Solutions

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

2.1

- set `Number` to 3 before the loop
- `WHILE Number <= 15 … ENDWHILE`
- output, then add 3 inside the loop

```
Number ← 3
WHILE Number <= 15
    OUTPUT Number
    Number ← Number + 3
ENDWHILE
```

2.2

- `REPEAT … UNTIL` structure
- `INPUT Age` inside the loop
- the condition `Age >= 0 AND Age <= 120`

```
REPEAT
    INPUT Age
UNTIL Age >= 0 AND Age <= 120
```

2.3

- input the first height and store it as the smallest
- a loop that runs for the other 24 heights
- compare each height with the smallest, and replace it if smaller
- output the smallest after the loop

```
DECLARE Count : INTEGER
DECLARE Height, Smallest : REAL
INPUT Height
Smallest ← Height
FOR Count ← 2 TO 25
    INPUT Height
    IF Height < Smallest THEN
        Smallest ← Height
    ENDIF
NEXT Count
OUTPUT Smallest
```

2.4

- a **pre-condition** loop
- the condition is tested before any word is processed, so the loop can run zero times when `"END"` is typed first

2.5

Outer	Inner	Total	OUTPUT
		0	
1	1	1	
2	1	2	
	2	4	
3	1	5	
	2	7	
	3	10	
			10

- `Outer` and `Inner` columns correct; `Total` column correct; output **10**

3.1

- declarations; a count-controlled loop from 1 to 30; input inside the loop
- count the days with rainfall 0; running total
- keep the largest rainfall and its day number, replacing only when greater
- output all three after the loop

```
DECLARE Day, DryDays, WettestDay : INTEGER
DECLARE Rain, Total, MostRain : REAL
DryDays ← 0
Total ← 0
MostRain ← -1
WettestDay ← 0
FOR Day ← 1 TO 30
    INPUT Rain
    IF Rain = 0 THEN
        DryDays ← DryDays + 1
    ENDIF
    Total ← Total + Rain
    IF Rain > MostRain THEN
        MostRain ← Rain
        WettestDay ← Day
    ENDIF
NEXT Day
OUTPUT DryDays
OUTPUT Total
OUTPUT WettestDay
```

- `MostRain` starts at -1, below any real rainfall, so day 1 is always stored first; `>` rather than `>=` keeps the earlier day

3.2

(a) a loop that ends when the choice is 4; input the choice inside the loop

- `CASE OF … ENDCASE`; choices 1 and 2; choice 3; choices 4 and `OTHERWISE`

```
DECLARE Choice : INTEGER
DECLARE Total, Number : REAL
Total ← 1
REPEAT
    INPUT Choice
    IF Choice = 1 OR Choice = 2 THEN
        INPUT Number
    ENDIF
    CASE OF Choice
        1 : Total ← Total + Number
        2 : Total ← Total * Number
        3 : OUTPUT Total
        4 : OUTPUT "Stopped"
        OTHERWISE : OUTPUT "Invalid"
    ENDCASE
UNTIL Choice = 4
```

- *(inputting `Number` inside the clauses for 1 and 2 of the `CASE` structure is also correct)*

(b) a **post-condition** loop

- the choice must be input at least once before it can be tested

3.3

(a)

Value	Count	Total	OUTPUT
	0	0	
6	1	6	
-2			
9	2	15	
0			
999			7.5

- `Value` column; `Count` column; `Total` column; output **7.5**

(b) it outputs the mean (average) of the **positive** values input

- it stops at the rogue value 999, and outputs `"No data"` if there were no positive values

(c) the test comes at the end of the loop, so 999 is processed before the loop stops

- 999 is positive, so it is added to `Total` and counted, and the output is 999 instead of `"No data"`
- correct it by changing the condition to `Value > 0 AND Value <> 999`