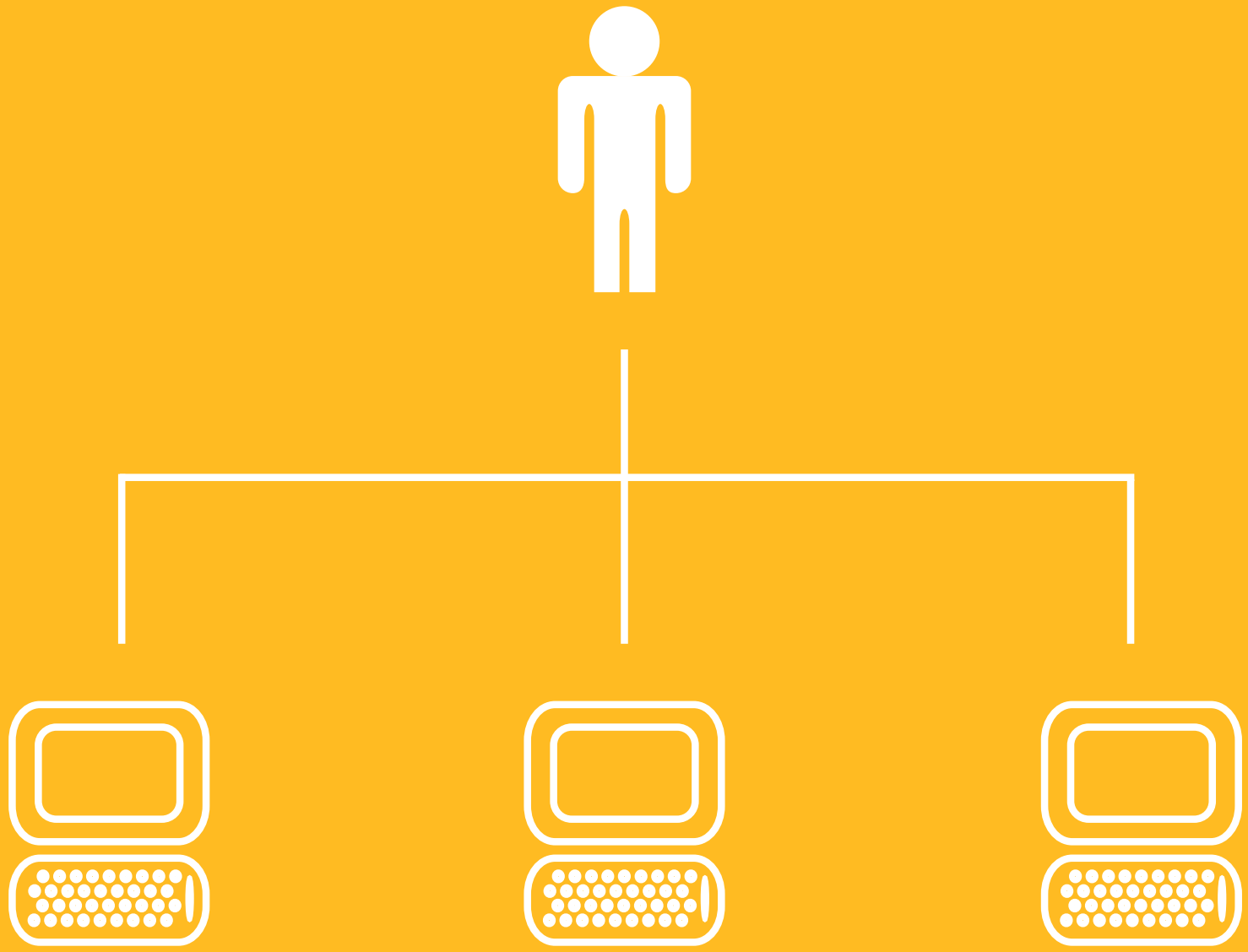
Only Perl
can *parse*
Perl 6

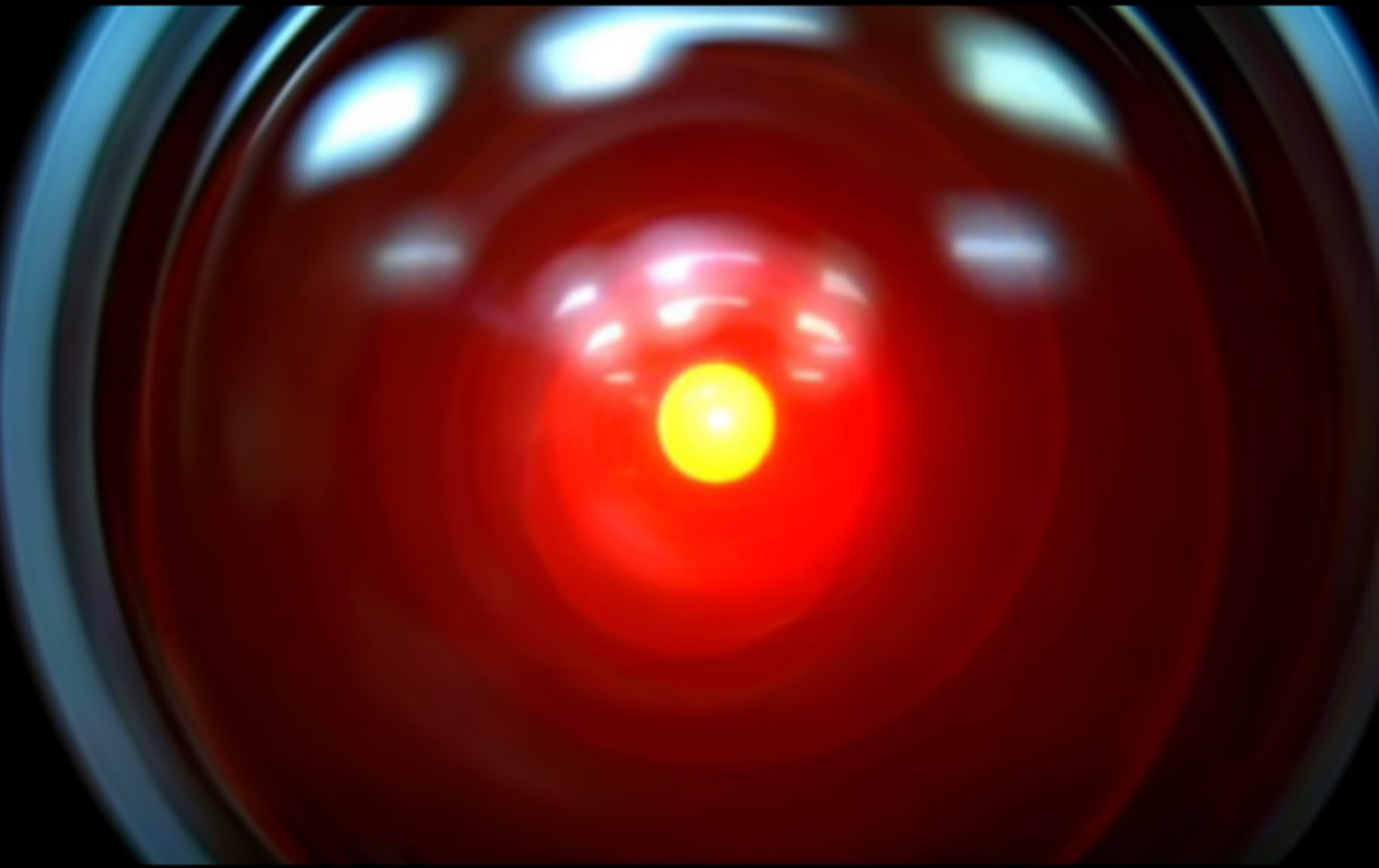
OH HAI

Carl Mäsak

γνῶθι σεαυτόν


LONDON
(no pun
intended)





why know itself?

how know itself?

if/while

introspection

are you a kind of Foo?

