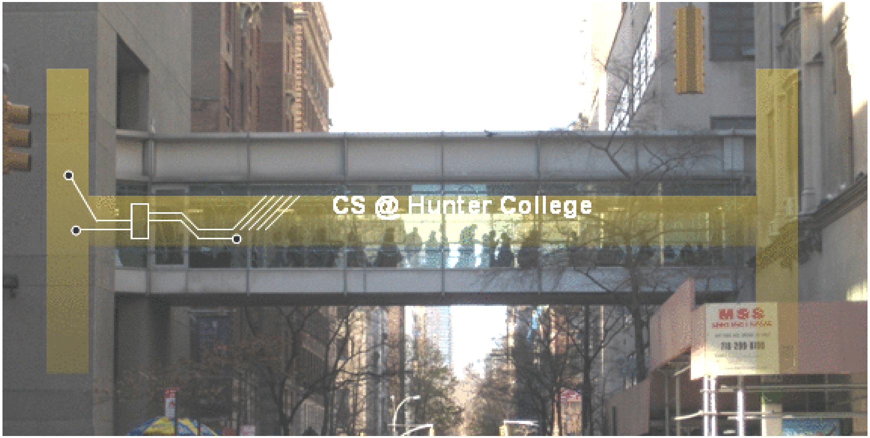
# CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.
- For those with a final at the same time (e.g. PHILO 101 & POLSC 100 students), we are arranging an alternative time.

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.
- For those with a final at the same time (e.g. PHILO 101 & POLSC 100 students), we are arranging an alternative time.
- Best way to prepare for final:

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.
- For those with a final at the same time (e.g. PHILO 101 & POLSC 100 students), we are arranging an alternative time.
- Best way to prepare for final:
  - Do the programming problems & labs.

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.
- For those with a final at the same time (e.g. PHILO 101 & POLSC 100 students), we are arranging an alternative time.
- Best way to prepare for final:
    - Do the programming problems & labs.
    - Do the sample final problems (in lecture & full sample exam come December).

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.
- For those with a final at the same time (e.g. PHILO 101 & POLSC 100 students), we are arranging an alternative time.
- Best way to prepare for final:
  - Do the programming problems & labs.
  - Do the sample final problems (in lecture & full sample exam come December).
  - Attend lecture & review the lecture notes.

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.
- For those with a final at the same time (e.g. PHILO 101 & POLSC 100 students), we are arranging an alternative time.
- Best way to prepare for final:
  - Do the programming problems & labs.
  - Do the sample final problems (in lecture & full sample exam come December).
  - Attend lecture & review the lecture notes.
  - Do the associated reading.

## Announcements

- Final will be Wednesday, 20 December, 9am to 11am.
- For those with a final at the same time (e.g. PHILO 101 & POLSC 100 students), we are arranging an alternative time.
- Best way to prepare for final:
  - Do the programming problems & labs.
  - Do the sample final problems (in lecture & full sample exam come December).
  - Attend lecture & review the lecture notes.
  - Do the associated reading.
- Today's lecturers include:
  - Genady Maryash (adjunct coordinator) &
  - Katherine Howitt (tutor coordinator).

## Today's Topics

- Folium Recap
- Indefinite Loops
- Searching Data
- Random Numbers

## folium

- A module for making HTML maps.

Folium

## folium

Folium

- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.

## folium

Folium

- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.
- Outputs `.html` files which you can open in a browser.

## folium

- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.
- Outputs `.html` files which you can open in a browser.
- An extra step:

## folium

- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.
- Outputs `.html` files which you can open in a browser.
- An extra step:

*Write code.* $\rightarrow$ *Run program.* $\rightarrow$ *Open .html in browser.*

## From Last Time: folium example

*What does this code do?*

```
import folium
import pandas as pd

cuny = pd.read_csv('cunyLocations.csv')
mapCUNY = folium.Map(location=[40.75, -74.125])

for index,row in cuny.iterrows():
    lat = row["Latitude"]
    lon = row["Longitude"]
    name = row["Campus"]
    if row["College or Institution Type"] == "Senior Colleges":
        collegeIcon = folium.Icon(color="purple")
    else:
        collegeIcon = folium.Icon(color="blue")
    newMarker = folium.Marker([lat, lon], popup=name, icon=collegeIcon)
    newMarker.add_to(mapCUNY)

mapCUNY.save(outfile='cunyLocationsSenior.html')
```

