Syllabus Content:

2.1.2 Structure chart

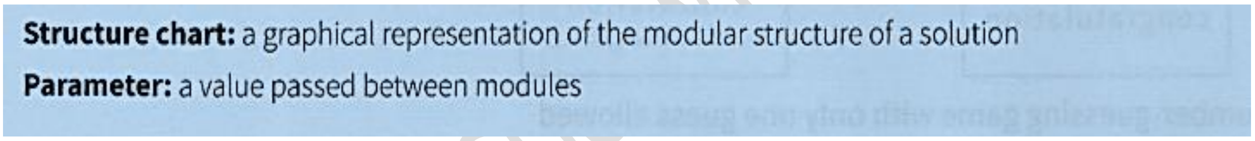
- use a structure chart to express the parameters passed between the various modules/ procedures/functions which are part of the algorithm design
- describe the purpose of a structure chart
- construct a structure chart for a given problem
- derive equivalent pseudocode from a structure chart

Structure charts:

An alternative approach to modular design is to choose the sub-tasks and then construct a structure chart to show the interrelations between the modules. Each box of the structure chart represents a module. Each level is a refinement of the level above.

A structure chart also shows the interface between modules, the variables. These variables are referred to as 'parameters'.

A **parameter** supplying a value to a lower-level module is shown as a downwards pointing arrow. A parameter supplying a new value to the module at the next higher level is shown as an upward pointing arrow.

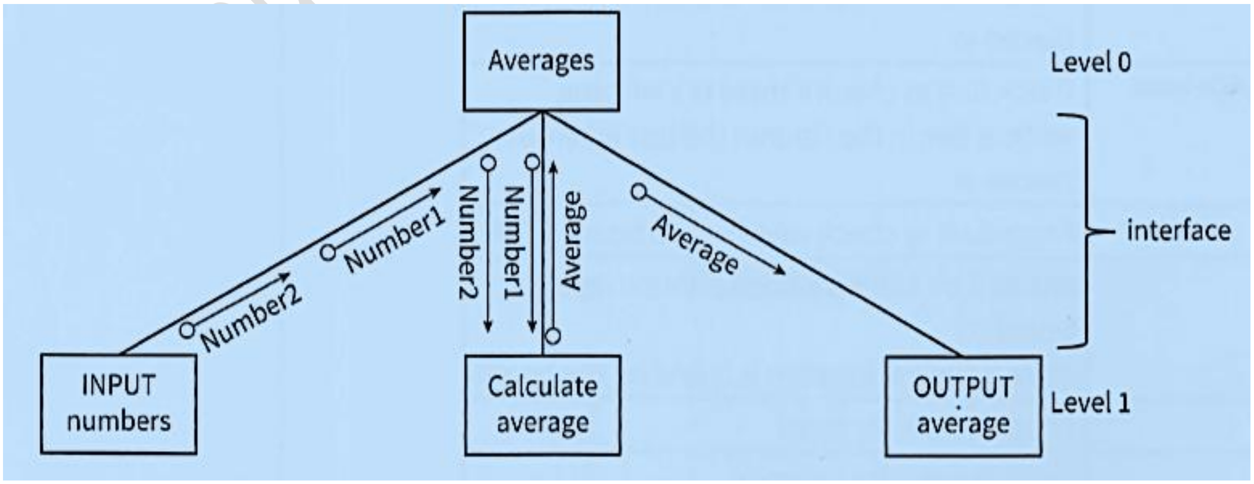


Structure chart: a graphical representation of the modular structure of a solution

Parameter: a value passed between modules

Figure below shows a structure chart for a module that calculates the average of two numbers. The top-level box is the name of the module, which is refined into the three subtasks of Level 1. The input numbers (parameters **Number1** and **Number2**) are passed into the **'Calculate Average'** sub-task and then the **Average parameter** is passed into the **'OUTPUT** Average' sub-task. The arrows show how the parameters are passed between the modules.