reflection

features from the language

accessible inside the language

metaobject protocol

metaobject

strange loops

metacircular evaluator

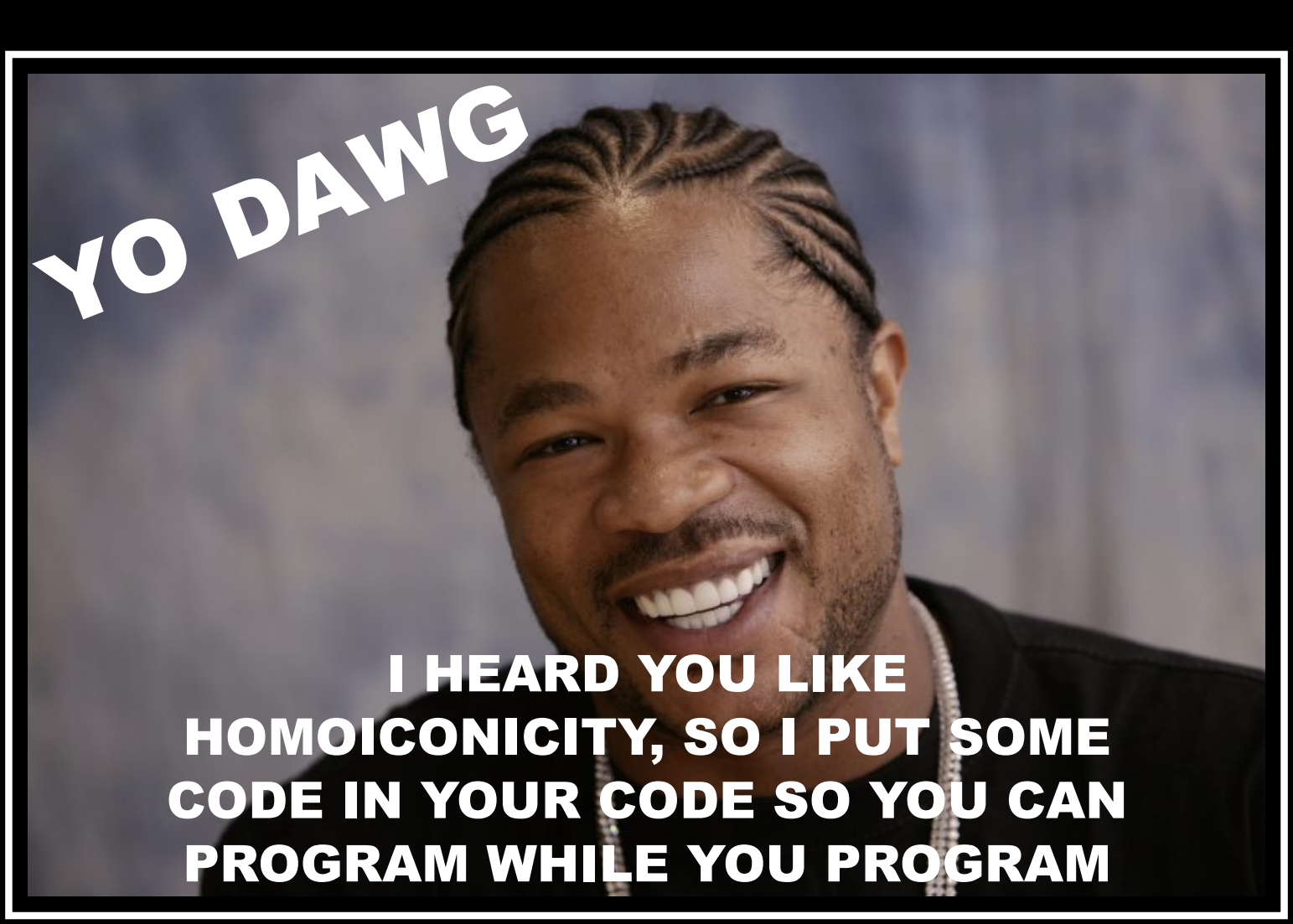
```
(define (eval exp env)
  (cond ((self-evaluating? exp) exp)
        ((variable? exp) (lookup-variable-value exp env))
        ((quoted? exp) (text-of-quotation exp))
        ((assignment? exp) (eval-assignment exp env))
        ((definition? exp) (eval-definition exp env))
        ((if? exp) (eval-if exp env))
        ((lambda? exp)
         (make-procedure (lambda-parameters exp)
                         (lambda-body exp)
                         env))
        ((begin? exp)
         (eval-sequence (begin-actions exp) env))
        ((cond? exp) (eval (cond->if exp) env))
        ((application? exp)
         (apply (eval (operator exp) env)
                (list-of-values (operands exp) env)))
        (else
         (error "Unknown expression type -- EVAL" exp))))

(define (apply procedure arguments)
  (cond ((primitive-procedure? procedure)
         (apply-primitive-procedure procedure arguments))
        ((compound-procedure? procedure)
         (eval-sequence
           (procedure-body procedure)
           (extend-environment
             (procedure-parameters procedure)
             arguments
             (procedure-environment procedure))))
        (else
         (error
          "Unknown procedure type -- APPLY" procedure))))
```

define the language

using the language

homoiconicity

program fragments as data structure


YO DAWG
I HEARD YOU LIKE HOMOICONICITY, SO I PUT SOME CODE IN YOUR CODE SO YOU CAN PROGRAM WHILE YOU PROGRAM

```
Name:Ddired-execute-do
Handles doing something on every marked line.  Execute arg1 with an
argument of the filename.
[*]#(Ddired-filename,(
    #(sp,^)
    #(==,##(rm,>),D,(
        #(==,arg1,,(
            #(sp,$>)
            #(==,##(rm,>),,,(
                #(SELF,(arg1))
            ))
        ))
    ),(
        #(sp,$>)
        #(==,##(rm,>),,,(
            #(SELF,(arg1))
        ))
    ))
))[*]
```

