Assignment, input and output

Ref Pseudocode

The assignment arrow

The **assignment** 赋值 operator is ←, a left-pointing arrow. It means *take the value on the right and put it in the box on the left.*

```
<identifier> ← <value>
```

```
DECLARE Counter : INTEGER
Counter ← 0
Counter ← Counter + 1
Counter ← Counter + 1
OUTPUT Counter
```

`Counter ← Counter + 1` looks impossible as mathematics and is ordinary as an instruction: work out the right-hand side with the value the box holds now, then store the result back.

- If you cannot type ←, write `<-`. Both are accepted, here and in the exam.
- The left side must be a **single identifier** or one array element. Never a calculation.

Common mistakes

- `Counter = 0`. In pseudocode `=` is a **comparison**, not an assignment — and confusing the two is the single commonest syntax error in this paper.
- `Counter + 1 ← Counter`. The arrow points at the thing being changed.

INPUT and OUTPUT

`INPUT` reads a value from the user into a variable. `OUTPUT` displays one or more values.

```
DECLARE Name : STRING
DECLARE Age : INTEGER
INPUT Name
INPUT Age
OUTPUT "Hello ", Name
OUTPUT "Next year you will be ", Age + 1
```

- `INPUT` takes exactly **one** variable, and that variable must already be declared.

- `OUTPUT` takes any number of items separated by **commas**; they are printed one after another with nothing between them.
- A literal 字面量 string in an `OUTPUT` goes in double quotes; a variable does not.

Prompting the user

An `INPUT` on its own shows nothing on screen, so a well-written algorithm prints a prompt 提示 first:

```
DECLARE Radius : REAL
OUTPUT "Enter the radius: "
INPUT Radius
OUTPUT "The diameter is ", Radius * 2
```

Common mistakes

- `INPUT "Enter a number", X` — the prompt is a separate `OUTPUT` statement.
- `OUTPUT "Total: " + Total`. 0478 joins output items with a comma; `+` is arithmetic and `&` belongs to A-Level 9618 (topic 11).
- Forgetting the quotes: `OUTPUT Hello` looks for a *variable* called `Hello`.