

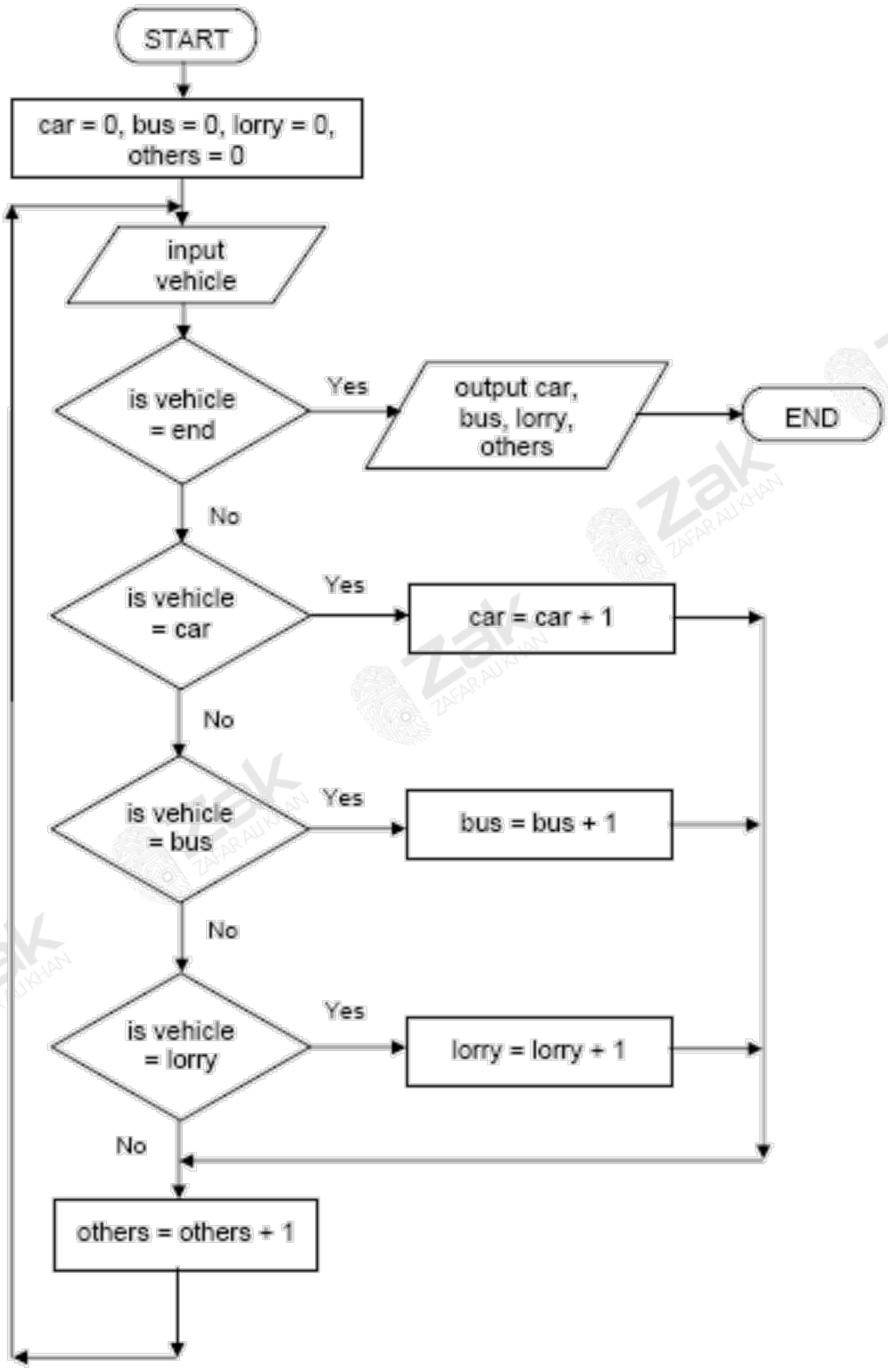
8. **Draw the trace table and determine the output from the following flowchart using the following data:**

number = 45, -2, 20.5



9. **Draw the trace table and determine the output from the following flowchart using the following data (NOTE: input of the word "end" stops the program and outputs results of the survey):**

