Conclusion

August 12th, 2020

Wrap-Up Announcements

Announcements

- The final is tomorrow from 5 - 8 PM PDT. The exam was emailed.
  - More exam info is on the course website and Piazza.
- All grades except for the final will be shortly on OKPY after the lecture.
  - See howamidoing to see your progress in the class and what you need on the final to get the grade you want.
  - If your howamidoing and OKPY disagrees, OKPY is correct
  - **Regrade requests for all assignments are due Friday at 11:59 PM.**
- We'll work to grade your final as soon as we can and should get grades back to you Sunday or Monday.
  - Because of campus deadlines for submitting grades, we'll only allow final exam regrades for 24 hours after we release them.
- Fill out the Scheme Art Contest Votes + Department Course Evaluation!
  - If >= 85% of the class fills both surveys out, everyone will receive 1 EC point on the final

Scheme Contest gallery

What is CS 61A? *Revisited*

What is CS 61A?

- A course about **managing complexity**
  - Mastering functional abstraction and data abstraction
  - Learning trade-offs of various programming paradigms
- An introduction to **programming**
  - Learning how to write programs in Python, Scheme, and SQL
  - Combining multiple programming techniques in large projects
  - Understanding how computers interpret programming languages
- A **challenging course** that demanded a lot of you

“Building Blocks” of Programs

Primitive Expressions: 2 “hello!” add

numbers strings names

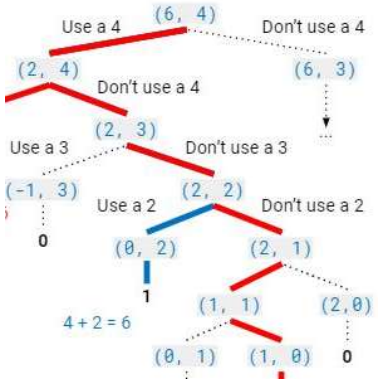
Arithmetic Expressions: `1 + 2` `15 // 3`

Call Expressions:

```
add(3, 4)
max(add(2, 3), 5 * min(-1, 4))
```

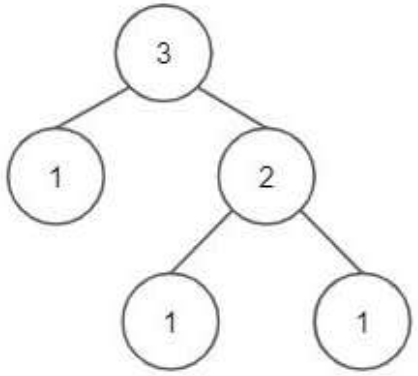
```
f4: a_plus_bc [parent=Global]
          a   4
          b   3
          c   81
         bc   243
Return value  247
```

```
i, total = 0, 0
while i < 3:
    i = i + 1
    total = total + i
```



Representing Collections of Data

```
i = 0
for elem in [8, 9, 10]:
    print(i, ":", elem)
    i += 1
```



`sum(iterable[, start]) -> value`

- Return the sum of an iterable of numbers (NOT strings) plus the value of parameter 'start' (which defaults to 0). When the iterable is empty, return start.
- *Note: cannot be called on strings instead must use built in string method .join*

`max(iterable[, key=func]) -> value` **OR** `max(a, b, c, ...[, key=func]) -> value`

- With a single iterable argument, return its largest item.
- With two or more arguments, return the largest argument.

Mutability

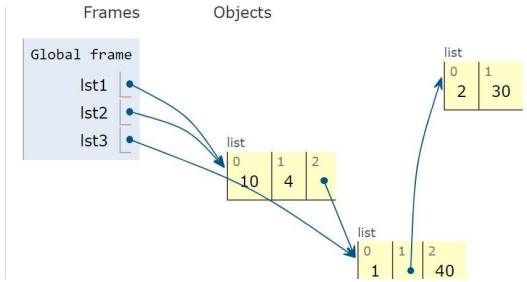
`x = 2`

Status	Effect
•No nonlocal statement •"x" **is not** bound locally	Create a new binding from name "x" to object 2 in the first frame of the current environment
•No nonlocal statement •"x" **is** bound locally	Re-bind name "x" to object 2 in the first frame of the current environment
•nonlocal x •"x" **is** bound in a non-local frame	Re-bind "x" to 2 in the first non-local frame of the current environment in which "x" is bound
•nonlocal x •"x" **is not** bound in a non-local frame	SyntaxError: no binding for nonlocal 'x' found
•nonlocal x •"x" **is** bound in a non-local frame •"x" also bound locally	SyntaxError: name 'x' is parameter and nonlocal