different kinds of knowing-yourself

now with the parser

parsing

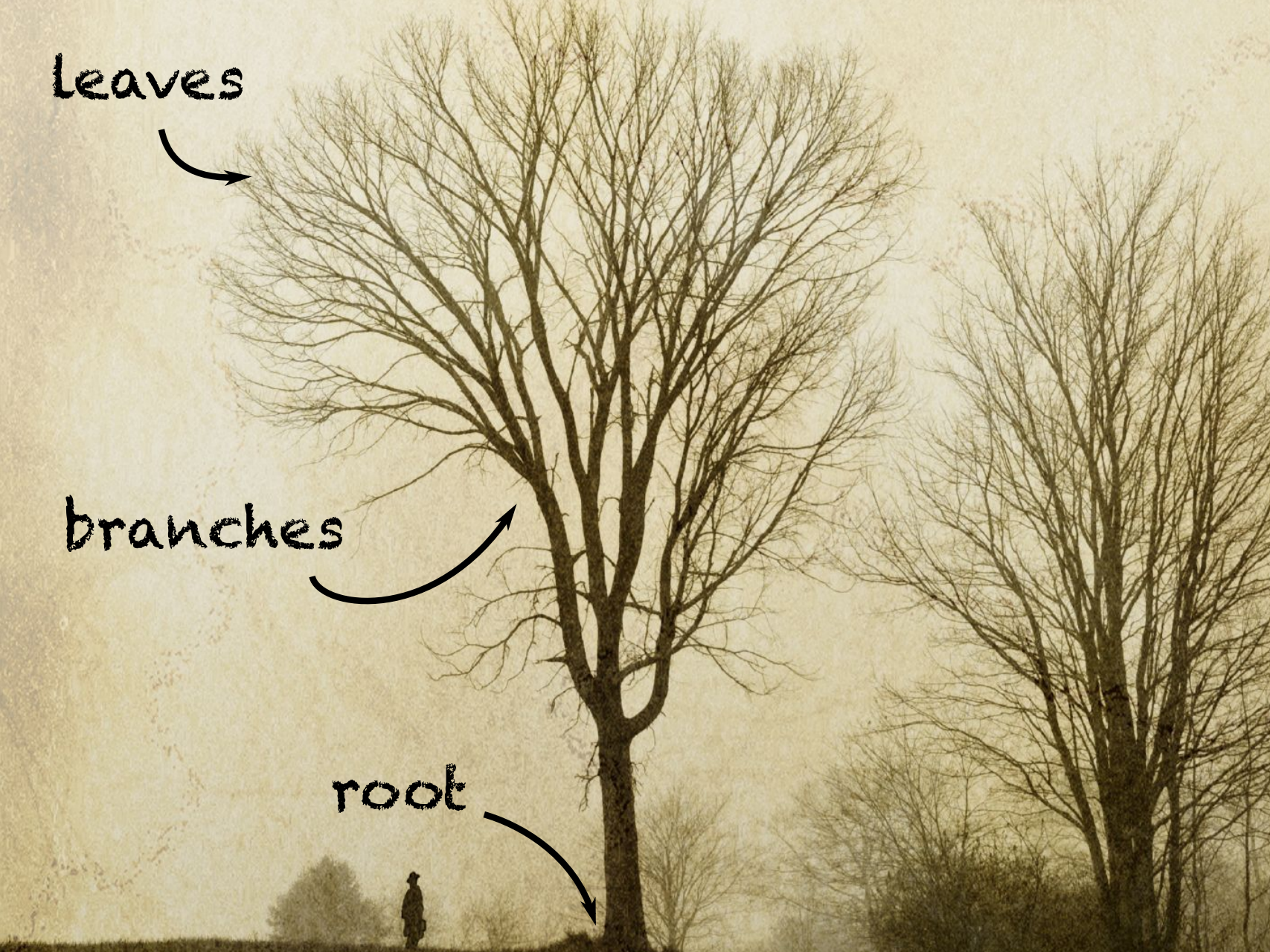
quick definition

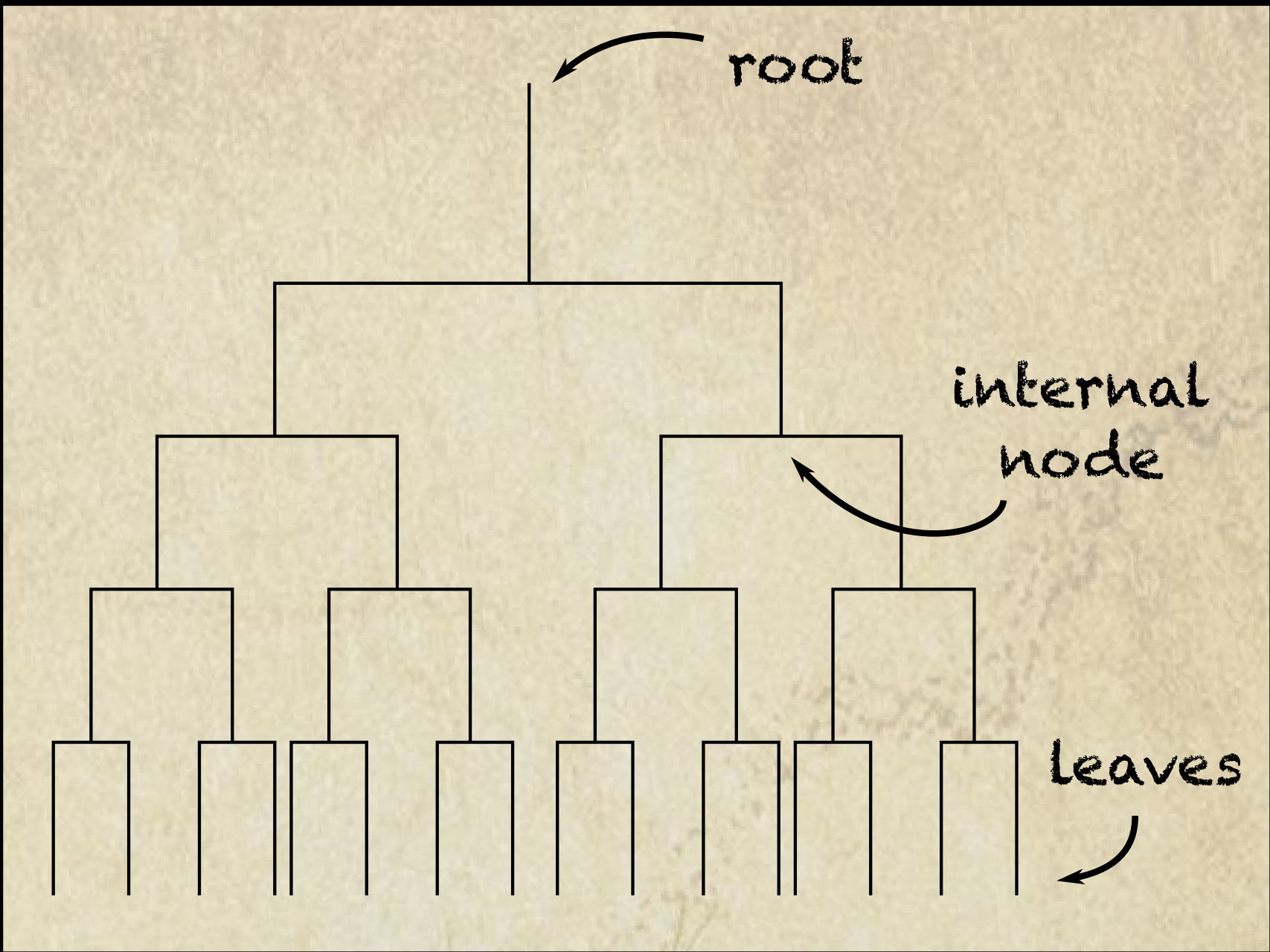
string ⇝ tree

why tree?

just what we need

nested regions in the string


leaves
branches
root


root
internal node
leaves

two kinds of parsing

bottom-up and top-down

bottom-up

start with leaves

glue pieces together

best suited for expressions

DSLs

Term 42 Lexical units

Operator + Lexical adhesive
Lexical wrappers

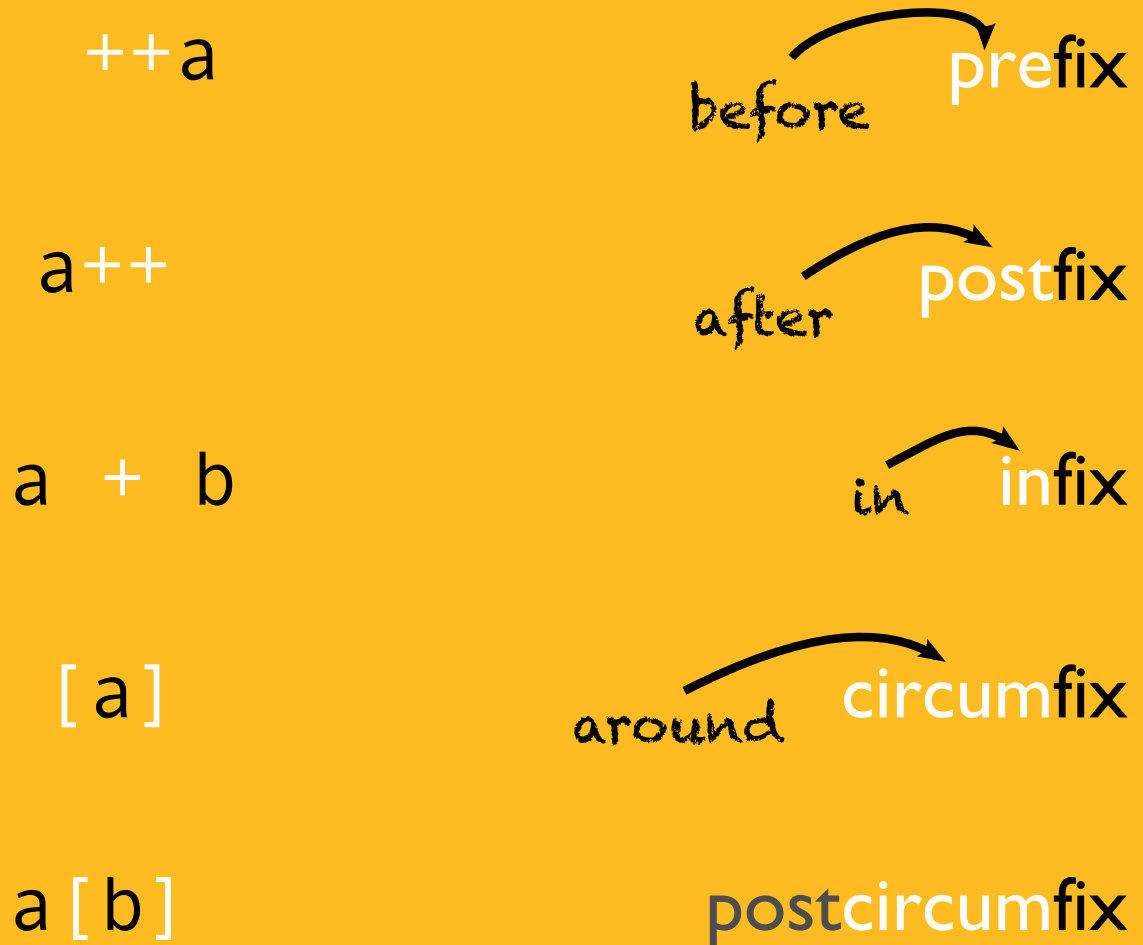
terms simple

(1, 2, 3)