## From Last Time: folium example

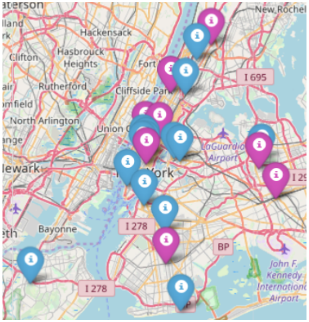
*What does this code do?*

```
import folium
import pandas as pd

cuny = pd.read_csv('cunyLocations.csv')
mapCUNY = folium.Map(location=[40.75, -74.125])

for index,row in cuny.iterrows():
    lat = row["Latitude"]
    lon = row["Longitude"]
    name = row["Campus"]
    if row["College or Institution Type"] == "Senior Colleges":
         collegeIcon = folium.Icon(color="purple")
    else:
         collegeIcon = folium.Icon(color="blue")
    newMarker = folium.Marker([lat, lon], popup=name, icon=collegeIcon)
    newMarker.add_to(mapCUNY)

mapCUNY.save(outfile='cunyLocationsSenior.html')
```

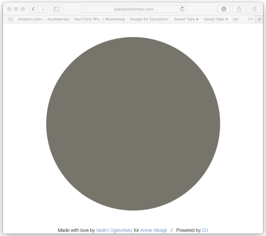
# From Last Time: folium example

*What does this code do?*

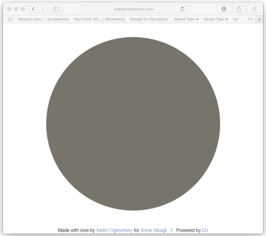
```python
import folium
import pandas as pd

cuny = pd.read_csv('cunyLocations.csv')
mapCUNY = folium.Map(location=[40.75, -74.125])

for index,row in cuny.iterrows():
    lat = row["Latitude"]
    lon = row["Longitude"]
    name = row["Campus"]
    if row["College or Institution Type"] == "Senior Colleges":
         collegeIcon = folium.Icon(color="purple")
    else:
         collegeIcon = folium.Icon(color="blue")
    newMarker = folium.Marker([lat, lon], popup=name, icon=collegeIcon)
    newMarker.add_to(mapCUNY)

mapCUNY.save(outfile='cunyLocationsSenior.html')
```

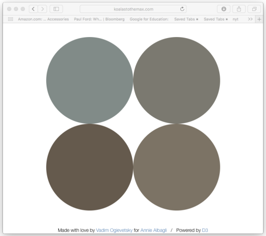


# From Last Time: koalas

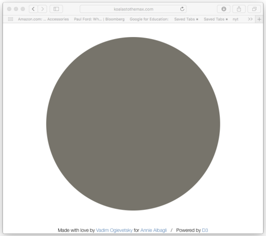

koalastothemax.com
Amazon.com: ... Accessories
Paul Ford: Wh... | Bloomberg
Google for Education:
Saved Tabs
Saved Tabs
nyt
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

