

## Solutions

---

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

### 2.1

- (a) INTEGER
- (b) REAL
- (c) BOOLEAN

### 2.2

- it groups **several fields of different data types** under **one identifier**, so values that belong together are stored as one unit

### 2.3

- TYPE ... ENDTYPE with three fields of the right types

```
TYPE TBook
  DECLARE Title      : STRING
  DECLARE Pages      : INTEGER
  DECLARE InLibrary  : BOOLEAN
ENDTYPE
```

### 2.4

- Book1.Title ← "Maths"

### 3.1

- A → **CHAR** (or **STRING** — one character from **LEFT**)
- B → **REAL** (a price/decimal)
- C → **INTEGER** (whole-number add)
- D → **BOOLEAN** (a comparison)

### 3.2

- (i) TYPE ... ENDTYPE with the four field types

```
TYPE Component
  DECLARE Item_Num  : INTEGER
  DECLARE Reject    : BOOLEAN
  DECLARE Stage     : CHAR
  DECLARE Limit_1   : REAL
ENDTYPE
```

- (ii) DECLARE Item : ARRAY[1:2000] OF Component

### 3.3

- any two: one identifier for all the data; easy to process with a **loop** and an index; all fields for one item stay together; easy to pass to a procedure

### 3.4

(a) HireRef — STRING

- HoursHired — INTEGER; CostPerHour — REAL
- Returned — BOOLEAN

(b)

```
TYPE HireRecord
  DECLARE HireRef : STRING
  DECLARE BicycleNumber : INTEGER
  DECLARE HireDate : DATE
  DECLARE HoursHired : INTEGER
  DECLARE CostPerHour : REAL
  DECLARE Returned : BOOLEAN
ENDTYPE
```

- TYPE/ENDTYPE; all six fields; correct types; DATE for the date; correct pseudocode syntax

(c) DECLARE Hire : ARRAY[1:500] OF HireRecord

(d) Hire[1].CostPerHour  $\leftarrow$  4.50

(e) any three: the fields belonging to one hire are kept **together**, so a record can be passed to a procedure as a single item

- there is no risk of the arrays drifting **out of step** with one another
- adding a new field means changing one type definition instead of creating another array

(f)  $3 \times 4.50 = \mathbf{13.50}$

- the result is REAL, because multiplying an INTEGER by a REAL gives a REAL