"OH HAI $name"

bottom-up: operator precedence

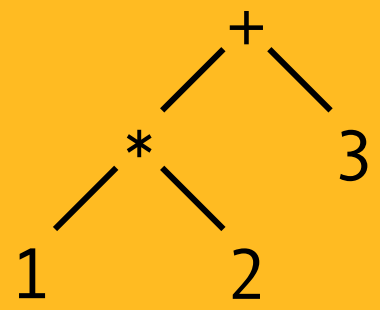
shift/reduce


++a
before
prefix
a++
after
postfix
a + b
in
infix
[a]
around
circumfix
a[b]
postcircumfix

Precedence

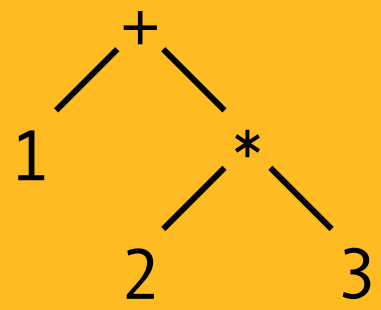
1 * 2 + 3

terms	ops
shift 1	
	shift infix:<*>
shift 2	
reduce △ * △	
	shift infix:<+>
shift 3	
reduce △ + △	



1 + 2 * 3

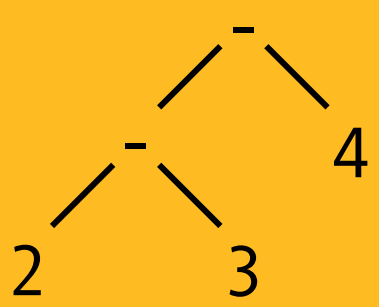
terms	ops
shift 1	
	shift infix:<+>
shift 2	
	shift infix:<*>
shift 3	
reduce △ * △	
reduce △ + △	



Associativity

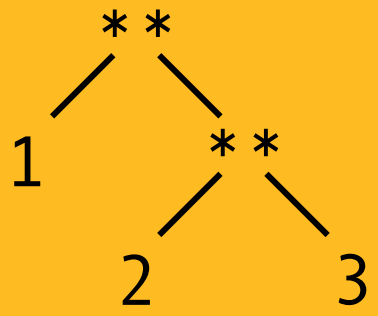
2 - 3 - 4

terms	ops
shift 2	
	shift infix:<->
shift 3	
reduce △ - △	
	shift infix:<->
shift 4	
reduce △ - △	



2 ** 3 ** 4

terms	ops
shift 2	
	shift infix:<**>
shift 3	
	shift infix:<**>
shift 4	
reduce △ ** △	
reduce △ ** △	



permutations of shift/reduce

```
@opstack[*-1]
```

```
$current-token
```

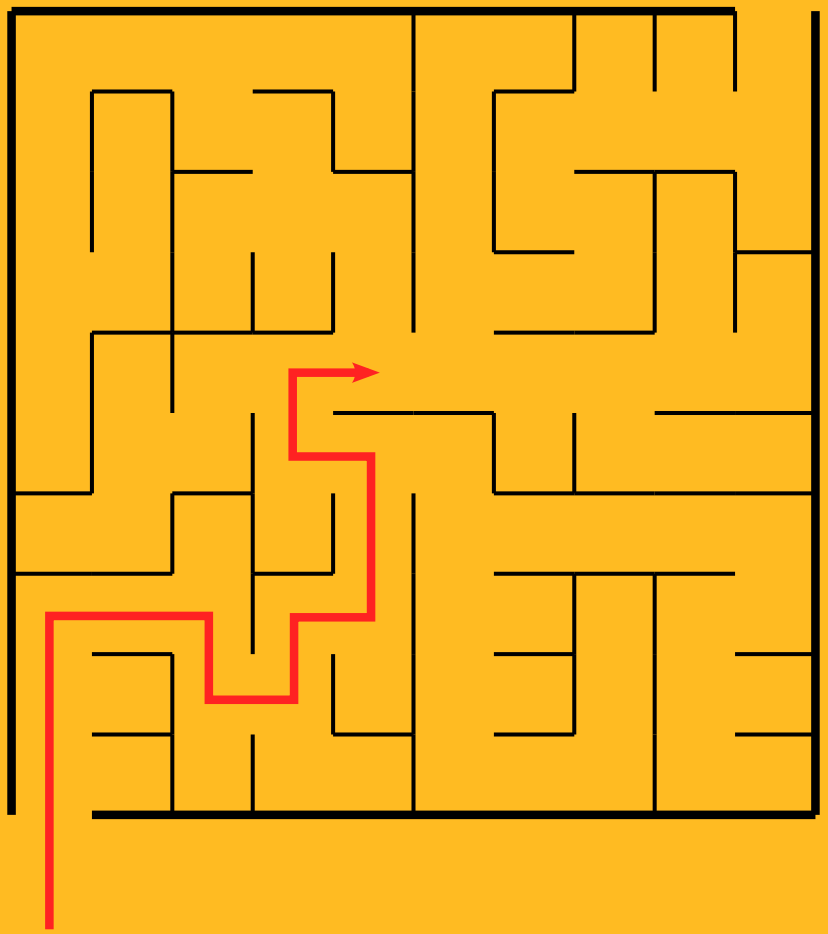
top of stack	current token					
	close	infix	prefix	postfix	circfix	postcir
empty	E	S	S	S	S	S
close	R	P	S	P	X	P
infix	R	P/A	S	P	S	P
prefix	R	P	S	P	S	P
postfix	R	R	X	R	S	S
circfix	S	S	S	S	S	S
postcir	S	S	S	S	S	S