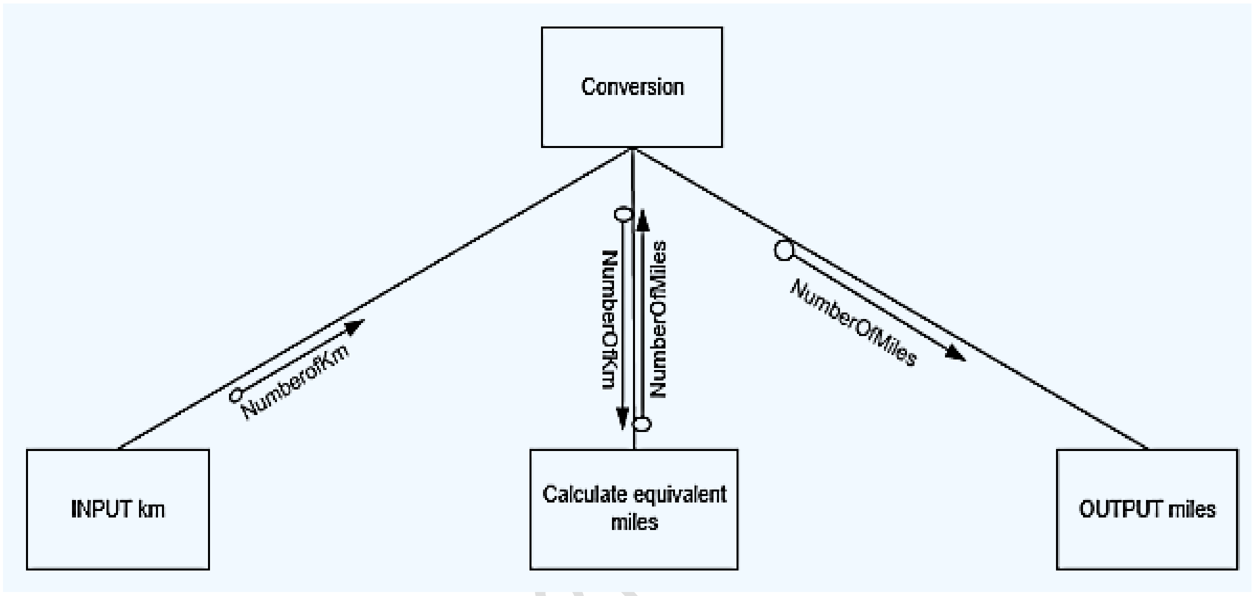
This parameter passing is known as the **'interface'**.



