Discussion 4

Tree Recursion + Python Lists

Antonio Kam

anto [at] berkeley [dot] edu

Announcements

- Cats got released 🐈
  - Be on the lookout for question 7 - start early, and start often
  - Getting started videos
- No HW released this week! 🥳

Comments from last section

- Do you like Cheese?
  - big yes
- How to get better at reading python output
  - good question tbh
  - error messages
  - debugging page
- Address more environment diagrams
  - for this point in the course (recursion), environment diagrams start being a bit worse
  - they'll come back eventually

Comments from last section

- Do you watch anime? If so, what's your favorite (or the one you like the most right now)?
  - i only watch the pokemon anime, and only do this occasionally (however it do be kinda hype)
- How many hours of sleep do you get per night? Are you a night owl or morning lark? Hope your classes are going well!
  - 7-8; prefer night, but i shift between both depending on what i have to do that day; classes are not going well 💀
- If you could upload lab notes on to your website that would be great!
  - they're on the website! 🥳

Temperature Check 🌡️

- Recursion
- Tree Recursion
- `for` loops
- Lists
  - List Comprehensions
- Dictionaries

All slides can be found on

`teaching.rouxl.es`

Tree Recursion 

Tree Recursion

- Tree recursion is recursion but with two (or more!) recursive calls
- Useful when you need to break down a problem in more than 1 way
- Useful when there are multiple choices to deal with at one function call
- The recursive call diagram will expand similar to the roots of a tree

Example 1: Recursive Fibonacci

```
def fib(n):
  if n == 0:
    return 0
  elif n == 1:
    return 1
  else:
    return fib(n - 1) + fib(n - 2)
```

- Notice how this still follows the rules of recursion
  - We have base case(s)
  - We reduce our problem (`fib(n - 1)` and `fib(n - 2)`)
  - We connect it together (with `+`)
- Often you combine things with `+`, `-`, `*`, `/` or some other function (`max`, `min`, etc).

Example 1: Recursive Fibonacci

You can also write down

```
def fib(n):
  if n == 0 or n == 1:
    return n
  else:
    return fib(n - 1) + fib(n - 2)
```

Worksheet!

Lists

- So far, we've only really been able to store one piece of data at a time
- A list allows you to store multiple pieces of data in 1 variable
- Lists can store any data type, and can be a *mix* of different data types
  - For example: `[1, "hello", [2, "hi"]]`
- Very useful for storing data/information

Creating Lists

In general, to create a list, you wrap something in square brackets

- For example: `[1, 2, 3]` creates a list.

(List) Slicing

Syntax is `lst[<start index>:<end index>:<step>]`; this creates a *copy* of all or part of your list (will be important later)

- `start index` is inclusive (if not included, it defaults to the first value)
- `end index` is exclusive (if not included, it defaults to the end of the list)
- `step` can be negative!

```
lst = [3, 4, 5]
lst[:] # Makes a copy; returns [3, 4, 5]
lst[1:] # [4, 5]
lst[::-1] # [5, 4, 3]
lst[::2] # [3, 5]
```

