Flow Control

This cheat sheet tells how to influence the order in which commands are executed. It shows how to use loops, jumps, subroutines, and how to end a program. You will usually only use these things in a program, not at the command prompt.

Jumping to a different line

To continue execution at an arbitrary line, and not the next one as would usually be the case, use the GOTO command.

```
20 GOTO10
```

Running a subroutine

To run (or ”call”) a subroutine, use the CALL command:

```
40 CALL myroutine
50 ...
```

This will call the subroutine called myroutine, and once myroutine is finished, execution will resume at line 50. For this to work, the subroutine has to be defined somewhere using the PROC command. The subroutine also has to ”return” at some point:

```
100 PROC myroutine
110 PRINT"Subroutine!"
120 ...
130 RETURN
```

⚠**When writing a subroutine, you must always make sure that the RETURN command is executed at some point.**

Running the same piece of code a number of times

To execute a part of your program several times, and keep count in a variable while doing so, use a FOR loop:

```
10 FOR A=1 TO 5
20 ...
30 ...
40 NEXT
```

The code in lines 20 and 30 will be executed five times. The first time, the variable A is set to 1, the second time to 2, etc.

Running the same piece of code until a condition is met

If you want to repeat a portion of your program until a certain condition is met, use a DO/LOOP loop:

```
10 DO
20 INPUT A
30 LOOP UNTIL A=42
```

Line 20 will be repeated until the user enters the number 42.

It is also possible to loop *as long as* a condition is met, using a DO/LOOP WHILE loop.

Doing things only if a condition is met

To run a command only if a condition is true, use the IF/THEN command:

```
50 IF A=1 THEN PRINT"A is 1!"
```

Here, the text A is 1! is only printed if variable A has the value 1.

Doing different things depending on a condition

If you want to do one thing if a condition applies, and a different thing if it does not, use an IF/THEN/ELSE command:

```
50 IF A=1 THEN PRINT"A is 1!" ELSE PRINT"A is not 1!"
```

Here, the text A is 1! is printed if variable A has the value 1, and A is not 1! is printed in any other case.

Running longer program blocks under a condition

Oftentimes you will want to make more than a single command conditional using IF. That is possible if you start a new line after the THEN statement, and later say ENDIF once the conditional part of your program block is finished:

```
50 IF A>100 THEN
60 PRINT"That's too much!"
70 A=100
80 ENDIF
```

If the variable A has a value above 100, the string "That's too much!" will be printed, and A will be reset to 100.

You can combine this with an ELSE block; add the lines

```
72 ELSE
74 PRINT"Close enough"
```

to print "Close enough" if the value of A is 100 or less.

Ending a program

To end a program before its last line has been executed, use the END command:

```
450 END
460 REM This line will not be run.
```

Aborting a program

If you notice that something does not look right, you can abort the program:

```
100 IF A$="Uli ist doof!" THEN STOP
```

This will halt execution if the string variable A$ contains the text "Uli ist doof!" and print the message Break in line 100 so you know where to look for the error.