TASK 12.03

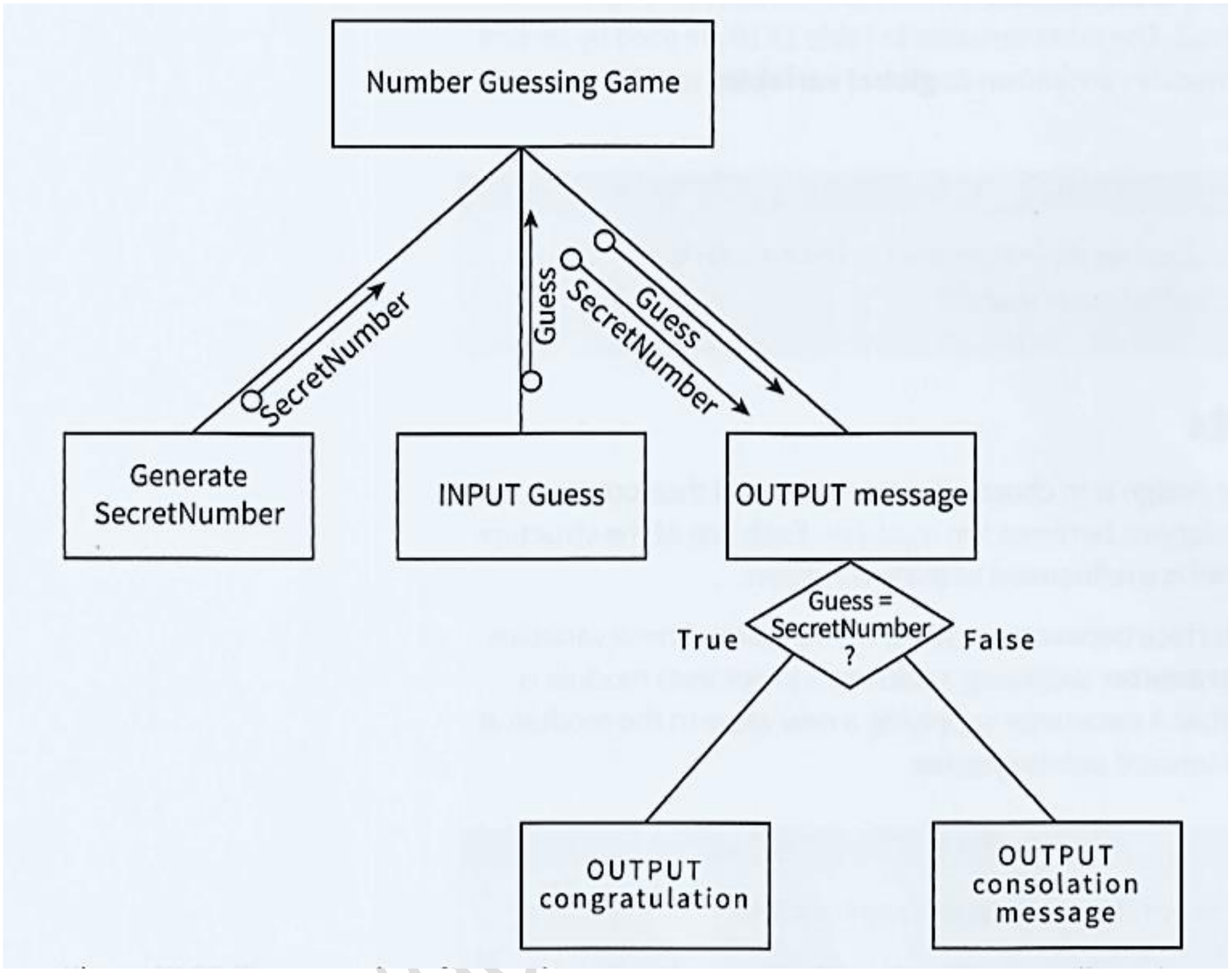
Draw a structure chart for the following module: Input a number of km, output the equivalent number of miles.

Answer:



Structure charts can also show control information: selection and repetition.

The simple number-guessing game could be modularised and presented as a structure chart, as shown in Figure below



Structure chart for number-guessing game with only one guess allowed

Past paper questions on structure charts:

