6.009: Fundamentals of Programming

Week 10 Lecture: More Fun with Functions

Adam Hartz
hz@mit.edu

26 April 2021

6.009 Labs 9-10

In labs 9-10, we're implementing an interpreter for a dialect of LISP.

Lab 9: basics of evaluation (including function calls)
Lab 10: conditionals, lists, and other nice things

LISP

```
def fib(n):
    if n <= 1:
        return n
    return fib(n-1) + fib(n-2)

fib(20)
```

```
(:= fib
  (function (n)
    (if (<= n 1)
      n
      (+ (fib (- n 1)) (fib (- n 2))))))

(fib 20)
```

Why Bother Writing Interpreters?

- It is *just so cool!!!*
- It can help you understand the semantics of languages you already know (and contrast differing semantics).
- There is something powerful about the idea that an interpreter (CPython, for example) is *just another program*.

Why LISP?

- LISP is weird/cool :)
  - *"A language that doesn't affect the way you think about programming, is not worth knowing"* -Alan Perlis

- MIT and LISP have a long history
  - invented here in 1958 (McCarthy)
  - one widely-used dialect (Scheme) implemented here as well, used in 6.001 from ∼1980-2007

- Generally has very minimal syntax, so we can spend less time thinking about tokenizing/parsing, and more time thinking about rules for evaluation.

Functions

A key feature of our little LISP (and of Python) is the ability to define *functions* to abstract away the details of a particular computation.

Two Pieces: Function **Definition** and Function **Application**

Example:

```
def foo(x):
    return x+7

print(foo(3))
```

```
(:= (foo x) (+ x 7))

(foo 3)
```

Another Example

```
def deriv(f, dx):
    return lambda x: (f(x+dx) - f(x-dx)) / (2 * dx)
```

Another Example: in LISP

```
(:= (deriv f dx)
    (function (x) (/ (- (f (+ x dx)) (f (- x dx)))
                     (* 2 dx))))

(:= (nth-deriv f n dx)
    (if (=? n 0)
        f
        (deriv (nth-deriv f (- n 1) dx) dx)
    )
)
```

Functions are Awesome!

Functions are really powerful; we can implement many other language features using them.

For example, our LISP does not have looping keywords like `for` or `while`. Does this mean that we can't write code that involves loops?