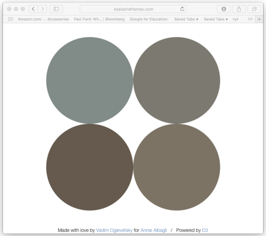
# From Last Time: koalas

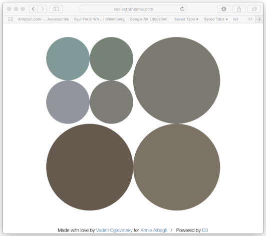

koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3


koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

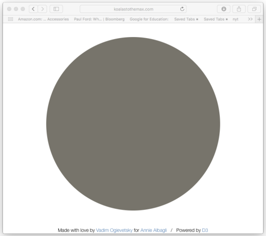
# From Last Time: koalas

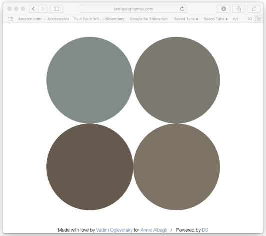

koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

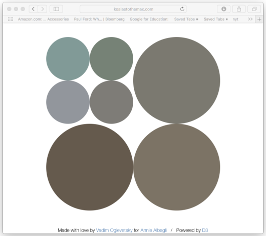

koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3


koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

## From Last Time: koalas

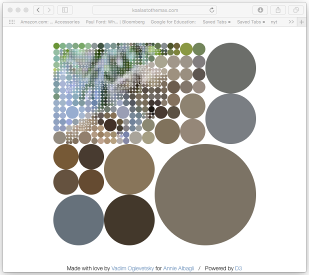

koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3


koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

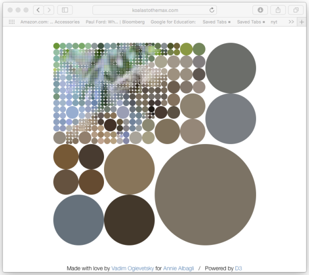

koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

`http://koalastothemax.com`

## From Last Time: koalas

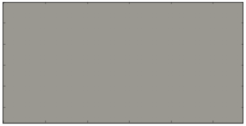

koalastothemax.com
Amazon.com: ... Accessories
Paul Ford: Wh... | Bloomberg
Google for Education:
Saved Tabs
Saved Tabs
nyt
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

## From Last Time: koalas

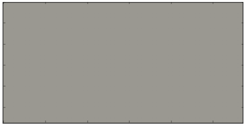

koalastothemax.com
Made with love by Vadim Ogievetsky for Annie Albagli / Powered by D3

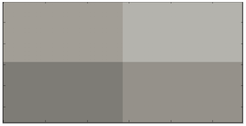
- Top-down design puzzle:
  - What does `koalastomax` do?
  - What does each circle represent?
- Write a high-level design for it.
- Translate into a `main()` with function calls.

# From Last Time: koalas

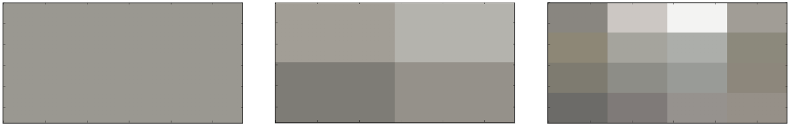


# From Last Time: koalas

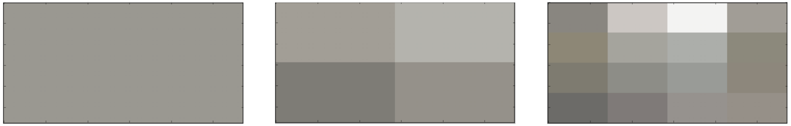




## From Last Time: koalas

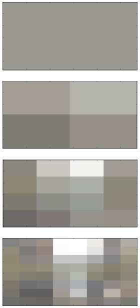


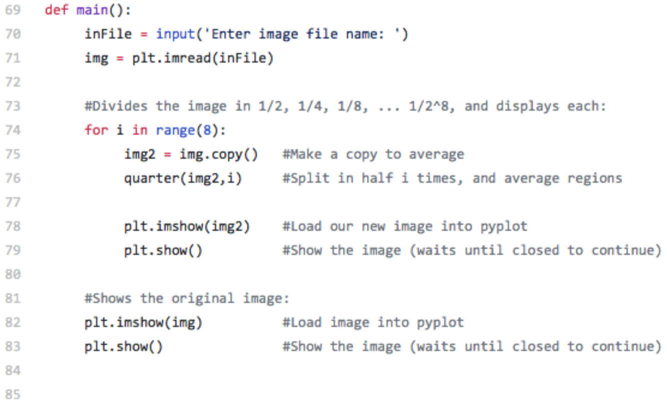
## From Last Time: koalas



- Top-down design puzzle:
  - What does `koalastomax` do?
  - What does each circle represent?
- Write a high-level design for it.
- Translate into a `main()` with function calls.