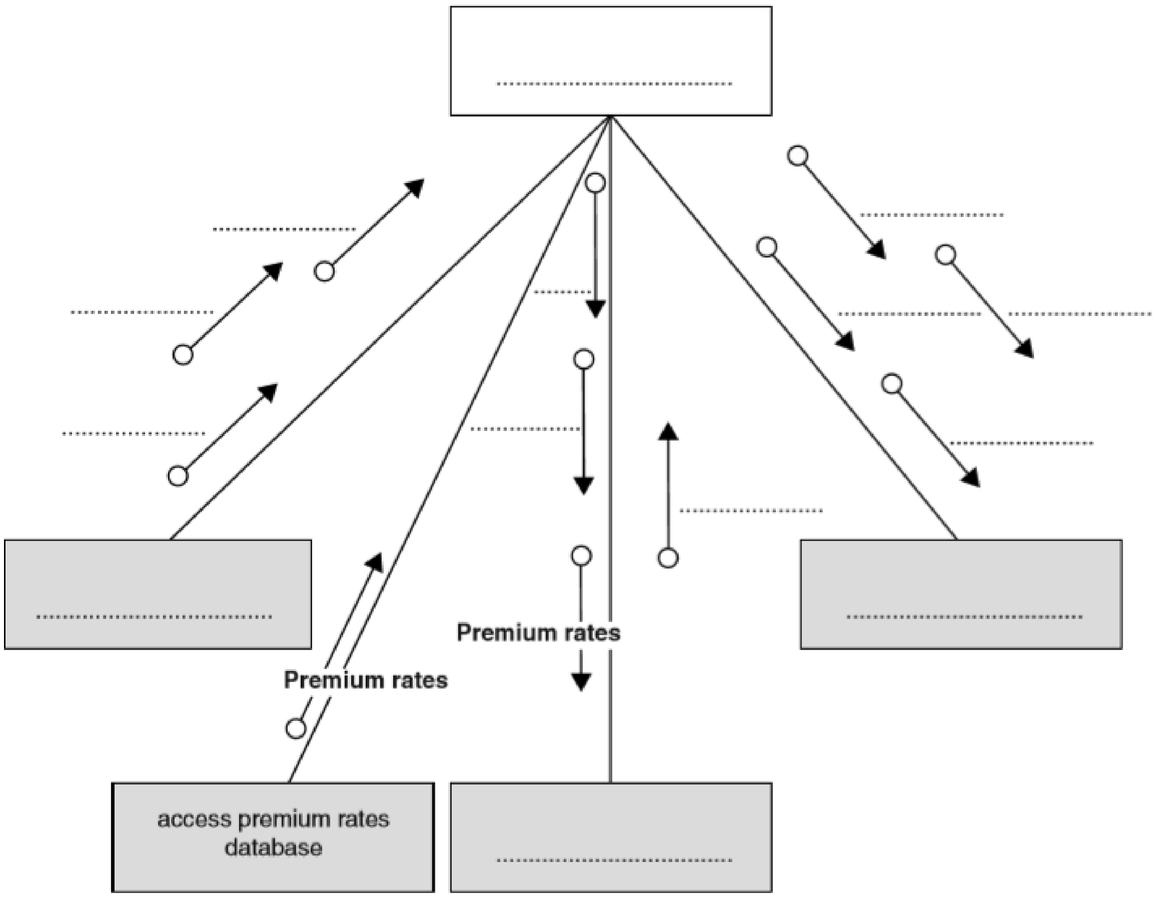
MJ 2015/ 21
Q 3:
When the guarantee on a computer runs out, the owner can take out insurance to cover breakdown and repairs.

The price of the insurance is calculated from:

- the model of the computer
- the age of the computer
- the current insurance rates

Following an enquiry to the insurance company, the customer receives a quotation letter with the price of the insurance.
A program is to be produced.

The structure chart below shows the modular design for this process:



(a) Using the letters **A** to **D**, add the labelling to the chart boxes on the opposite page.

Modules	
A	Send quotation letter
B	Calculate price
C	Produce insurance quotation
D	Input computer details

[2]

(b) Using the letters E to J, complete the labelling on the chart opposite.

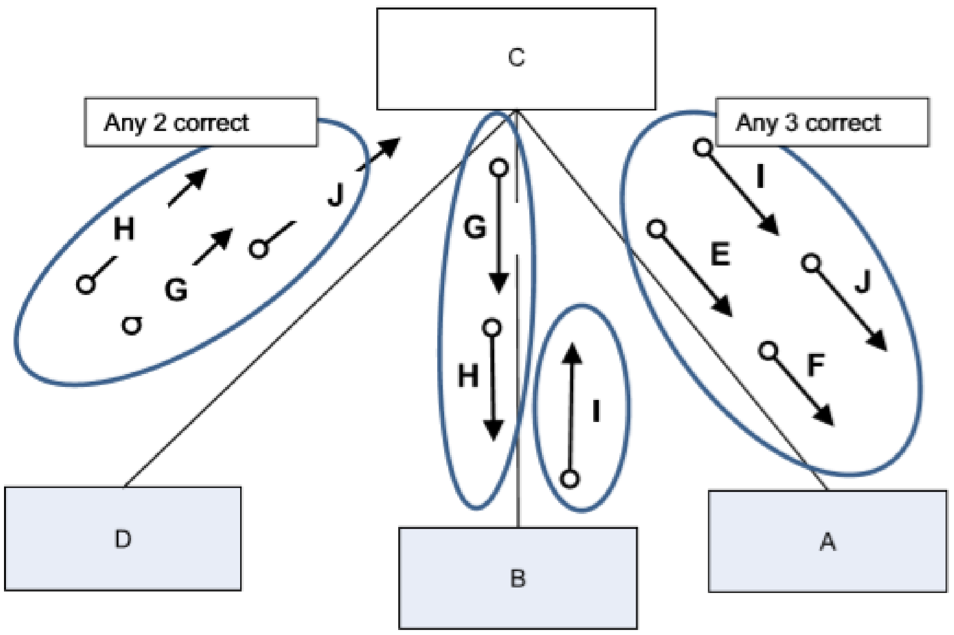
Some of these letters will be used more than once.