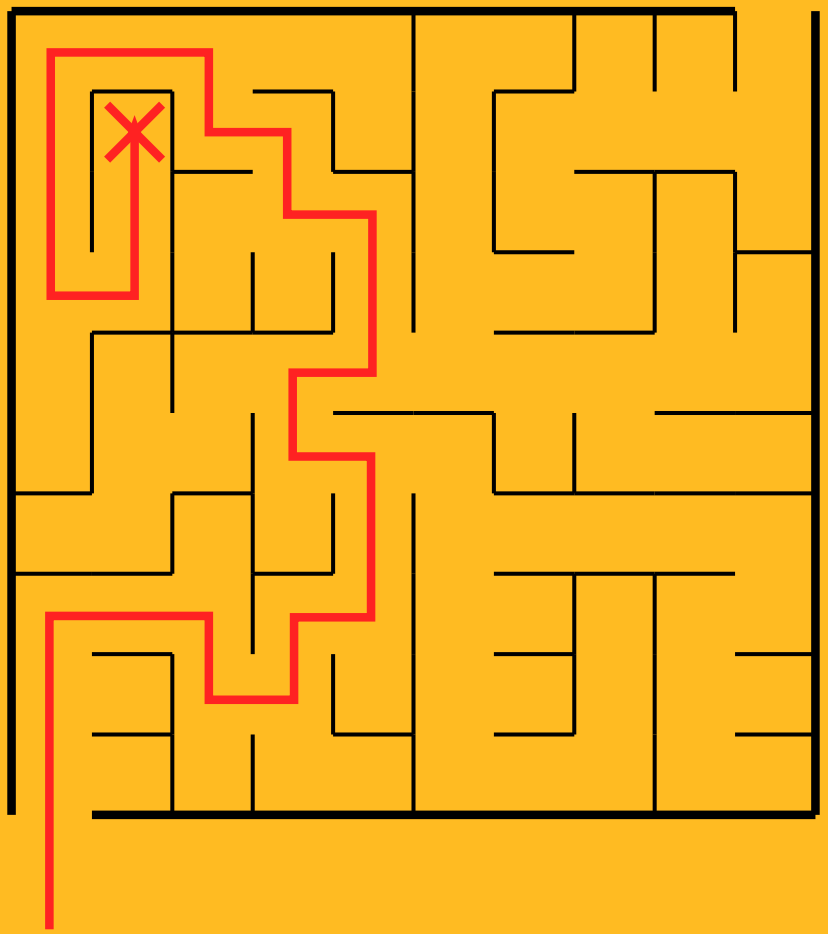
top-down: recursive descent

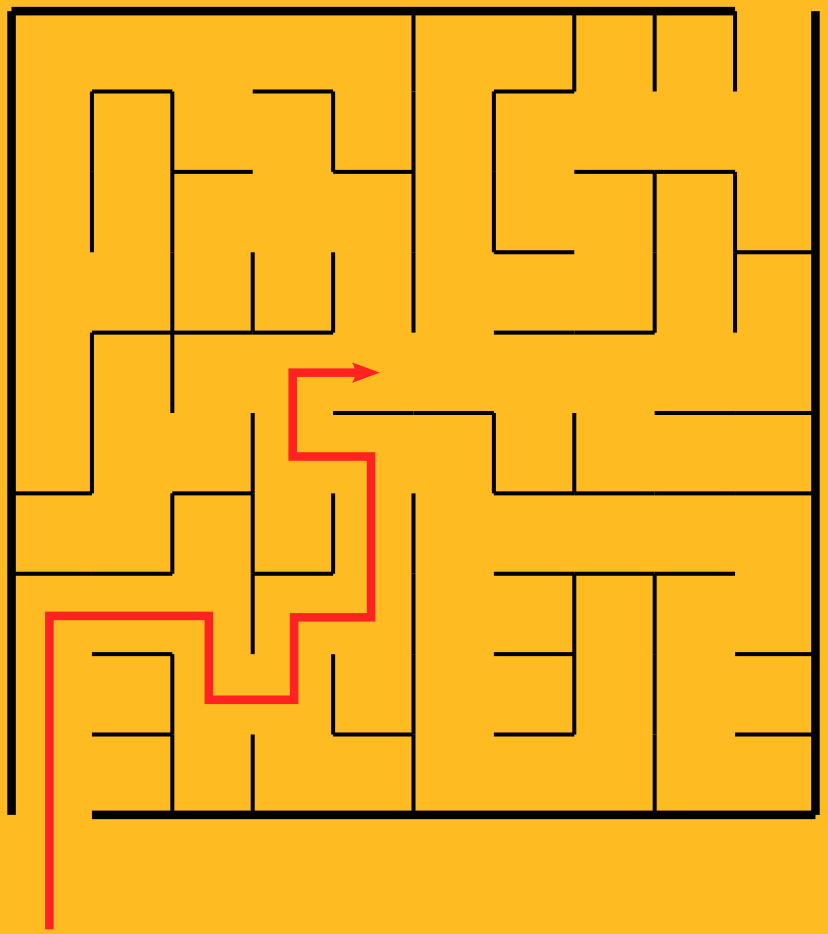
well suited for parsing whole programs

