

REPETITION IS SO SO CONVENIENT

the language for cell strain programs is a stack-based, reverse polish notation language, as are forth and postsript. a cell's memory is the language's stack. (V:1)

constant: a gene that, when executed, is pushed as a value onto the cell's* memory* stack.*

memory, memory stack: the first-in, last-out set of limited size where the cell stores integers. a cell dies* when it runs out of memory*.*

unit of memory (n): a position on the memory stack where a cell* may push an integer. a cell's memory is measured in units of memory.*

this title would be appropriate for a story about the application of pavlovian or mechanical laws to an organism which, like a fruit, was capable of colour and sweetness. — anthony burges, about the title “a clockwork orange”

it is a corollary of the principle of parsimony that leadership implies an economy of words. hey, bum! shave off that ugly beard with occam's razor!

every cell has its own memory stack, where it can push and pop integers. that is how a strain's program passes parameters to instructions of the language.

a cell's memory is really just the number of integers its stack can hold; as soon as the stack holds more numbers than memory permits, the cell dies.

instruction *repeat* lets a program execute the intruction that follows it a certain number of times, the exact number being taken off the stack.

after instruction *turn*, replace multiple calls to *step* with a single loop: push 42, followed by instructions *repeat* and *step*.

try values like 1066, 1759 and 1976.

click “fastest” to simulate as fast as your machine will run.