Data items	
E	`CustomerName`
F	`CustomerEmail`
G	`Model`
H	`Age`
I	`PolicyCharge`
J	`PolicyNumber`

[4]

Answer:

(b)



Data items	
E	CustomerName
F	CustomerEmail
G	Model
H	Age
I	PolicyCharge
J	PolicyNumber

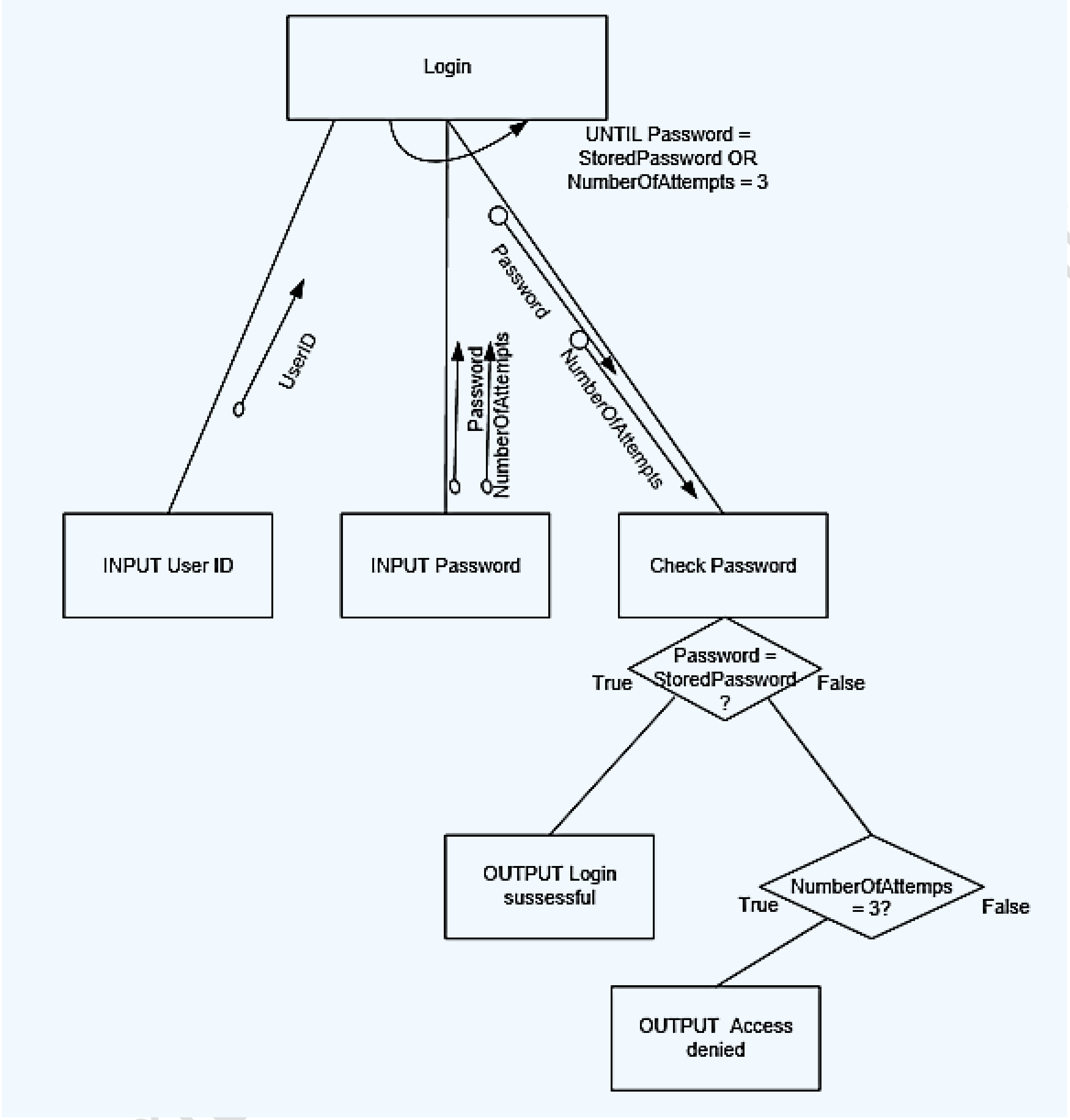
[4]