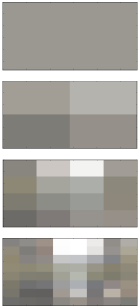
# From Last Time: koalas

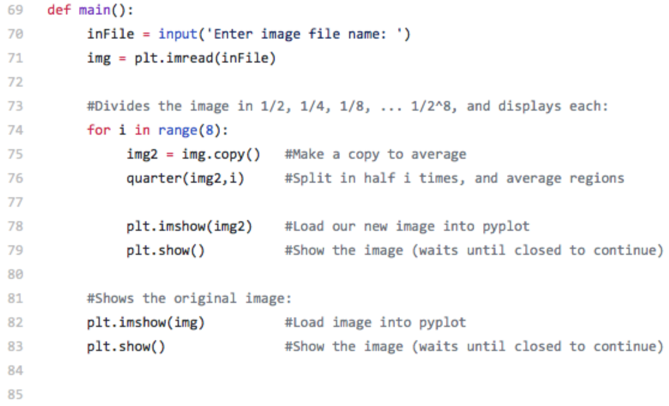




```
def main():
    inFile = input('Enter image file name: ')
    img = plt.imread(inFile)

    #Divides the image in 1/2, 1/4, 1/8, ... 1/2^8, and displays each:
    for i in range(8):
        img2 = img.copy()    #Make a copy to average
        quarter(img2,i)      #Split in half i times, and average regions

        plt.imshow(img2)     #Load our new image into pyplot
        plt.show()           #Show the image (waits until closed to continue)

    #Shows the original image:
    plt.imshow(img)          #Load image into pyplot
    plt.show()               #Show the image (waits until closed to continue)
```

# From Last Time: koalas

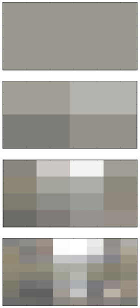


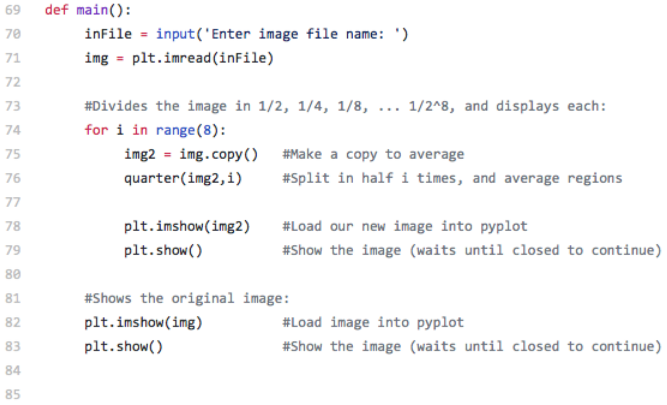


```
def main():
    inFile = input('Enter image file name: ')
    img = plt.imread(inFile)

    #Divides the image in 1/2, 1/4, 1/8, ... 1/2^8, and displays each:
    for i in range(8):
        img2 = img.copy()    #Make a copy to average
        quarter(img2,i)      #Split in half i times, and average regions

        plt.imshow(img2)     #Load our new image into pyplot
        plt.show()           #Show the image (waits until closed to continue)

    #Shows the original image:
    plt.imshow(img)          #Load image into pyplot
    plt.show()               #Show the image (waits until closed to continue)


```

- The `main()` is written for you.

## From Last Time: koalas



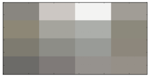


```
def main():
    inFile = input('Enter image file name: ')
    img = plt.imread(inFile)

    #Divides the image in 1/2, 1/4, 1/8, ... 1/2^8, and displays each:
    for i in range(8):
        img2 = img.copy()    #Make a copy to average
        quarter(img2,i)      #Split in half i times, and average regions

        plt.imshow(img2)     #Load our new image into pyplot
        plt.show()           #Show the image (waits until closed to continue)

    #Shows the original image:
    plt.imshow(img)          #Load image into pyplot
    plt.show()               #Show the image (waits until closed to continue)
```

- The `main()` is written for you.
- Only fill in two functions: `average()` and `setRegion()`.

