Summary of Pseudo-code Language Constructions

An algorithm is an ordered sequence of unambiguous and well-defined instructions that performs some tasks. Pseudo-code allows ones to focus on the logic of the algorithm without being distracted by details of language syntax. At the same time, the pseudo-code needs to be complete. It describes the entire logic of the algorithm so that implementation is a task of translating line by line into source code.

Three Categories of Algorithmic Operations

Three basic constructs for flow of control are sufficient to implement any “proper” algorithm.

1. sequential operations (**Sequence**) - instructions are executed in order
2. conditional operations (**If-Then-Else**) - a control structure that asks a true/false question and then selects the next instruction based on the answer
3. iterative (loop) operations (**While**) - a control structure that repeats the execution of a block of instructions

Although these constructs are sufficient, it is often useful to include three more constructs:

4. **Repeat-Until** is a loop with a simple conditional test at the bottom.
5. **Case** is a multiway branch (decision) based on the value of an expression. **Case** is a generalization of **If-Then-Else**.
6. **For** is a “counting” loop.

Pseudo-code Structure:

INPUT:
OUTPUT:
Step1:
Step2:
etc…

Computation/Assignment

```
set the value of "variable" to :"arithmetic expression" or
"variable" equals "expression" or
"variable" = "expression"
```

Input/Output

```
get "variable", "variable", ...
display "variable", "variable", ...
```

Conditional

```
if  "condition" then
        (subordinate) statement 1
         etc ...
else
        (subordinate) statement 2
        etc ...
```

Iterative

```
while "condition"
        (subordinate) statement 1
        (subordinate) statement 2 ...

for "iteration bounds"
        (subordinate) statement 1
        (subordinate) statement 2 ...
```