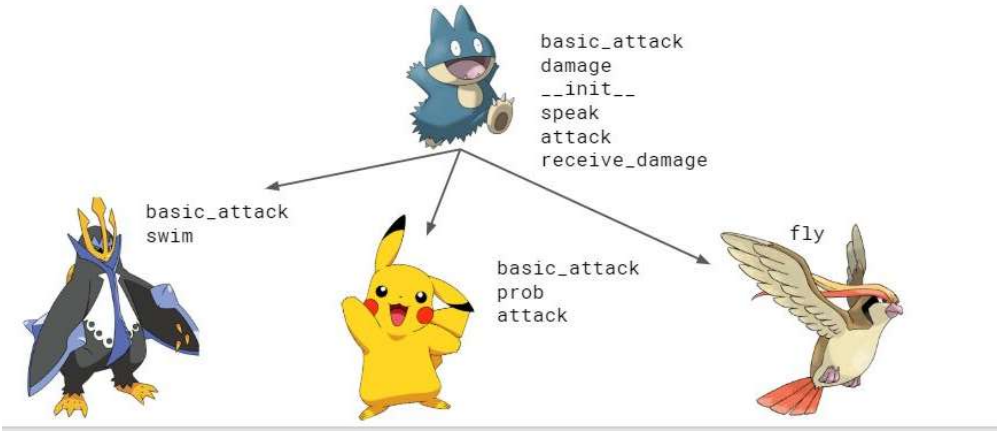
Python 3.6

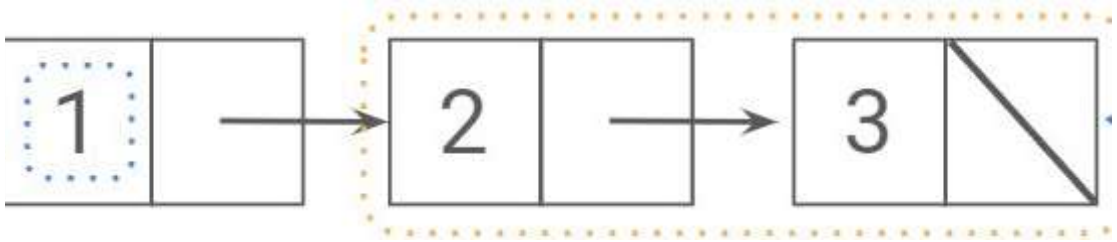
```
1  lst1 = [1, [2, 3], 4]
2  lst2 = lst1
3  lst3 = lst1[:]
4  lst1[0] = 10
5  lst3[2] = 40
6  lst2[1][1] = 30
7  lst2.pop(1)
8  lst1.append(lst3)
```