## From Last Time: koalas

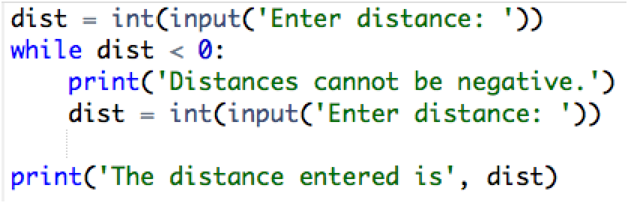
*Process:*



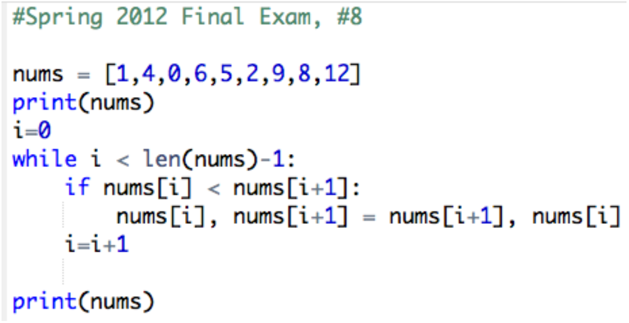
Get template $\rightarrow$ Fill in missing $\rightarrow$ Test locally $\rightarrow$ Submit to
from `github` $\rightarrow$ functions $\rightarrow$ idle3/python3 $\rightarrow$ Gradescope

## In Pairs or Triples:

*Predict what the code will do:*

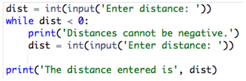
- 

```python
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```

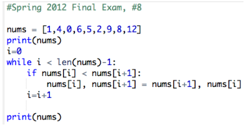
- 

```python
#Spring 2012 Final Exam, #8

nums = [1,4,0,6,5,2,9,8,12]
print(nums)
i=0
while i < len(nums)-1:
    if nums[i] < nums[i+1]:
        nums[i], nums[i+1] = nums[i+1], nums[i]
    i=i+1

print(nums)
```

## Python Tutor



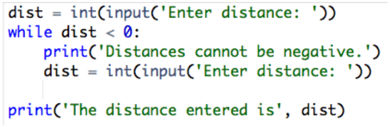
```
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```



```
#Spring 2012 Final Exam, #8

nums = [1,4,0,6,5,2,9,8,12]
print(nums)
i=0
while i < len(nums)-1:
    if nums[i] < nums[i+1]:
        nums[i], nums[i+1] = nums[i+1], nums[i]
    i=i+1

print(nums)
```

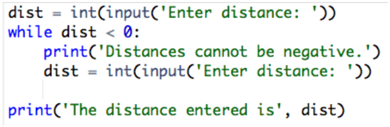
(Demo with `pythonTutor`)

## Indefinite Loops

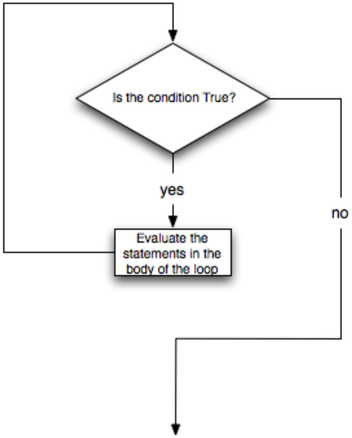


```
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```

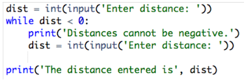
## Indefinite Loops



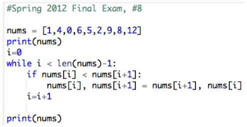
```
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```



# Indefinite Loops



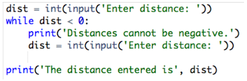
```
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```



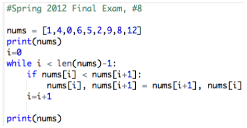
```
#Spring 2012 Final Exam, #8

nums = [1,4,0,6,5,2,9,8,12]
print(nums)
i=0
while i < len(nums)-1:
    if nums[i] < nums[i+1]:
        nums[i], nums[i+1] = nums[i+1], nums[i]
    i=i+1

print(nums)
```

- Indefinite loops repeat as long as the condition is true.

## Indefinite Loops



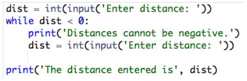
```
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```



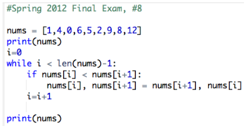
```
#Spring 2012 Final Exam, #8

nums = [1,4,0,6,5,2,9,8,12]
print(nums)
i=0
while i < len(nums)-1:
    if nums[i] < nums[i+1]:
        nums[i], nums[i+1] = nums[i+1], nums[i]
    i=i+1

print(nums)
```

- Indefinite loops repeat as long as the condition is true.
- Could execute the body of the loop zero times, 10 times, infinite number of times.