For loops

```
for <variable> in <sequence>:
  [body of for loop]
```

Example:

```
lst = [3, 4, 5]
for elem in lst:
  print(elem)
```

List Comprehensions

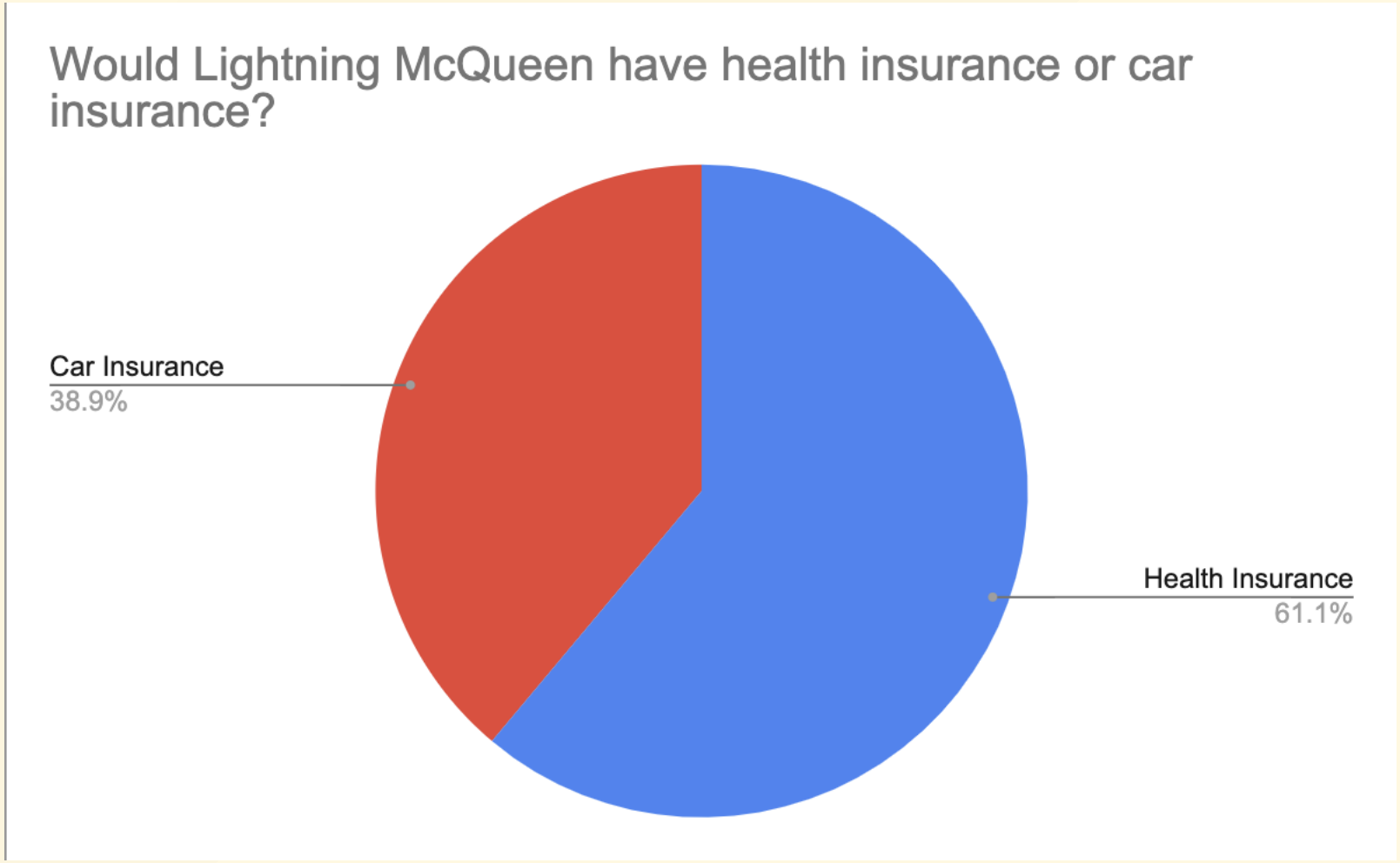
Very similar to `for` loops, but can be done in 1 line!

```
[<expression> for <variable> in <sequence> [if <condition>]]

lst = [3, 2, 1]
[x * 2 for x in lst if x < 3] # [4, 2]
[x * 2 for x in [3, 2, 1]] # [6, 4, 2]
```

Worksheet!

Results from last section (`links.rouxl.es/disc`)



Dictionaries 📖

- Maps `keys` to `values`
  - `keys` must be 'hashable'
    - you can just think of keys needing to be immutable, but this is not too important
- `values` can be anything (including other dictionaries)
- Doesn't really have an order
- Access elements using `keys` rather than indices

Dictionaries 📖

- Maps `keys` to `values`
- Doesn't really have an order
- Access elements using `keys` rather than indices
- Defined with curly braces (`{}`)
  - `{key: value}`

Demo:

```
pokemon = {'pikachu': 25, 'dragonair': 148, 25: 'hello'}
pokemon['pikachu'] # 25
pokemon['hello'] = 'hi'
pokemon # {'pikachu': 25, 'dragonair': 148, 25: 'hello', 'hello': 'hi'}
```

Worksheet!

Mental Health Resources

- CAPS:
  - If you need to talk to a professional, please call CAPS at 510-642-9494.
- After Hours Assistance
  - For any assistance after hours, details on what to do can be found at this link

Anonymous Feedback Form

`links.rouxl.es/feedback`

Thanks for coming! 🥳

Please give me feedback on what to improve!