I learned it last night! Everything is so simple! Hello world is just print("Hello, world!")

I dunno... dynamic typing? whitespace?

Come join us! Programming is fun again! It's a whole new world up here! But how are you flying?

I just typed import antigravity

That's it?...

I also sampled everything in the medicine cabinet for comparison. But I think this is the Python.

Python 3
Take-home midterm #2

**Available:** this Sunday night (10/28)

**Must return by:** the following Sunday (11/4) at 5pm

**Time-limit:** one sitting (with small breaks)

**Covers:** from the end of Hmmm to before break
functional programming, higher-order functions, recursion,
analysis (recurrence and Big O), use-it-or-lose-it, trees

**Resources:** one, 8½ x 11 sheet of notes (double-sided)

**Honor code:** don’t discuss exam questions
What’s still unclear about FP?

In the larger space, write down any lingering questions you have about functional programming, recursion, “use it or lose it”, analysis (e.g., recurrence relations), Racket, etc.

Use the “lingering questions” portion for questions about today’s material.

Firstname Lastname

(Your response)

Th. 10 / 25
There’s so much more to Racket

```rkt
#lang web-server/insta
(define (start request)
  (response/xexpr
    '(html
      (head)
      (body
        (h1 "Behold, the platypus!")
        (img ((src "http://bit.ly/2dpeUzo")))))))
```

racket-lang.org  clojure.org
Racket is syntactic sugar for the λ-calculus.
\[ x \in \text{Identifiers} \]

\[ \text{expr ::= x} \]

\[ \mid (\text{lambda} \ (x) \ \text{expr}) \]

\[ \mid (\text{expr} \ \text{expr}) \]
Preface:
What follows will never be on a CS 42 assignment or exam.
(define (\_\_\_\_\_\_ x)
  (if x false true))

(define (\_\_\_\_\_\_ x y)
  (if x y false))

(define (\_\_\_\_\_\_ x y)
  (if x true y))
(define (NOT x)
  (if x false true))

(define (AND x y)
  (if x y false))

(define (OR  x y)
  (if x true y))
(define NOT (λ (x) 
  (if x false true)))

(define AND (λ (x y) 
  (if x y false)))

(define OR (λ (x y) 
  (if x true y)))
(define NOT (λ (x)
          (if x FALSE TRUE)))

(define AND (λ (x y)
            (if x y FALSE)))

(define OR (λ (x y)
            (if x TRUE y)))

(define TRUE (λ (x y) x))
(define FALSE (λ (x y) y))
(define NOT (\(x\) (x FALSE TRUE)))

(define AND (\(x\ y\) (x y FALSE)))

(define OR (\(x\ y\) (x TRUE y)))

(define TRUE (\(x\ y\) x))

(define FALSE (\(x\ y\) y))
Prior experience: programming languages

- None
- Assembly
- Racket
- Python
- Java

Later
Soon

None | Lots
repl.it/languages/python3

slido.com (event code: Z314)
The essence of Python

Everything is an object.*

Every object has

a value.

a type.

an identity.

a namespace.

* but some objects (e.g., numeric and boolean literals) are “special”.

```python
>>> 1  # value
1
>>> type(1)  # type
<type 'int'>
>>> id(1)  # identity
140686900921016
>>> dir(1)  # namespace
['__abs__', ..., 'real']
```