# 11.2 Constructs

Name: _____ Class: _____ Date: _____ **Total: 43 marks**

**KEY WORDS** selection • nested • logic error • pseudocode • condition

## Objective

Learn to write **selection** with the `IF` statement: `IF … THEN … ENDIF`, the `ELSE` clause, and one `IF` inside another (a **nested** `IF`). This is the first sheet of 11.2. Sheet 11.2.1 adds the `CASE` structure, and sheets 11.2.2 to 11.2.4 deal with loops and trace tables. The conditions themselves (`AND`, `OR`, ranges) were on sheet 9.2.1.

**You must be able to:**

- write an `IF` statement with and without an `ELSE` clause
- write nested `IF` statements to choose between three or more paths
- put the tests of a nested `IF` in the correct order
- follow an `IF` statement for given values, and find errors in one

## 1 Worked examples

Study these first. Each one shows a **method** that a question later on this sheet needs.

### ■ IF … THEN … ENDIF

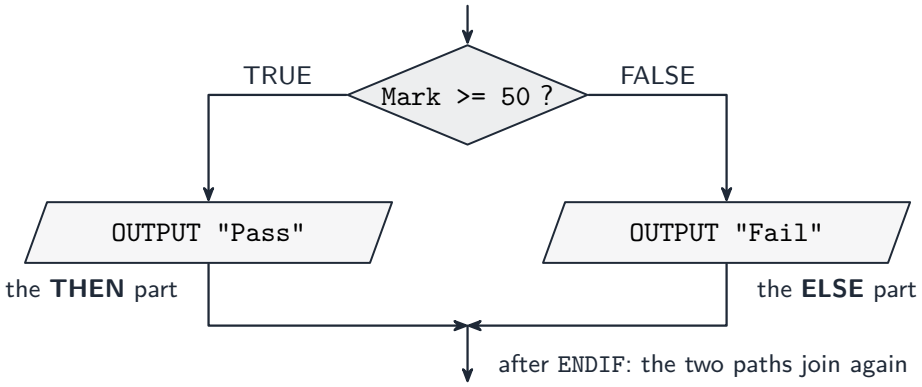
Use it when something is done **only if** a condition is TRUE. If the condition is FALSE, nothing happens.

```
IF Temperature < 0 THEN
    OUTPUT "Ice warning"
ENDIF
```

- The condition is a logic statement: it must be TRUE or FALSE.
- The statements between `THEN` and `ENDIF` are **indented**, and every `IF` needs its own `ENDIF`.

### ■ IF … THEN … ELSE … ENDIF

Use it to choose between **two** paths. Exactly one of them runs.




```
IF Mark >= 50 THEN
    OUTPUT "Pass"
ELSE
    OUTPUT "Fail"
ENDIF
```

**Following an `IF`:** work out the condition with the given values. TRUE → run only the `THEN` part. FALSE → run only the `ELSE` part. Then carry on after `ENDIF`.

### Nested IF: three or more paths

An `IF` inside the `ELSE` part of another `IF` gives a third path.

*Worked example: a mark of 70 or more gets a Merit, a mark of 50 or more gets a Pass, and any other mark gets a Fail.*

```
IF Mark >= 70 THEN
    OUTPUT "Merit"
ELSE
    IF Mark >= 50 THEN
        OUTPUT "Pass"
    ELSE
        OUTPUT "Fail"
    ENDIF
ENDIF
```

- The inner `IF` is reached only when `Mark >= 70` is FALSE. So at that point the mark is already known to be below 70, and `Mark >= 50` alone is enough. There is no need to write `Mark >= 50 AND Mark < 70`.
- Count the keywords: two `IF`s need two `ENDIF`s. The indentation shows which `ENDIF` belongs to which `IF`.

### The order of the tests matters

The **first** condition that is TRUE wins, and the other tests are never reached. Test the **most demanding** condition first.

Wrong order:

```
IF Mark >= 50 THEN
    OUTPUT "Pass"
ELSE
    IF Mark >= 70 THEN
        OUTPUT "Merit"
    ENDIF
ENDIF
```

For a mark of 85, `Mark >= 50` is TRUE, so the output is `"Pass"`: the test `Mark >= 70` is never reached. This is a **logic error** : the program runs, but the result is wrong.