Edit this code

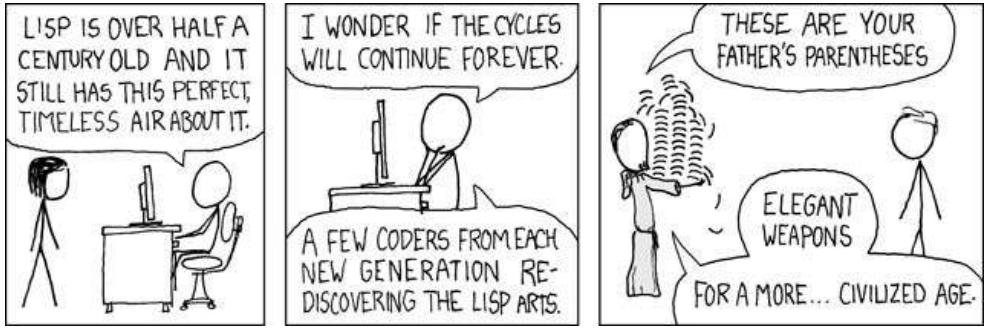


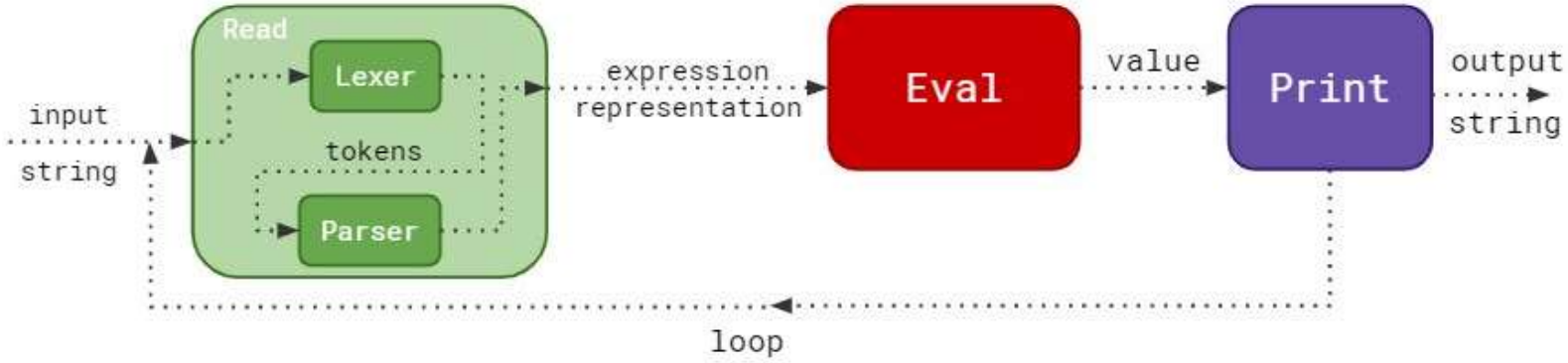
Object-Oriented Programming





Languages / Interpreters





Paradigms & Databases

```
CREATE TABLE parents AS
  SELECT "delano" AS parent,  "herbert" AS child UNION
  SELECT "abraham",           "barack"           UNION
  SELECT "abraham",           "clinton"          UNION
  SELECT "fillmore",          "abraham"          UNION
  SELECT "fillmore",          "delano"           UNION
  SELECT "fillmore",          "grover"           UNION
  SELECT "eisenhower",        "fillmore";
```

table **dogs**

name	fur	age
eisenhower	short	7
delano	long	10
grover	short	2
fillmore	curly	8
herbert	curly	4

delano	long	10
eisenhower	short	7
grover	short	2
fillmore	curly	8
herbert	curly	4

fur	avg_age
short	4.5
long	10
curly	6

Applications

What's Next?

The 61 Series

61A teaches **what** programming is - the techniques it involves, and how programs are executed

61B shows you **why** these techniques are important for programming - you'll apply them to even larger projects and use them to make your code even more efficient

61C shows you **how** programs are actually executed and the implications that has for writing code

Other classes

CS 70 - Discrete Mathematics and Probability Theory

Data 8 - Foundations of Data Science

Upper division classes:

- CS 170 - Efficient Algorithms and Intractable Problems, requires CS 61B & CS 70
- CS 188 - Introduction to Artificial Intelligence, requires CS 61B & CS 70
- CS 186 - Introduction to Database Systems, requires CS 61B (and CS 61C)
- CS 161 - Computer Security, requires CS 61C and CS 70
- And many more!

Teaching

- Become a Computer Science Mentor
  - Lead small-group mentoring sections
  - Help create the community you want to see
  - Inspire others to pursue computer science
- Learn teaching and one on one tutor through CS 370
  - 3 unit pedagogy course offered during the semester
  - Learn how to be a good teacher
  - One on one tutor students
- There will be no opportunity for academic interning next semester :(. But look out for the future semesters!

Parting Thoughts

Sleep, eat, get sunlight

Never be afraid to reach out to someone or ask for help

Look for intersections and go an abstraction level below what you are interested in

Coming out of this class, you know how to program

That puts you easily in the top 1% of programming skill among people who are currently alive

You can (maybe even should) take pride in this

But

don't become an elitist

College is not just about academics.

Take time in college to explore what you really want your life to be about


Thank you!

Q & A

Ask the staff anything (non-technical) you want!