# How your program syntax can look like

**syntax** argument (with or without squared brackets) specifier (with or without squared brackets) , OPTion (with or without squared brackets; specifications - if used - in normal brackets)

| type | options |
|---|---|
| - | |
| varlist<br>varname<br>newvarlist<br>newvarname | min=#; max=#; fv; ts; numeric/string/str#/strL; generate (only for newvar*) |
| namelist<br>name | local; min=#; max=# |
| anything | |

| type | options |
|---|---|
| - | |
| if | / |
| in | / |
| using | / |
| fweight aweight pweight iweight | / |
| =exp | =/exp |

| type | options |
|---|---|
| - | |
| integer | # |
| real | # |
| numlist | int; >#; >=#; <#; <=#; min=#; max=#; sort |
| varname/varlist | numeric/string; min=#; max=#; fv; ts |
| name/namelist | local; min=#; max=# |
| string | asis |
| passthru | |

**Notes:**

- ***This list is not complete***. See the pdf-manual (accessible via the "syntax" help file) for more.
- Squared brackets make a phrase optional.
- You can only specify up to ONE argument (but as many options as you like).
- Using "/" after "if", "in", "using" or "weight" changes how the information is stored in the local. For expressions, specify "=/exp" instead.
- You can name the options as you like (if it is local compatible), but the locals which store the input from an option are called the same as the option.
- With "integer" and "real", you can specify default values (this is what # stands for).
- Sometimes, it makes sense to specify "name" instead of "string": It only allows strings which could be used as names for objects such as matrices. In other words, only one element, no special characters etc. Plus, you can specify min & max.

# Example

accepts no argument (there is nothing between syntax and the comma except for if/in/using)

“in” optional, stores the phrase in local `in’ ***without*** the word “in”

***allows*** the option root with an integer, otherwise takes the integer 2 as default, and stores the input in the local `root’

```
syntax [if] [in/], RUNning(varlist numeric) [root(integer 2)]
```

“if” optional, stores the phrase in local `if’ ***with*** the word “if”

***requires*** the option running (abbrev. run), but only with numeric variables, and stores the input in the local `running’