# *Algorithm constructs*

Key algorithm constructs are:
- Sequence
- Selection
- Repetition

You also need to be able to identify
- Inputs and Outputs
- Variable declaration and initialisation
- Subprograms
- Data structures such as arrays (lists)

# *Algorithm constructs*

**Sequence** is simply the order that instructions follow on from each each.

If instructions are out of sequence the result of the algorithm might not be what is intended.

A **trace table** can be used to check that the sequence of instructions is correct.

# *Algorithm constructs*

**Selection** is the use of a decision within a program used to decide what to do next in a program.

Selection uses **IF** statements.
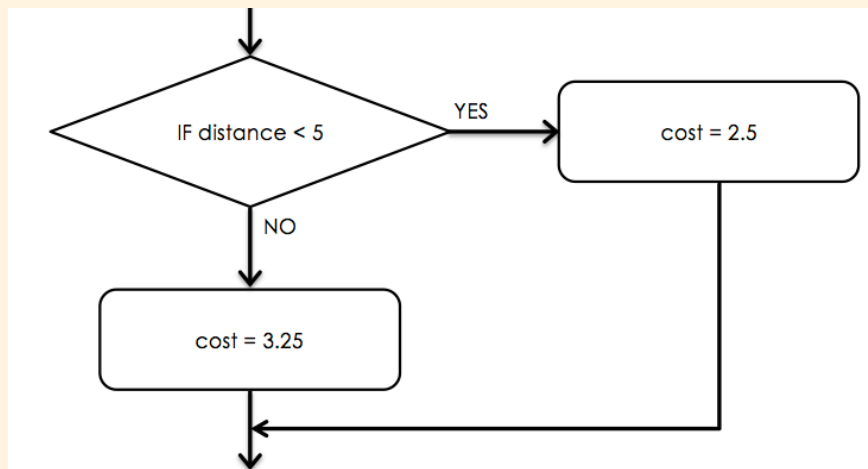


```
11
12    IF distance < 5 THEN
13        SET cost TO 2.5
14    ELSE
15        SET cost TO 3.25
16    END IF
17
```

```
IF quickTime > times[i] THEN
    SET quickTime TO times[i]
END IF
```

# *Algorithm constructs*

**Repetition** is a block of code which is executed more than once – it is repeated.

Also known as **Iteration**.

Repetition uses **FOR** or **WHILE** loops.

```
5
6      FOR i FROM 1 TO LENGTH(times) DO
7          SET totalTime TO totalTime + times[i]
8      END FOR
9
```

```
10
11         WHILE check = "N" DO
12             RECEIVE name FROM (STRING) KEYBOARD
13             SEND name TO DISPLAY
14
15             SEND "Is this correct?" TO DISPLAY
16             RECEIVE check FROM (STRING) KEYBOARD
17         END WHILE
18
```

# *Algorithm constructs*

**Inputs** allow data to be entered into an algorithm – either by a user to through some other kind of input device.
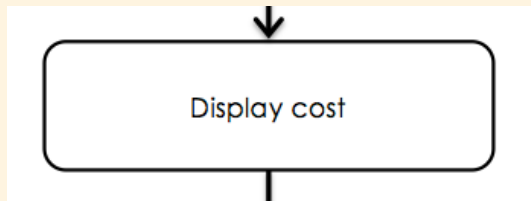
Inputs use **RECEIVE FROM** commands in pseudocode
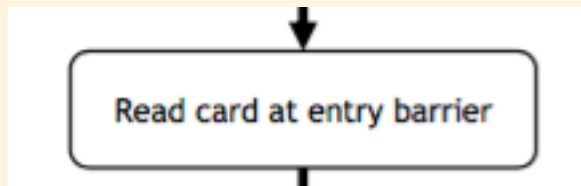


```
1    RECEIVE name FROM (STRING) KEYBOARD
2
3    RECEIVE cost FROM (REAL) KEYBOARD
4
```

# *Algorithm constructs*

**Outputs** allow data to be sent from the algorithm.

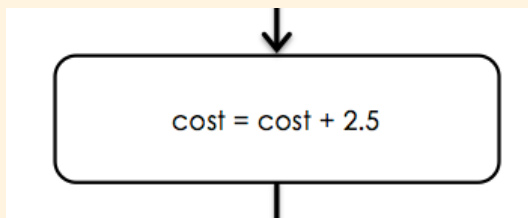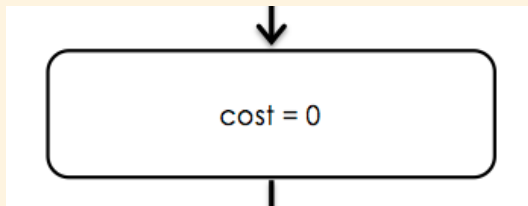Outputs use **SEND TO** commands in pseudocode


Display cost


```
8
9    SEND cost TO DISPLAY
10
```

# *Algorithm constructs*

**Variables** are named area of memory which hold data.

☐ Declaration creates the variable.

☐ Initialisation sets the first value in a variable

cost = 0

cost = cost + 2.5

```
5    SET cost TO 0
6
7    SET cost TO cost + 2.5
```

Usually declaration and initialisation occur at the same time

# *Algorithm constructs*

**Subprograms** break programs down into smaller blocks.

Subprograms are **FUNCTIONS** or **PROCEDURES** in pseudocode.

```
19
20   PROCEDURE calcCost (distance)
21   BEGIN PROCEDURE
22       SET cost TO 0
23       IF distance < 5 THEN
24           SET cost TO 2.5
25       ELSE
26           SET cost TO 3.25
27       END IF
28
29       SEND cost TO DISPLAY
30   END PROCEDURE
```

```
31
32   FUNCTION updateBalance (balance, cost)
33   BEGIN FUNCTION
34       SET balance TO balance - cost
35       SEND "Your balance is ", balance TO DISPLAY
36       RETURN BALANCE
37   END FUNCTION
38
```

In pseudocode, FUNCTIONS return values, PROCEDURES don't

# *Algorithm constructs*

**Data Structures** hold data in **Arrays** or **Lists**. These allow more than one data item to be held within a named area.

To process Arrays or Lists you usually use a loop to work through the structure.

```
 5
 6        FOR i FROM 1 TO LENGTH(times) DO
 7            SET totalTime TO totalTime + times[i]
 8
 9            IF quickTime > times[i] THEN
10                SET quickTime TO times[i]
11            END IF
12        END FOR
```

# *Algorithm constructs*

You may see **subprograms** and **inputs/outputs** shown differently on some flowcharts.
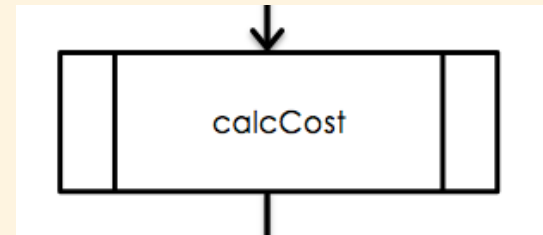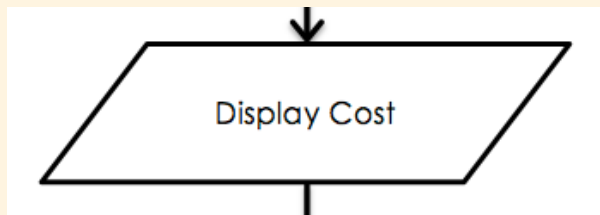




These symbols are not on the syllabus so you shouldn't see them in an exam