### Common errors

| error | example | correction |
|---|---|---|
| ← used to compare | `IF Age ← 18 THEN` | `IF Age = 18 THEN` |
| no `ENDIF` | an `IF` left open | one `ENDIF` for each `IF` |
| a range as in mathematics | `IF 1 <= X <= 10 THEN` | `IF X >= 1 AND X <= 10 THEN` |
| an `ELSE` with a condition | `ELSE Mark < 50` | `ELSE`, then a nested `IF` |

**Past-paper practice:** complete sheets 9.2.1, 11.2 and 11.2.1 before attempting past-paper set 11.2.1.

## 2 Practice

Now use the methods above. The first questions need one idea each.

**2.1** Study this pseudocode.

```
IF Speed > 60 THEN
    OUTPUT "Too fast"
ELSE
    OUTPUT "OK"
ENDIF
OUTPUT "Checked"
```

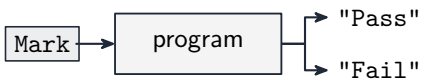
60

State everything that is output when:

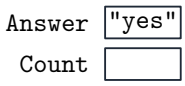
(a) `Speed` is 75 [1]

(b) `Speed` is 60 [1]

**2.2** Write an `IF…ELSE` that outputs `"Pass"` when `Mark >= 50`, otherwise `"Fail"`. [2]




**2.3** Write an `IF` statement that adds 1 to `Count` when `Answer` is equal to `"yes"`. There is no `ELSE` clause. [2]



**2.4** Each `IF` statement contains **one** error. State the error and write the correct first line. [4]

(a) `IF Age ← 16 THEN`

(b) `IF 0 <= Mark <= 100 THEN`

**2.5** Study this pseudocode.

```
IF Age < 5 THEN
    Price ← 0
ELSE
    IF Age < 16 THEN
        Price ← 6
    ELSE
        Price ← 10
    ENDIF
ENDIF
```

ticket

State the value of `Price` when:

(a) `Age` is 3 [1]

(b) `Age` is 15 [1]

(c) `Age` is 16 [1]

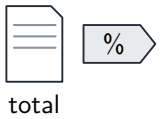
**2.6** A student writes the pseudocode in 2.5 again, but leaves out the last `ENDIF`. State why this is an error. [1]

## 3 Exam-style questions

**3.1** A shop gives a discount that depends on the total of an order:

- a total of 200 or more: a discount of 20%
- a total of 100 or more, but less than 200: a discount of 10%
- any other total: no discount.

Write pseudocode that inputs `Total`, and outputs the amount to pay after the discount. Use nested `IF` statements, and declare your variables. [6]




**3.2** A student writes this pseudocode to output a message about the temperature of a room.

```
IF Temp > 18 THEN
    OUTPUT "Comfortable"
ELSE
    IF Temp > 28 THEN
        OUTPUT "Too hot"
    ELSE
        OUTPUT "Too cold"
    ENDIF
ENDIF
```

(a) State what is output when `Temp` is 30. [1]

(b) Explain why `"Too hot"` can never be output. [2]

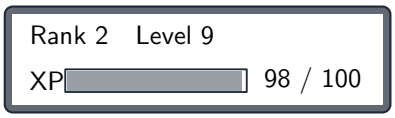
(c) Rewrite the pseudocode so that it works correctly. A temperature above 28 is too hot, and a temperature of 18 or below is too cold. [3]

**3.3** A program inputs three different integers `A`, `B` and `C`, and outputs the largest of them.

Write pseudocode for the program. Declare your variables. [5]

**3.4** A game stores a player's progress in three integer variables `Rank`, `Level` and `XP` (experience points). For example, a player at rank 2 and level 9 with 98 experience points has `Rank` = 2, `Level` = 9 and `XP` = 98.

(a) Write pseudocode that adds **one** experience point. `XP` goes back to 0 when it reaches 100, and `Level` then goes up by 1. `Level` goes back to 0 when it reaches 10, and `Rank` then goes up by 1. `Rank` is never more than 5: when it reaches 6, it is set back to 5. [6]




(b) Complete the table to show the values after your pseudocode has run **three** times, one after the other. [3]

| run | Rank | Level | XP |
|---|---|---|---|
| start | 2 | 9 | 98 |
| 1 | _____ | _____ | _____ |
| 2 | _____ | _____ | _____ |
| 3 | _____ | _____ | _____ |

(c) A student does not nest the `IF` statements. After adding 1 to `XP`, the student writes three separate `IF` statements in this order: the test of `Rank`, then the test of `Level`, then the test of `XP`. The player is at rank 2 and level 9, with 99 experience points. **State** the values that the student's pseudocode gives, and **explain** the error. [3]