Discussion 11

Programs as Data

Aditya Balasubramanian

`aditbala [at] berkeley [dot] edu`

Slides available at `teaching.aditbala.com`

Announcements

- Lecture 32 Youtube videos do not contain all examples: watch live lecture or the recording.
- Homework 8 due next Thursday 11/17 (extended).
- Scheme project checkpoint 2 due Sunday 11/13.
- Project parties: 5-7:30 Wednesday, 5-7:30 Thursday in Warren. Thursday 12-5 office hours are online only: oh.cs61a.org
- Try to finish Checkpoint 2 by TODAY (no weekend office hours)!
- Scheme is due Tuesday 11/22.
  - 1 bonus point for submitting by Monday 11/21.
  - 2 bonus points are automatically granted if you submit the project by Tuesday 11/22.
  - 1 bonus point for the extra credit question.

Programs as Data

- So far we've seen expressions like these...

```
>>> print(print(5))
5
None
>>> (lambda f, a: f(a))(lambda x: x * x, 3)
9
```

- But what if we wanted to do this?

```
[<expr> for i in range(5)]
```

- execute `<expr>` five times

Some Problems

```
def list_5(expr):
        return [expr for i in range(5)]

>>> lst = list_5(print(10))
10
>>> lst
[None, None, None, None, None]
```

- What's happening?
  - Python is evaluating the operands before executing the function, so `expr` is bound to `None` when executing the list comp

A Solution

- delay evaluation by passing in `expr` as a String

```
def list_5(expr):
        return f"[{expr} for i  in range(5)]"

>>> list_5("print(10)")
"[print(10) for i in range(5))]"
>>> lst = eval(list_5("print(10)"))
10
10
10
10
10
>>> lst
[None, None, None, None, None]
```

Scheme Programs as Data

- Scheme stores all non-primitive expressions to store data
    - can use lists to represent programs as data rather than Strings

```
scm> (define expr (list '+ 2 2))
expr
scm> expr
(+ 2 2)
scm> (eval expr)
4
```

Quasiquotation

- functions very similarly to quote, but we can also `unquote` or `,` to evaluate SOME expressions in a quoted list

```
scm> (define a 5)
a
scm> (define b 3)
b
scm> `(* a b)
(* a b)
scm> `(* a ,b)
(* a 3)
scm> '(* a ,b)
(* a (unquote b))
```

Thank you!!!

Anon Feedback -> https://tinyurl.com/adit-anon