DS2500
4/31 - Fri :)

Admin

- HW1 is graded!
  - any Qs on your grade until weds 9pm
    - clarity • mistake?
- HW2 due 9pm
- HW3 out, due 2/7 9pm
- Optional XC during Finals
  - 8-10AM 4/18, 4/23, 4/24
    - present your semester project
    - sign ups open after exam 2
    - in person or online
    - 0 - 3 points
- Lab 3 monday
- Lab 1-3 late deadline 2/7 9pm
- Exam #1 2/14

Agenda

1. Reading/writing unit tests
2. writing + testing new/existing code
3. Python

0. Comprehension
   - last time: debugging starter code
     resolving error msgs as they come up
     Python threw an error in lst_to_dct
     - error msg 😵 unhashable type: List
       - read the function doc
       - fixed the argument to function

List/dictionary comprehensions
- shortcuts
- never the only way

ex long way (from tvcs code)

```
for i in range(len(ages)):
    #print(f"Ages in loop became an int: {ages}")
    ages[i] = int(ages[i])
```

Shortcut:

ages = [int(age) for age in ages]

---

ex long way

```
dct = {}
for i in range(3):
    dct[i] = i+1
```

shortcut

dct = {i:i+1 for i in range(3)}

They both create dct = {0:1, 1:2, 2:3}

---

ex lst = [['a', 'b', 'c'],
[1, 3, 5],
[2, 4, 6]]

```
dct = {h : [] for h in lst[0]}
for row in lst[1:]:
    for i in range(len(row)):
        dct[lst[0][i]].append(row[i])
return dct
```

what is in dct?

- lst[0] ['a', 'b', 'c']

  dct = {'a': [], 'b': [], 'c': []}

- row [1, 3, 5]

  i = 0

  dct[lst[0][i]].append(row[i])

  a

  dct = {'a': [1], 'b': [3], 'c': [5]}

- row [2, 4, 6]

  → dct = {'a': [1, 2], 'b': [3, 4], 'c': [5, 6]}

turns 2D list into dictionary
row[0] is keys
rest of col becomes value

1. Reading / Writing Unit Tests

- does our code work? (no errors)
- does our code work as expected? (correct for the job)

unit test:
- verify correctness of indv functions
- primary focus: does our code work as expected?
- confidence that whole program works if indv functions are correct independently

- example arguments
- expected results

- call the function
- compare expected vs actual

unit tests for tves code
- filter_age
  - 46 (int)
  - [47, 48] (ints)
  - ['x', 'y'] (strings)

} example inputs

expected output: ['x', 'y']

Best practice: at least 3 tests per function
- returns everything in list
- returns some things in list
- returns empty list

need to:
- install pytest
- create a new file: test_~~~.py
- write one function per function to test
- call the function
- assert that expected and actual results match

```
def test_filter_age():
    - test 1
    - test 2
    - test 3
    assert [call func] == [expected]
```