

Arithmetic and logic

Ref Pseudocode

Arithmetic operators

Five arithmetic operators, in the order a calculator uses them.

Operator	Meaning	A is 7, B is 2
+	add	A + B is 9
-	subtract	A - B is 5
*	multiply	A * B is 14
/	divide	A / B is 3.5
^	raise to a power	A ^ B is 49

```
DECLARE A, B : INTEGER
A ← 7
B ← 2
OUTPUT "Sum: ", A + B
OUTPUT "Product: ", A * B
OUTPUT "Quotient: ", A / B
OUTPUT "Power: ", A ^ B
```

Brackets come first, then ^, then * and /, then + and - — the usual precedence . Use brackets whenever the reader would have to stop and think.

```
DECLARE Total : REAL
Total ← (12 + 18) / 2
OUTPUT "Mean: ", Total
```

Common mistakes

- Expecting / to give a whole number. 7 / 2 is 3.5, not 3; for a whole number use DIV.
- Writing A x B or A ÷ B. Use * and /.

DIV and MOD

Integer division answers two different questions, and 0478 writes each of them as a **function** , with the two values in brackets.

- DIV(a, b) — how many whole times **b** goes into **a**, the **quotient** .
- MOD(a, b) — what is left over, the **remainder** .

```
OUTPUT DIV(10, 3)
OUTPUT MOD(10, 3)
OUTPUT DIV(17, 5)
OUTPUT MOD(17, 5)
```

Both take integers, and both return an integer.

What they are for

MOD is the standard test for *divides exactly*, which makes it the way to ask whether a number is even :

```
DECLARE Number : INTEGER
Number ← 12
IF MOD(Number, 2) = 0
    THEN
        OUTPUT Number, " is even"
    ELSE
        OUTPUT Number, " is odd"
ENDIF
```

DIV converts a total into whole units — seconds into minutes, pence into pounds:

```
DECLARE Seconds : INTEGER
Seconds ← 200
OUTPUT DIV(Seconds, 60), " minutes and ", MOD(Seconds, 60), " seconds"
```

Common mistakes

- 10 MOD 3. That is the A-Level 9618 form; **0478 uses the function form** MOD(10, 3).
- Mixing the two up. DIV gives the whole part; MOD gives what does not fit.

Relational and logic operators

A **relational operator** compares two values and produces TRUE or FALSE.

Operator	Means
=	is equal to
<>	is not equal to
>	is greater than
<	is less than
>=	is greater than or equal to
<=	is less than or equal to

```
DECLARE Mark : INTEGER
Mark ← 55
OUTPUT "Pass: ", Mark >= 50
OUTPUT "Perfect: ", Mark = 100
OUTPUT "Not zero: ", Mark <> 0
```

The three **logic operators** join conditions:

- AND — both sides must be TRUE.
- OR — at least one side must be TRUE.
- NOT — turns TRUE into FALSE and back.

```
DECLARE Age : INTEGER
DECLARE HasTicket : BOOLEAN
Age ← 16
HasTicket ← TRUE
IF Age >= 15 AND HasTicket = TRUE
    THEN
        OUTPUT "Admitted"
ENDIF
IF NOT (Age > 18)
    THEN
        OUTPUT "Still a student rate"
ENDIF
```

Common mistakes

- <> is the *not equal* sign here — not !=, which belongs to other languages.
- IF Age >= 15 AND <= 18. Each side of AND needs its own complete comparison:
Age >= 15 AND Age <= 18.
- Calling these *comparison operators* in an answer. The syllabus term is **relational**.