## Indefinite Loops



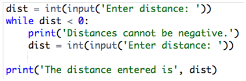
```
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```



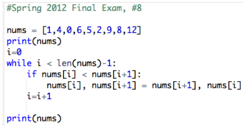
```
#Spring 2012 Final Exam, #8

nums = [1,4,0,6,5,2,9,8,12]
print(nums)
i=0
while i < len(nums)-1:
    if nums[i] < nums[i+1]:
        nums[i], nums[i+1] = nums[i+1], nums[i]
    i=i+1

print(nums)
```

- Indefinite loops repeat as long as the condition is true.
- Could execute the body of the loop zero times, 10 times, infinite number of times.
- The condition determines how many times.

## Indefinite Loops



```
dist = int(input('Enter distance: '))
while dist < 0:
    print('Distances cannot be negative.')
    dist = int(input('Enter distance: '))

print('The distance entered is', dist)
```



```
#Spring 2012 Final Exam, #8

nums = [1,4,0,6,5,2,9,8,12]
print(nums)
i=0
while i < len(nums)-1:
    if nums[i] < nums[i+1]:
        nums[i], nums[i+1] = nums[i+1], nums[i]
    i=i+1

print(nums)
```

- Indefinite loops repeat as long as the condition is true.
- Could execute the body of the loop zero times, 10 times, infinite number of times.
- The condition determines how many times.
- Very useful for checking input, simulations, and games.

## In Pairs or Triples:

*Answer the following questions on your lecture slip (paper or* `tinyurl.com/yamkjh96`*):*

Of the students in the room,

- Whose name comes first alphabetically?
- Whose name comes last alphabetically?
- Is there someone in the room with your initials?

## In Pairs or Triples:

*While we tabulate results:*

Design a program that takes a CSV file and a set of initials:

- Whose name comes first alphabetically?
- Whose name comes last alphabetically?
- Is there someone in the room with your initials?

## Results: Lecture Slip Question

(Show tabulated results...)

## Design Question: Find first alphabetically

- In Pandas, lovely built-in functions:

## Design Question: Find first alphabetically

- In Pandas, lovely built-in functions:
  - `df.sort_values('First Name')` and
  - `df['First Name'].min()`

## Design Question: Find first alphabetically

- In Pandas, lovely built-in functions:
  - `df.sort_values('First Name')` and
  - `df['First Name'].min()`
- What if you don't have a CSV and DataFrame, or data not ordered?

## Design Question: Find first alphabetically

- In Pandas, lovely built-in functions:
  - `df.sort_values('First Name')` and
  - `df['First Name'].min()`
- What if you don't have a CSV and DataFrame, or data not ordered?

## Design Question: Find first alphabetically

- What if you don't have a CSV and DataFrame, or data not ordered?

## Design Question: Find first alphabetically

- What if you don't have a CSV and DataFrame, or data not ordered?
- Useful *Design Pattern*: min/max

## Design Question: Find first alphabetically

- What if you don't have a CSV and DataFrame, or data not ordered?
- Useful *Design Pattern*: min/max
  - Set a variable to worst value (i.e. `maxN = 0` or `first = "ZZ"`).

## Design Question: Find first alphabetically

- What if you don't have a CSV and DataFrame, or data not ordered?
- Useful *Design Pattern*: min/max
  - Set a variable to worst value (i.e. `maxN = 0` or `first = "ZZ"`).
  - For each item, X, in the list:

## Design Question: Find first alphabetically

- What if you don't have a CSV and DataFrame, or data not ordered?
- Useful *Design Pattern*: min/max
  - Set a variable to worst value (i.e. `maxN = 0` or `first = "ZZ"`).
  - For each item, X, in the list:
    - ★ Compare X to your variable.

## Design Question: Find first alphabetically

- What if you don't have a CSV and DataFrame, or data not ordered?
- Useful *Design Pattern*: min/max
  - Set a variable to worst value (i.e. `maxN = 0` or `first = "ZZ"`).
  - For each item, X, in the list:
    - ★ Compare X to your variable.
    - ★ If better, update your variable to be X.

## Design Question: Find Matching Initials

- How do we stop, if we find a match?

## Design Question: Find Matching Initials

- How do we stop, if we find a match?
- Change the loop to be indefinite (i.e. `while` loop):
  - Set a variable to `found = False`

## Design Question: Find Matching Initials

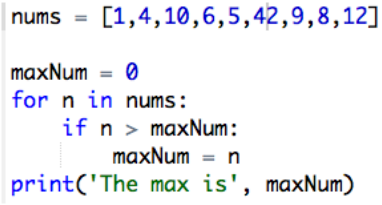
- How do we stop, if we find a match?
- Change the loop to be indefinite (i.e. `while` loop):
  - Set a variable to `found = False`
  - while there are items in the list and not `found`

## Design Question: Find Matching Initials

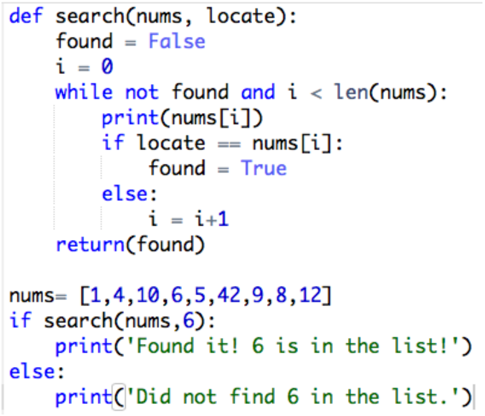
- How do we stop, if we find a match?
- Change the loop to be indefinite (i.e. `while` loop):
  - Set a variable to `found = False`
  - while there are items in the list and not `found`
    - If item matches your value, set `found = True`

## In Pairs or Triples:

- *Predict what the code will do:*



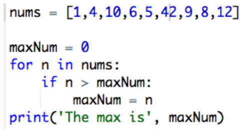
```
nums = [1,4,10,6,5,42,9,8,12]

