START – NEXT

This is a simple loop, the syntax is like this:

```
InitialValue FinalValue
START
  [body of the loop]
NEXT
```

When the program find a START, a internal variable is initialized to InitialValue Then the body of the loop is executed and when the NEXT is reached the internal variable increments in 1 and if its value is less or equal than FinalValue the control return to START and the loop is executed again, if the internal variable value is greater than FinalValue the control exits the loop and continue to the next statement in the program after NEXT. If the FinalValue is less or equal than the InitialValue the loop is executed just once. Notice that you don't have access to the internal variable.

Example 1:

```
« 1 3
   START
     "RICK"
   NEXT
»
```

This program put the word RICK in level 1:, 2: and 3:.

Example 2:

```
« 1 2.5
   START
     "RICK"
   NEXT
»
```

This program put the word RICK in level 1: and 2:.

START – STEP

This is a more complex loop, the syntax is :

```
InitialValue FinalValue
START
  [body of the loop]
  StepValue
STEP
```

StepValue has to be right before STEP and can be a constant value or a value that is calculated each time the loop is executed, it also can be positive or negative. We have two cases here.

1. **InitialValue is less than FinalValue**: In this case StepValue has to be positive and the way this loop works is similar to the START-NEXT loop, except that when the STEP is reached the

internal variable is incremented in StepValue instead of 1. If StepValue is negative, the control exits the loop and continue to the next statement in the program.

2. **InitialValue is grater than FinalValue**: In this case StepValue has to be negative. When START, is reached, a internal variable is initialized to InitialValue Then the body of the loop is executed, then, when the STEP is reached the internal variable is decremented in StepValue, if its value is grater or equal than FinalValue the program control return to START and the loop is executed again, if the internal variable value is less than FinalValue, the control exits the loop and continue to the next statement in the program. If StepValue is positive, the control exits the loop and continue to the next statement in the program.

You can try simple programs like:

Example 1:

```
« 2 6
  START
   "RICK"
   2
  STEP
»
```

and see how this works.

Example 2: The next sample generates a random number between 0 and 9 and in 10 interaction generates different random numbers (between 0 and 9), in each interaction the program compares if the new number generated is equal to the first number generated, if the numbers are equal the program tags the first number with the word "EQUAL" and exits the loop (by setting StepValue to -1) or else sets StepValie to 1 to continue the loop. At the end if none of the numbers generated were equal to the first number generated, the program just shows the first number generated (with no tag).

```
«
 RAND 10 * IP               @ Generates the first number
 1 10
 START                      @ Starts a loop from 1 to 10
   IF RAND 10 * IP OVER ==  @ Generate another number and compares them
   THEN "EQUAL" →TAG -1     @ Tags the number and sets StepValue to -1
   ELSE 1                   @ Sets StepValue to 1
   END
 STEP                      @ End of the loop
»
```