pseudo-code

what is pseudo-code?

Pseudo-code is a method of expressing an algorithm independently from a given programming language. Therefore, it should be easier to understand for all programmers as it does not rely on specific knowledge of a particular programming language.

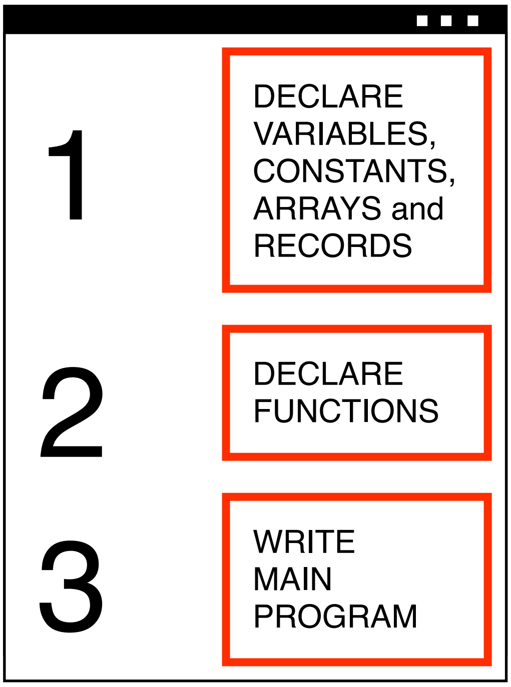
It is generally used at the planning stage, before the choice of programming language has been made.

structuring pseudo-code

All programming languages have a method of structuring code so that it can be understood but the problem is that they are all different! For instance, in Python you can declare variables anywhere in the program but in Pascal you must declare them at the top of the program.

Pseudo-code uses the following convention:

- **First** declare any variables
- **Second** declare functions
- **Third** write the main program



1 DECLARE VARIABLES, CONSTANTS, ARRAYS and RECORDS

2 DECLARE FUNCTIONS

3 WRITE MAIN PROGRAM

Variables

	Syntax	Examples

Declaring	`VariableName: DataType`	`TestScore1: Integer`
		`IsPass: Boolean`
		`StudentName: String`
		`TestScore1, TestScore2: Integer`

Assigning	`VariableName ← Value`	`TestScore1 ← 5`
		`IsPass ← TRUE`
		`StudentName ← “John”`

Using	`INPUT VariableName (input from keyboard)`
	`OUTPUT VariableName (output to screen)`
		`INPUT TestScore1`
		`OUTPUT IsPass`

Constants

	Syntax	Examples

Declaring	`CONSTANT ConstantName: DataType ← Value`	`CONSTANT MaxScore: Integer ← 20`
		`CONSTANT SchoolName: String ← “Long Road”`

Assigning	Once a constant has been declared its value can’t be changed	N/A

Using	`VariableName ← VariableName * ConstantName`
	`OUTPUT VariableName (output to screen)`
		`VAT ← SubTotal * VatRate`
		`OUTPUT VAT`

Arrays (Lists in Python)

	Syntax	Examples
Declaring	ARRAY ArrayName: DataType [Size]* ARRAY ArrayName: DataType [Size] [Size]** * 1 Dimensional Array ** 2 Dimensional Array	ARRAY ClassScores: Integer [10] ARRAY ClassScores: Integer [∞]* ARRAY ClassScores: Integer [10][2] ARRAY ClassScores: Integer [∞][∞] *would be used to represent a dynamically sized list in Python
Assigning	ArrayName [Position] ← Value ArrayName [Position] [Position] ← Value	ClassScores [1] ← 5 ClassScores [1] [3] ← 5
Using	INPUT ArrayName [Position] OUTPUT ArrayName [Position]	INPUT ClassScores [2] OUTPUT ClassScores [2]

Records

	Syntax	Examples

Declaring	RECORD RecordName IS
	    AttributeName: DataType
	END RECORD
	
	Any required records should be declared before the statements in a main program or function
		RECORD Student IS
		   FirstName: String
		   Surname: String
		   PassedGCSE: Boolean
		   Age: Integer
		END RECORD

Assigning	RecordName.AttributeName ← Value
	ListName[Position].AttributeName ← Value*
	
	*It sometimes useful to have a list of records
		Student.FirstName ← “John”
		StudentList[1].FirstName ← “John”

Using	INPUT RecordName.AttributeName OUTPUT RecordName.AttributeName* *You can’t just INPUT RecordName or OUTPUT RecordName as you must tell the programmer exactly what is to be done	INPUT Student.FirstName INPUT StudentList[1].FirstName OUTPUT Student.FirstName OUTPUT StudentList[1].FirstName