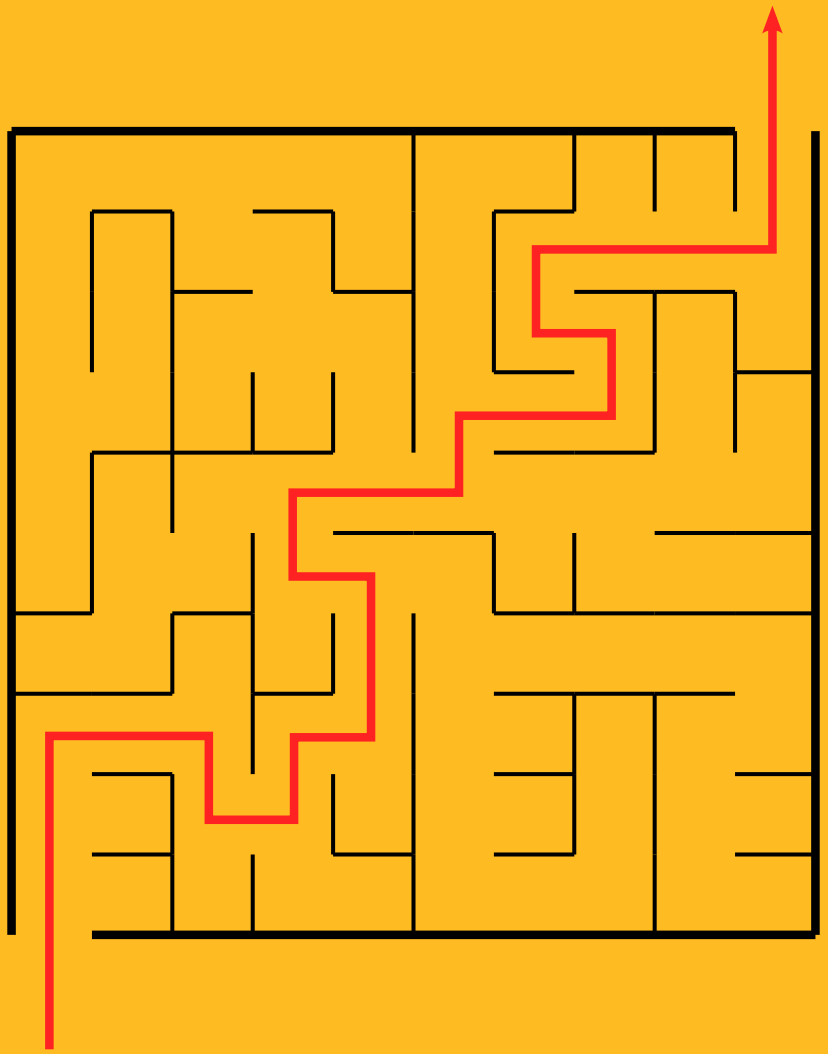
classes:methods::grammars:rules

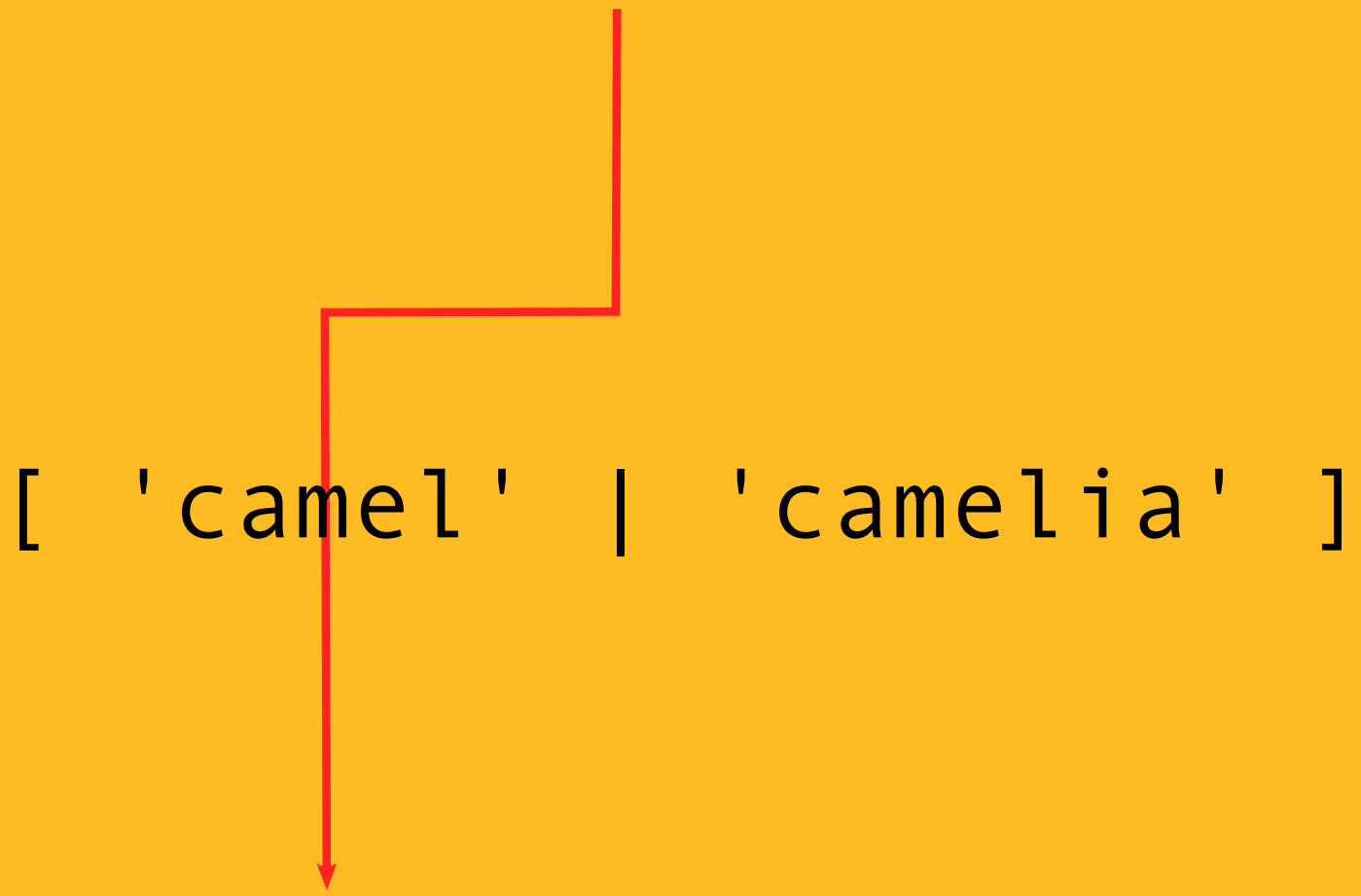
backtracking

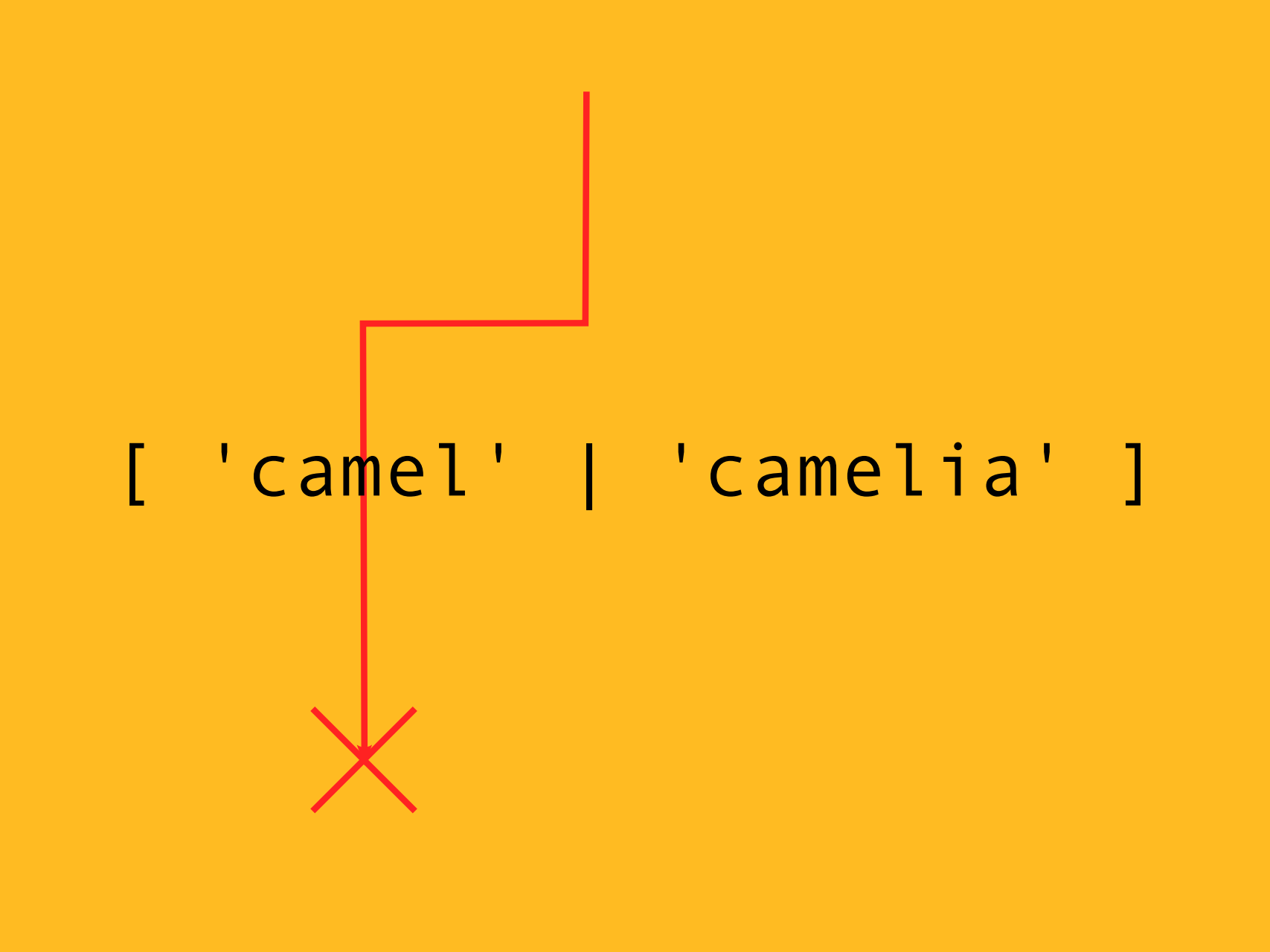


