NetLogo syntax

- Notes based on Programming Guide, examples at http://ccl.northwestern.edu/netlogo/docs/
- Variables
- Control
- Lists
- Randomization

Variables

- Naming: dashes are ok
- Scopes
  - Global variables
  - Agent variables
  - Parameters, Local variables

Global variables

- No type declarations
- Declare at top of file using `globals` keyword
- Ex: `globals [ cleaner DIRTY CLEAN ]`
- Enforcing constants not supported
  - “trust the programmer”

Agent variables

- Like fields in an OO language, one instance per agent within a type
- NetLogo predefines some agent variables
- You can add one using `turtles-own patches-own links-own`
- Ex: `turtles-own [ energy age ]`

Agent variables

- Indicate agent context first, then specify variable
- Ex: `ask turtle 0 [ set color black ]`
  - `set` is for assignment
- Ex: `[pcolor] of patch row col`

Parameters, Local variables

- Parameters added as optional list for procedures/reporters
- Ex: (`report` is like return)

```
to-report check [ row col ]
  report CLEAN = [pcolor] of patch row col
end
```

- Local variables declared using keyword `let`
- Ex: `let prey one-of sheep-here`
  - `one-of` randomly chooses one of an agentset
  - `sheep-here` generates all agents of the “sheep” breed at this location

Control

- if, ifelse
  - Conditional expressions use <, >, <=, >=, = (not ==), !=
  - Combine conditional expressions using not, and, or
  - Statement blocks marked by [and]
- Ex:

```
ifelse model-version = "sheep-wolves-grass" [
   report patches with [pcolor = green]
]
[ report 0 ]
```

Control

- Loops supported, though consider whether operation better supported for each agent in an agentset
- Ex: while loop (sample call: `show-num 5`)

```
; display 1 to num
to show-num [num]
  let counter 1
  while [counter <= num] [
    show counter ; show is like cout <<
    set counter counter + 1
  ]
end
```

Lists

- Space-separated, inside []'s
- Can mix types
- Can nest lists
- `n-values` is convenient for initialization
- Ex: reporter to produce 2-dimensional list of 0's

```
to-report make-2d [rows columns]
  report n-values rows [ n-values columns [0] ]
end
```

Indexing lists

- `item` command
  - item index list
  - Indexes start at 0
  - Incomplete by itself – like “list[0];” in C++
  - Ex: `show item 2 [1 true 4] ; displays 4`
- `replace-item` can be used to create a new list with one item replaced
  - Does not update list in place

Indexing lists

- `Ex:`
- update element of 2-dimensional list
- from https://stackoverflow.com/questions/23182872/how-to-do-replacing-item-in-use-nested-list

```
to-report update [matrix row col val]
  let current_row item row matrix
  report replace-item row matrix (replace-item
col current_row val)
end
```

Indexing lists

```
to try-update
  let temp [[1 2 3] [4 5 6]]
  show update temp 0 2 8
  show temp ; no change to temp
  set temp update temp 0 2 8
  show temp ; now it has changed
end
```

Randomization

- `random limit`
  - Returns random integer from 0 to limit-1
- `random-seed seed`
  - Set seed for generating random numbers
  - Seed must be in valid range for 4-byte integers
    - (NetLogo integers are generally 8 bytes)
  - Ex:

```
random-seed 5
random 100 ; produces 35
random 100 ; produces 10
```

If random-seed 5 run again, next 2 values generated by random 100 are 35, 10