maxNum = 0
for n in nums:
    if n > maxNum:
        maxNum = n
print('The max is', maxNum)
```



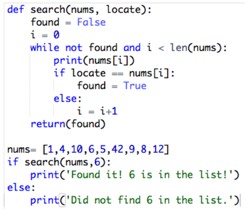
```
def search(nums, locate):
    found = False
    i = 0
    while not found and i < len(nums):
        print(nums[i])
        if locate == nums[i]:
            found = True
        else:
            i = i+1
    return(found)

nums= [1,4,10,6,5,42,9,8,12]
if search(nums,6):
    print('Found it! 6 is in the list!')
else:
    print('Did not find 6 in the list.')
```

- Write a function that asks a user for number after 2000 but before 2018. The function should repeatedly ask the user for a number until they enter one within the range and return the number.

## Python Tutor



```
nums = [1,4,10,6,5,42,9,8,12]

maxNum = 0
for n in nums:
    if n > maxNum:
        maxNum = n
print('The max is', maxNum)
```



```
def search(nums, locate):
    found = False
    i = 0
    while not found and i < len(nums):
        print(nums[i])
        if locate == nums[i]:
            found = True
        else:
            i = i+1
    return(found)

nums= [1,4,10,6,5,42,9,8,12]
if search(nums,6):
    print('Found it! 6 is in the list!')
else:
    print('Did not find 6 in the list.')
```

(Demo with `pythonTutor`)

## Coding

- Write a function that asks a user for number after 2000 but before 2018. The function should repeatedly ask the user for a number until they enter one within the range and return the number..

## Coding

- Write a function that asks a user for number after 2000 but before 2018. The function should repeatedly ask the user for a number until they enter one within the range and return the number.

```
def getYear():
```

## Coding

- Write a function that asks a user for number after 2000 but before 2018. The function should repeatedly ask the user for a number until they enter one within the range and return the number.