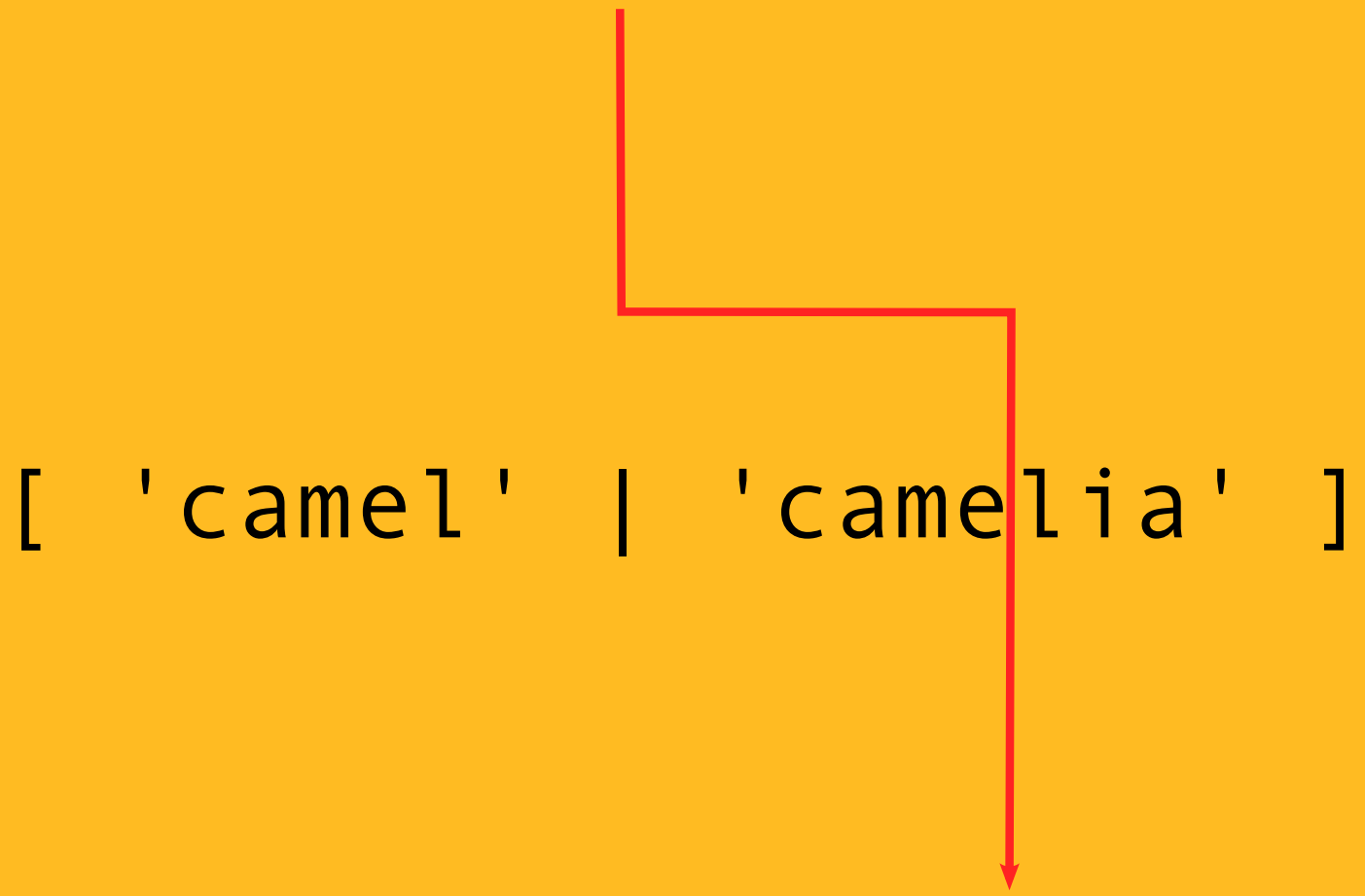


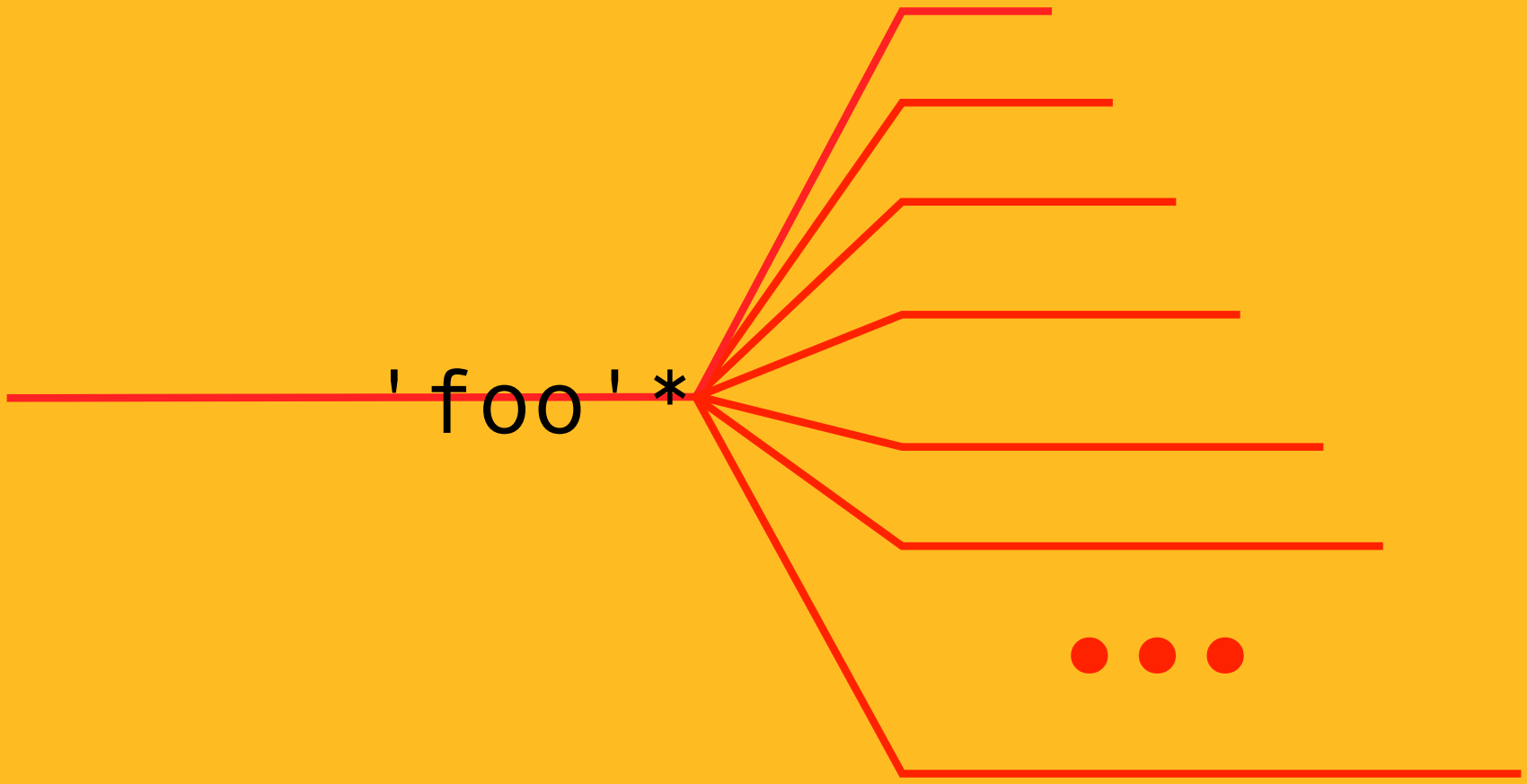





['camel' | 'camelia']


['camel' | 'camelia']


['camel' | 'camelia']

