- Questions
- Next week
  - Monday: course evals, AMA, review
    - **homework 7 due 9pm, those with remaining late days can use them**
  - Wednesday: greatest hits quiz, wrap up, project work time
    - **quiz on concepts I want you to have down**
      - writing a function
      - list indexing
      - for loops over elements and indexes
      - while loops
      - what return does
      - if, elif, else
      - and, or
      - when to use a dictionary or a list
    - **quiz reflections and peer evaluation due**
  - By 9pm Friday, each project group sends me two Google Presentation slides
  - Saturday, 8:30am, lightning presentations in CMC 102, project demos
  - Project code and documentation submitted via Moodle by 5pm Monday, Nov. 25 (hard deadline)
- map
  - take a function and apply it to each element of a sequence

- ```
  # apply abs (absolute value) to each element of a list
  nums = xs.copy() # xs is a list of numbers
  for i in range(len(nums)):
      nums[i] = abs(nums[i])
  ```
- becomes `nums = list(map(abs, xs))`
- recall how we processed election data in homework 3
  - ```
    fp = open("district_overall_2018.csv")
    fields = fp.readline()
    lines = fp.readlines()
    fp.close()

    for i in range(len(lines)):
        lines[i] = lines[i].strip().split(",")
    ```
  - ```
    fp = open("district_overall_2018.csv")
    fields = fp.readline()
    lines = fp.readlines()
    fp.close()

    def process_line(line):
        return line.strip().split(",")

    lines = list(map(process_line, lines))
    ```
- practice: rewrite this code using map
  - ```
    inputs = ["Meep", "MORP", "mergle"]
    inputs_lower = inputs.copy()
    for i in range(len(inputs_lower)):
        inputs_lower[i] = inputs_lower[i].lower()
    ```
  - ```
    inputs = ["Meep", "MORP", "mergle"]
    inputs_lower = list(map(str.lower, inputs))
    ```

filter
    ```

- given a predicate (function that returns True or False) and a sequence, apply the predicate to each element and generate a new sequence without elements that return False
- ```
  # take a list of numbers and produce a new list with only the positive numbers
  nums = []
  for x in xs: # xs is a list of numbers
      if x > 0:
          nums.append(x)
  ```
- becomes
  ```
  def positive(num):
      return num > 0
  nums = list(filter(positive, xs))
  ```
- again going back to homework 3, get the lines for Minnesota
  - ```
    mn_lines = []
    for line in lines:
        if line[1] == "MN":
            mn_lines.append(line)
    ```
  - ```
    def is_mn(line):
        return line[1] == "MN"
    mn_lines = list(filter(is_mn, lines))
    ```
- practice: count the number of words with length greater than 4 using filter and map instead of loops
  - ```
    lines = [['We', 'are', 'the', 'hollow', 'men'],
     ['We', 'are', 'the', 'stuffed', 'men'],
     ['Leaning', 'together'],
     ['Headpiece', 'filled', 'with', 'straw.', 'Alas!']]
    ```
  - ```
    count = 0
    for line in lines:
        for word in line:
            if len(word) > 4:
    ```
    ```

```
            count += 1
print(count)
```

- ```
  def more_than_4(word):
      return len(word) > 4
  def count_long_words(line):
      return len(list(filter(more_than_4, line)))
  print(sum(map(count_long_words, lines))
  ```

lambda

anonymous function

- **a function without a name we define just-in-time**
- `lines = list(map(lambda line: line.strip().split(","), lines))`
- `nums = list(filter(lambda x: x > 0, xs))`

practice: redo mn_lines example using a lambda

- `mn_lines = list(filter(lambda line: line[1] == "MN", lines))`
- `print(sum(map(lambda line: len(list(filter(lambda word: word > 4))), lines)))`

list comprehension

- a single expression that generates a new sequence from an existing sequence
- instead of `nums = list(map(abs, nums))`
  `nums = [abs(num) for num in nums]`

basically, [<resulting item> <start of a for loop>]

- **can extend with filtering**
- **[<resulting item> <start of a for loop> if <boolean expression>]**
  ```

- have list of temperature readings as strings
  - some are missing (""), some malfunctioned ("999")
- `temps = [float(t) for t in temps if len(t) > 0 and t != "999"]`
- get vote totals for MN district 2
  - `votes = [int(line[14]) for line in lines if line[1] == "MN" and line[7] == "District 2" and line[12] != "TRUE"]`