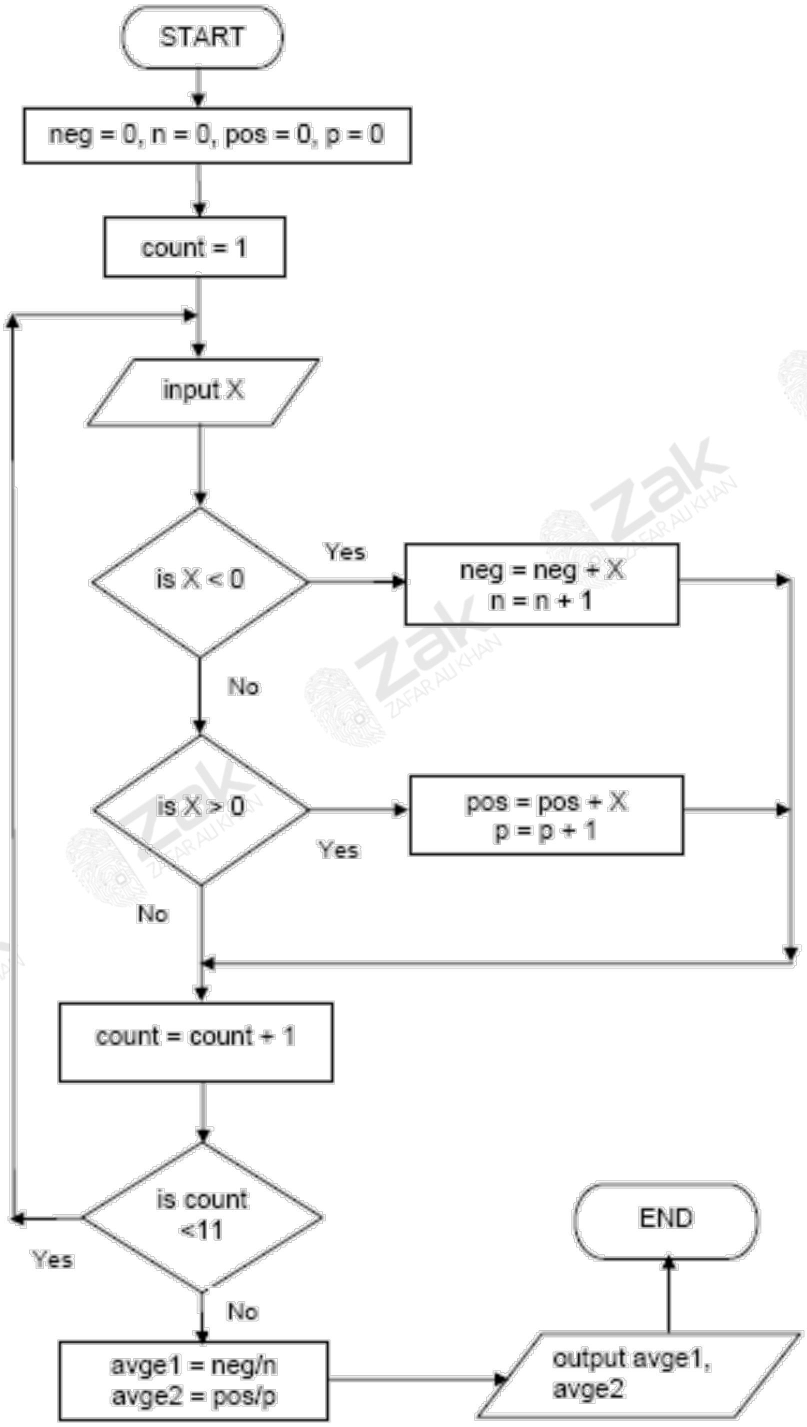
vehicle = car, car, lorry, bus, van, van, car, car, bus, car, end



Topic: 2.1.1 Problem-solving and design

10. Draw the trace table and determine the output from the following flowchart using the following data:

X = 5, -3, 0, -3, 7, 0, 6, -11, -7, 12



Topic: 2.1.1 Problem-solving and design

Questions:

Modular Design/Top-down design:

May/June 2006

5. (a) Programs can be designed in modular form.
Discuss the advantages and disadvantages of designing programs in modular form. **[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.