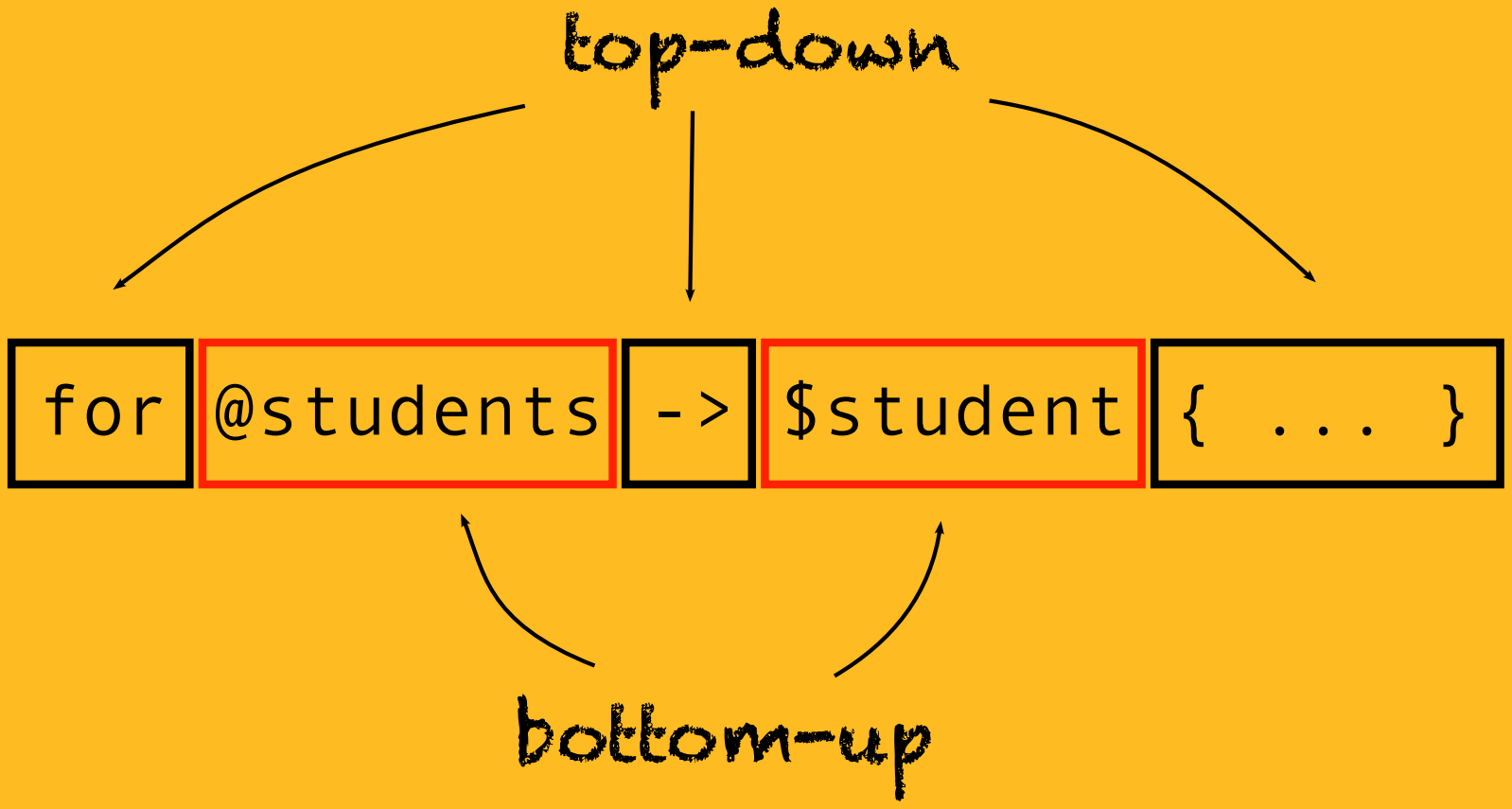

'foo'*

weaving it together

top-down parser for the program

bottom-up parser for the expressions

talking to each other


top-down
for
@students
->
$student
{ ... }
bottom-up

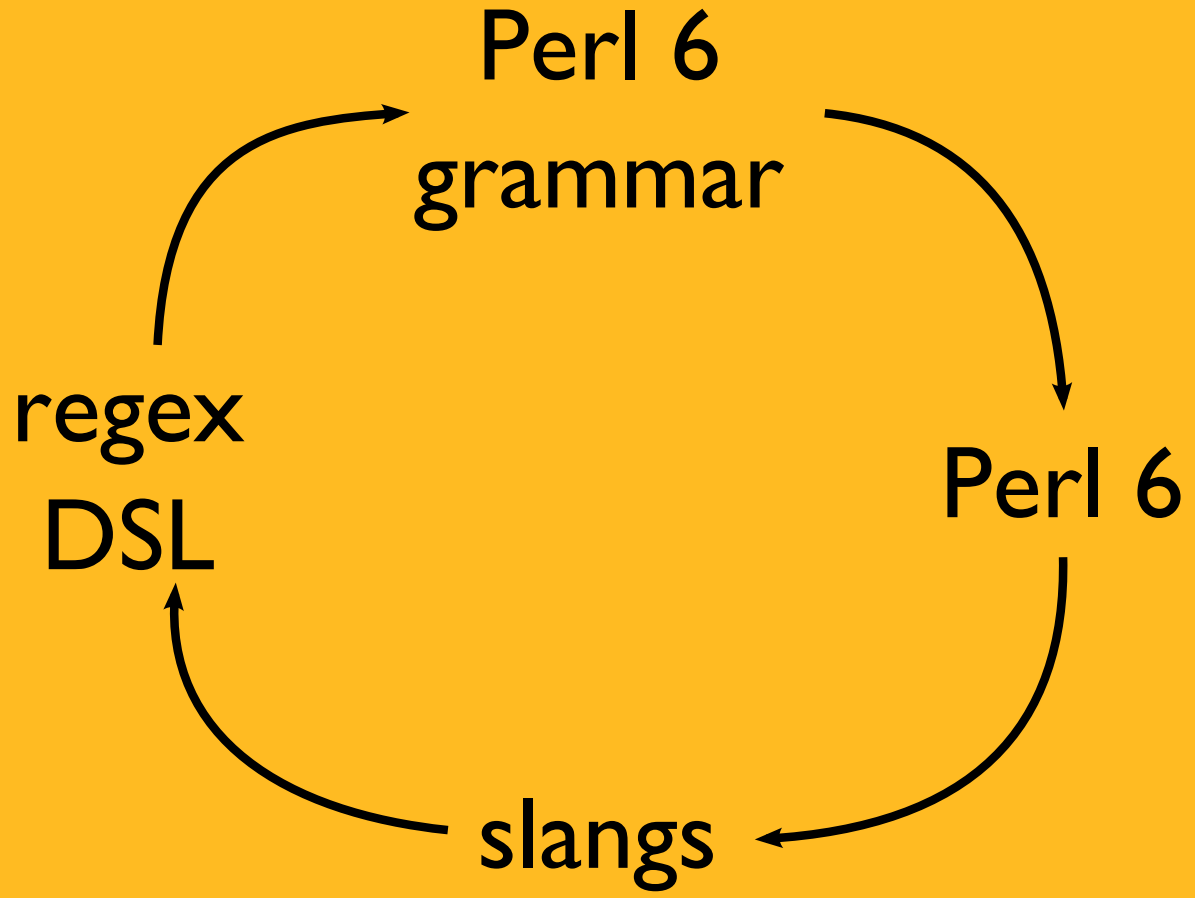
Perl 6 regexes and grammars

a language for talking about parsing

a DSL

(a slang)

(with quite good communication)


Perl 6
grammar
Perl 6
slangs
regex
DSL

```
token binint {
    <[ 0..1 ]>+ [ _ <[ 0..1 ]>+ ]*
}

token octint {
    <[ 0..7 ]>+ [ _ <[ 0..7 ]>+ ]*
}

token hexint {
    <[ 0..9 a..f A..F ]>+ [ _ <[ 0..9 a..f A..F ]>+ ]*
}

token decint {
    \d+ [ _ \d+ ]*
}

token integer {
    [
    | 0 [ b <binint>
        | o <octint>
        | x <hexint>
        | d <decint>
        | <decint>
            <!!{ $¢.worry("Leading 0 does not indicate octal in Perl 6") }>
        ]
    | <decint>
    ]
    <!!before ['.' <?before \s | ',' | '=' | <terminator> >
       <.panic: "Decimal point must be followed by digit">]? >
}
```

knowing-yourself in Perl 6

creating a class

introducing an operator

factorial example

```
sub postfix:<!>($n) {
    [*] 1 .. $n;
}

# later...
say 5!;  # 120
```

creating a macro

```
macro my-if($/) is parsed(
    rule {
        <cond=EXPR>  <iftrue=block>
        ['else'      <iffalse=block>]?
    }
) {
    quasi {
        {{{$<cond>.ast}}}
          ?? {{{$<iftrue>.ast}}}
          !! {{{$<iffalse>.ast}}}
    }
}

my-if $ticket === $winner {
    say 'You won the lottery! OMG!';
}
else {
    say 'Hm, no prize today...';
}
```

```
macro whenn't($/) is parsed(
    rule {
        <cond=EXPR> <ifwhenn't=block>
    }
) {
    quasi {
        if $_ !~ {{{$<cond>.ast}}} {
            {{{$<ifwhenn't>.ast}}}
        }
    }
}

given $celsius {
    when    * > 30 { warn 'HOT!'         }
    whenn't * < 0  { warn 'not freezing' }
    default        { warn 'freezing'     }
}
```

```
constant $DEBUG = True;

macro debug($ast) {
    quasi {
        if $DEBUG {
            warn({{{$ast}}})
        }
    }
}

debug( expensive_but_unnecessary_computation() );
```

creating a quote environment

creating a DSL

changing the MAIN slang

the Holy Grail:

manipulating on the tree level

(from inside the programming language)

manipulating strings: very impure

C-style macros

source filters

...just say no

OTOH, fully annotated trees

oh yes!

<autrijus> oh, there's another option
<autrijus> port PGE to Perl 6 :)
<autrijus> and have it generate perl 6 :)
<autrijus> (which is, the Ultimate Goal(tm))

wild plans for the future

macro preprocessor

static code analyzer

bootstrapping runtime

code coverage tool

time-traveling debugger

Attribution

Money Grab 2	http://www.flickr.com/photos/sgw/3435269871
Some hurry, some take their time	http://www.flickr.com/photos/prosto/4098119707
Not usual Ferris wheel	http://www.flickr.com/photos/hmr/79014848
Wall Street	http://www.flickr.com/photos/michaelaston/388570915
the smile of a man with a wild fan base	http://www.flickr.com/photos/notsogoodphotography/503637906
zen goddess reflecting on the granite table	http://www.flickr.com/photos/seandreilinger/133298903
HAL 9000	http://www.flickr.com/photos/litmuse/63615080
Structure and Interpretation of Computer Programs	http://mitpress.mit.edu/sicp/full-text/book/book.html
It's turtles all the way down	http://www.flickr.com/photos/goldtoe/282612809
Xzibit © Retna	http://www.superiorpics.com/event/Gridiron_Gang_Film_Promotion/index3.html
Freemacs/MINT © Russ Nelson	http://www.ibiblio.org/pub/micro/pc-stuff/freedos/files/edit/emacs/
The Observer	http://www.flickr.com/photos/chilledsalad/2094698476

kthxbai

questions?