TASK 12.05

Draw a structure chart for the following problem: A user attempts to log on with a user ID. User IDs and passwords are stored in two 1D arrays (lists). The algorithm searches the list of user IDs and looks up the password in the password list. The user is given three chances to input the correct password. if the correct password is entered, a suitable message is output. If the third attempt is incorrect, a warning message is output.

Answer:

Task 12.05



Exam-style Questions
1 A random number generator is to be tested to see whether all numbers within the range 1 to 20 are generated equally frequently.

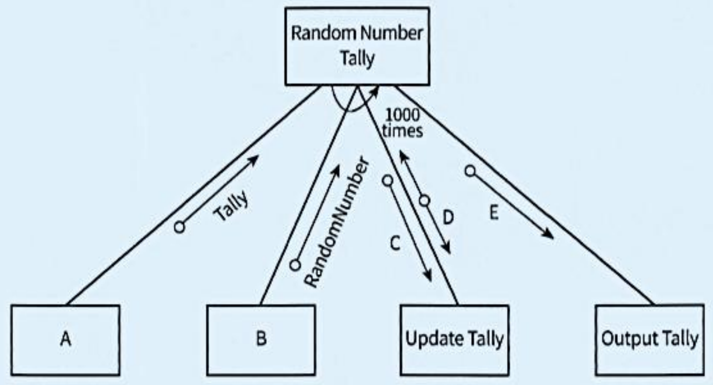
The structured English version of the algorithm is:

- Initialise a tally for the numbers 1 to 20
- Repeatedly generate numbers in range 1 to 20
- For each number generated, increment the relevant count
- Calculate how often each number should be generated (expected frequency)
- Output expected frequency
- Output the list of numbers as a table with actual frequency

The identifiers required are:

Identifier	Explanation
`Tally[1..20]`	1D array to store the count of how many times each number has been generated
`RandomNumber`	The random number generated
`NumberOfTests`	The number of times a random number is to be generated (1000 in this example)
`ExpectedFrequency`	The number of times any one number would be generated if all numbers are generated equally frequently (1000/20 in this example)

a Complete the structure chart below by naming the labels A to E: [5]



b Produce pseudocode from the structure chart. [12]

Answer:

Exam-style questions in Chapter

1 a A: Initialise Tally
B: Generate random number
C: RandomNumber
D: Tally
E: Tally

(b) Pseudo code of Random Number Tally

```
DECLARE Tally : ARRAY[1..20] OF INTEGER
CALL InitialiseTally(Tally)
    FOR Count ← 1 TO NumberOfTests
        RandomNumber ← GenerateRandomNumber(20)
        CALL UpdateTally(RandomNumber, Tally)
    ENDFOR
CALL OutputTally(Tally)
```

Reference:

- Cambridge International AS & A level Computer Science Course book by Sylvia Langfield and Dave Duddell