

Introduction Pseudo-Code

The Sequence (instructions in series)

```
[ some statement  
  another statement  
  another statement
```

If-Then-Else statement (decision)

```
[ If      some condition is TRUE  
  Then   [ some statement  
          another statement  
  Else   [ If some other condition is TRUE  
          Then [ some statement  
              another statement  
          Else (maybe) no statement
```

Repeat-Until statement (loop)

```
REPEAT  
  [ some statement  
    another statement  
UNTIL  the condition is TRUE
```

While-Do statement (loop)

```
WHILE the condition is TRUE do
  [
  some statement
  another statement
  ]
ENDWHILE
```

For number-of-times Do statement (loop)

```
FOR counter is 1 to n do
  [
  some statement
  another statement
  ]
ENDFOR
```

Case statement (multiple decision)

```
CASE condition is :
  value-1 : [
  statement
  another statement
  ]
  value-2: [
  statement
  another statement
  ]
  value-3: [
  statement
  another statement
  ]
ENDCASE
```

(more) Keywords:

Read , Write (also: Input, Ouput)	used for input description
true , false	used to state a logical result
Add , Subtract	
Multiply , Divide	used for calculations
AND , OR , NOT	used for combined Conditions

Hints and Tips:

In principal there are 3 Pseudocode structures:

Sequence, Loop and Decision.

With those three, you can describe any Algorithm (solution to a problem), using Pseudo-code.

The Three Structures Listed:

Sequence: any statement in a row

Loop: Repeat-Until
Condition is checked at the end of the loop

While-Do
Condition is checked at the very beginning

For number-of-times Do
Loop 'runs' a specific amount of times

Decision: If-Then-Else
A logical decision, can be true or false only!

Case
A multiple logical decision (like using more than one If-Then-Else at the same time)

Some examples

If-Then_Else (decision)

Problem: Read one number and check if it is positive!

```
Read      number
┌ If      number > zero
│ Then    Write "Number is Positive!"
└ Else    Write "Number is Negative!"
```

This simple algorithm will read one Number as Input (from the keyboard if nothing else is specified), will check if that Number is Positive or not and will print the result (on the Monitor, if nothing else is specified)

The solution above will not solve all cases though! That's because the number could also be zero, and then you cannot tell whether it is positive or negative! So we improve our solution to the problem by changing the PSEUDOCODE to:

```
Read number
┌ If      number > zero
│ Then    Write "Number is Positive!"
└ Else    ┌ If      number = zero
           │ ThenWrite "Number is Zero!"
           └ Else    Write "Number is Negative!"
```

Suppose we would like to use the CASE-statement here, then it would look like following:

```
Read  Number
CASE  Number is :
    > zero : Write "Number is Positive!"
    = Zero : Write "Number is zero!"
    < Zero : Write "Number is Negative!"
ENDCASE
```