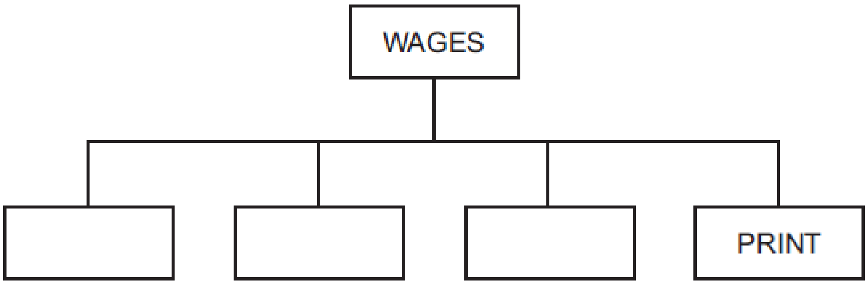
8. (a) As part of the process of designing a site, the company will use diagrams in order to make understanding easier.
Describe two types of diagram that may be used by the company. **[4]**

May/June 2011. P23

1. (a) Draw a suitable layout for the screen. **[5]**

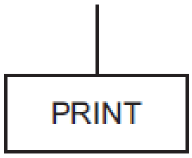
3. Kris has written a program that will work out the wages for her staff. The main steps for each employee are: to work out the hours worked, work out the total earnings, work out tax and finally print out how much will be taken home.

(a) Complete the structure diagram to show the modules that will be needed.



[3]

(b) The printout will be different for those staff who receive cash and those who have their earnings paid directly to a bank account. Add the next level to the print module.

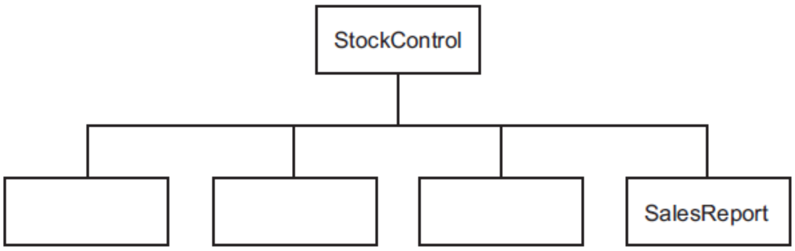


Oct/NOV 2011 P23

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.

(a) State why modularisation is a good idea. **[1]**

(b) As the first step in his design he splits the solution into the following main areas: Initialisation, StockOrdering, Sales, SalesReport.
Complete the following structure diagram.



[1]

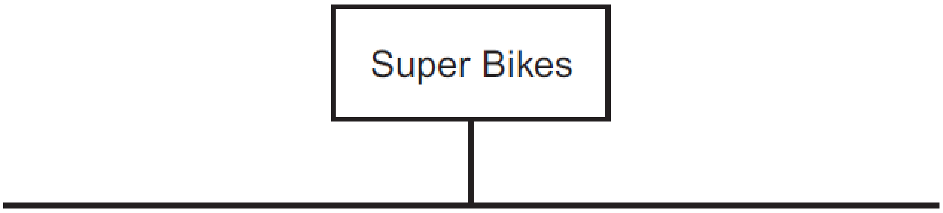
(c) SalesReport is made up of two modules, MonthlySalesReport and AnnualSalesReport.
Add them to the structure diagram in (b). **[2]**

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

(a) Complete the structure diagram showing these tasks.



[2]

(b) The collection of information about repairs has three subtasks:

- input the repair needed
- input the parts list
- input the cost of the repair

Add these to the structure diagram in part (a). **[1]**

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