Selection Statements (IF)

Syntax	Examples

IF Condition THEN
   Statement
ELSE IF
   Statement
ELSE
   Statement
END IF
	IF TestScore1 > 5 THEN
	   ClassScores[3] ← TestScore1
	END IF
	
	IF TestScore1 > 5 THEN
	   OUTPUT “PASS”
	ELSE
	   OUTPUT “FAIL”
	END IF
	
	IF TestScore1 > 5 THEN
	   OUTPUT “A”
	ELSE IF TestScore 1 > 3
	   OUTPUT “B”
	ELSE
	   OUTPUT “FAIL”
	END IF

Iteration Statements (FOR,WHILE)

<table>
<tr><th>Syntax</th><th>Examples</th></tr>
<tr><td><pre>
FOR CounterVariable ← StartValue TO EndValue DO
   Statement
END FOR
</pre></td><td><pre>
FOR EachRow ← 1 TO 10 DO
    OUTPUT “Please enter a test score:”
    INPUT TestScores[EachRow]
END FOR
</pre></td></tr>
<tr><td><pre>
WHILE Condition DO
    Statement
END WHILE
</pre></td><td><pre>
WHILE NoMoreScores = FALSE DO
   Counter ← Counter + 1
   OUTPUT “Please enter a test score:”
   INPUT TempScore
   IF TempScore = -1 THEN
       NoMoreScores ← TRUE
   ELSE
      TestScores[Counter] ← TempScore
   END IF
END WHILE
</pre></td></tr>
</table>

Functions

<table>
<tr><th></th><th>Syntax</th><th>Examples</th></tr>
<tr><td>Declaring</td><td><pre>
FUNCTION FunctionName (Parameter(s): DataType)
    Local Variable Declarations
    Statements
    RETURN Value
END FUNCTION
</pre></td><td><pre>
FUNCTION DisplayMenu
    OUTPUT “Main Menu”
    OUTPUT “1. Play Game”
    OUTPUT “2. Show Scores”
    OUTPUT “3. Exit”
END FUNCTION

FUNCTION GetName
    TempName: String
    OUTPUT “Please enter your name:”
    INPUT TempName
    RETURN TempName
END FUNCTION

FUNCTION GetMove (CPUMove: Boolean)
    X,Y: Integer
    IF CPUMove = FALSE THEN
        OUTPUT “Please enter X co-ordinate:”
        INPUT X
        OUTPUT “Please enter Y co-ordinate:”
        INPUT Y
    ELSE
        X ← CALL RANDOM(10)
        Y ← CALL RANDOM(10)
    END IF
    RETURN X,Y
END FUNCTION
</pre></td></tr>
<tr><td>Using</td><td><pre>
CALL FunctionName (Parameter(s))
VariableName ← CALL FunctionName (Parameter(s))*

*If the function has a return value it must be
used in this way
</pre></td><td><pre>
CALL DisplayMenu

CALL Random(10)

CALL Random(1,20)

Coordinates ← CALL GetMove(TRUE)
</pre></td></tr>
</table>

Text Files

	Syntax	Examples

Declaring	`FileVariableName ← OPEN ‘FileName.txt’ METHOD`	`HiScores ← OPEN ‘HiScores.txt’ READ`
		`HiScores ← OPEN ‘HiScores.txt’ WRITE`

Using	`INPUT FROM FileVariableName VariableName`
	`OUTPUT TO FileVariableName Value`
	`CLOSE FileVariableName`
		`FOR Next ← 1 TO 10 DO`
		`    INPUT FROM HiScores Scores[Next]`
		`END FOR`
		
		`For Next ← 1 TO 10 DO`
		`   OUTPUT TO HiScores Scores[Next[`
		`END FOR`
		
		`CLOSE HiScores`

Binary Files

	Syntax	Examples

Declaring	`FileVariableName ← OPEN BINARY ‘FileName.dat’ OF DataType METHOD`	`HiScores ← OPEN BINARY ‘HiScores.dat’ OF Integer READ`
		`HiScores ← OPEN BINARY ‘HiScores.dat’ OF Integer WRITE`

Using	`INPUT FROM FileVariableName VariableName`
	`OUTPUT TO FileVariableName Value`
	`CLOSE FileVariableName`
		`FOR Next ← 1 TO 10 DO`
		`    INPUT FROM HiScores Scores[Next]`
		`END FOR`
		
		`For Next ← 1 TO 10 DO`
		`   OUTPUT TO HiScores Scores[Next[`
		`END FOR`
		
		`CLOSE HiScores`