

Solutions

Each answer shows the steps, then the **result**.

2.1

- (a) a **terminator** – a rounded box
- (b) a **process** – a rectangle
- (c) an **input/output** symbol – a parallelogram
- (d) a **decision** – a diamond

2.2

- **two** exits
- each must be **labelled**, Yes and No (or True and False)

2.3

- for a **WHILE** loop the diamond is drawn **above** the body, so the condition is tested **before** the body runs
- for a **REPEAT** loop the diamond is drawn **below** the body, so the condition is tested **after** the body runs
- so a **WHILE** body may run **zero** times, while a **REPEAT** body always runs **at least once**

2.4

- only a **decision** box may have two exits; a process box does exactly one thing and has one arrow in and one arrow out

3.1

- **Start** and **Stop** in rounded boxes
- **INPUT** Temperature in a parallelogram
- a diamond reading Temperature > 30?
- **OUTPUT** "Hot" on the Yes branch and **OUTPUT** "Not hot" on the No branch, both in parallelograms
- both exits **labelled**, and the two branches rejoining before **Stop**

3.2

- **Start**, **Stop**, and **Total <- 0** in a process box
- **Count <- 1** initialised in a process box before the loop
- **INPUT** Number in a parallelogram and **Total <- Total + Number** in a process box
- **Count <- Count + 1** and a diamond testing **Count <= 5?**, whose **Yes** exit loops **back** to **INPUT** Number
- **OUTPUT** Total in a parallelogram on the No exit, before **Stop**
- a **FOR** loop counts a fixed number of times, so drawn as a flowchart it becomes an initialised counter, an increment and a test – the three parts the **FOR** line hides

3.3

- the declarations, and **INPUT** *n*
- *total* $\leftarrow$ 0 and *count* $\leftarrow$ 0
- a **post-condition** loop, because the diamond is drawn **below** the body
- the loop body: **INPUT** *value*, *total* $\leftarrow$ *total* + *value*, *count* $\leftarrow$ *count* + 1
- **UNTIL** *count* $\geq$ *n*, then *average* $\leftarrow$ *total* / *n* and **OUTPUT** *average*

```

DECLARE n, count, value, total : INTEGER
DECLARE average : REAL
INPUT n
total  $\leftarrow$  0
count  $\leftarrow$  0
REPEAT
    INPUT value
    total  $\leftarrow$  total + value
    count  $\leftarrow$  count + 1
UNTIL count  $\geq$  n
average  $\leftarrow$  total / n
OUTPUT average

```

- the diamond reads *count* < *n*? and its **Yes** loops back, so the **UNTIL** condition is the **negation**, *count* $\geq$ *n*
- a **WHILE** version scores nothing for the loop mark: the chart tests **after** the body, so the body must always run once

3.4

- a condition must be drawn in a **diamond**, not a rectangle: a rectangle is a process and may not have two exits
- the two arrows leaving a decision must be **labelled Yes** and **No**, or the reader cannot tell which path is which
- every path must reach **Stop**: a box with no arrow leaving it leaves the algorithm unfinished

3.5

- use **REPEAT** ... **UNTIL** when the body must run **at least once** whatever happens, because the condition cannot be tested until the body has run
- the value the condition depends on does not exist before the first pass
- example: asking the user for a password, or for a value in a valid range – you must ask once before you can check what was entered; a **WHILE** would have to test a variable that has not been given a value yet