```
def getYear():







    return(num)
```

## Coding

- Write a function that asks a user for number after 2000 but before 2018. The function should repeatedly ask the user for a number until they enter one within the range and return the number.

```
def getYear():
    num = 0




    return(num)
```

## Coding

- Write a function that asks a user for number after 2000 but before 2018. The function should repeatedly ask the user for a number until they enter one within the range and return the number.

```
def getYear():
    num = 0
    while num <= 2000 or num >= 2018:




    return(num)
```

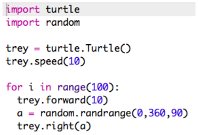
## Coding

- Write a function that asks a user for number after 2000 but before 2018. The function should repeatedly ask the user for a number until they enter one within the range and return the number.

```
def getYear():
    num = 0
    while num <= 2000 or num >= 2018:
        num = int(input('Enter a number > 2000 & < 2018'))
    return(num)
```

## Python's random package

- Python has a built-in package for generating pseudo-random numbers.

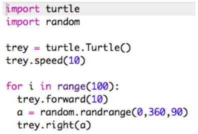


```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

## Python's random package

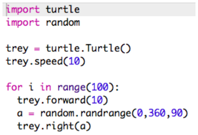
- Python has a built-in package for generating pseudo-random numbers.
- To use:

```
import random
```



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

## Python's random package



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Python has a built-in package for generating pseudo-random numbers.
- To use:

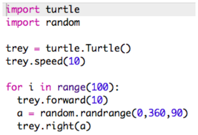
  ```
  import random
  ```

- Useful command to generate whole numbers:

  ```
  random.randrange(start,stop,step)
  ```

  which gives a number chosen randomly from the specified range.

## Python's random package



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Python has a built-in package for generating pseudo-random numbers.
- To use:

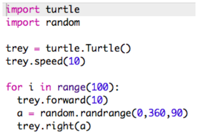
  ```
  import random
  ```

- Useful command to generate whole numbers:

  ```
  random.randrange(start,stop,step)
  ```

  which gives a number chosen randomly from the specified range.
- Useful command to generate real numbers:

## Python's random package



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Python has a built-in package for generating pseudo-random numbers.
- To use:

  `import random`

- Useful command to generate whole numbers:

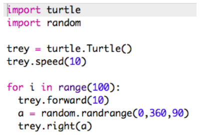
  `random.randrange(start,stop,step)`

  which gives a number chosen randomly from the specified range.
- Useful command to generate real numbers:

  `random.random()`

  which gives a number chosen (uniformly) at random from [0.0,1.0).

## Python's random package



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Python has a built-in package for generating pseudo-random numbers.
- To use:

  `import random`

- Useful command to generate whole numbers:

  `random.randrange(start,stop,step)`

  which gives a number chosen randomly from the specified range.

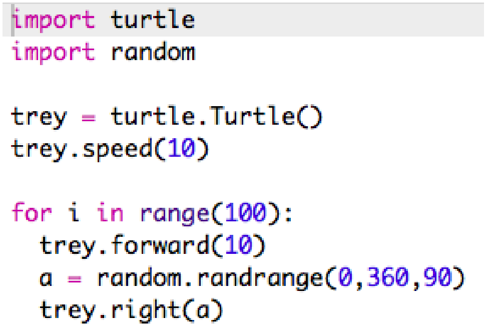
- Useful command to generate real numbers:

  `random.random()`

  which gives a number chosen (uniformly) at random from [0.0,1.0).

- Very useful for simulations, games, and testing.

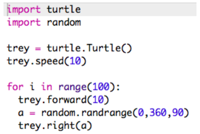
## Trinket



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

(Demo turtle random walk)

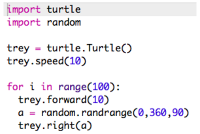
## Recap: Indefinite Loops & Random Numbers



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Indefinite (while) loops allow you to repeat a block of code as long as a condition holds.

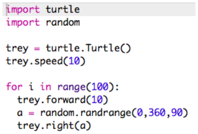
## Recap: Indefinite Loops & Random Numbers



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Indefinite (while) loops allow you to repeat a block of code as long as a condition holds.
- Very useful for checking user input for correctness.

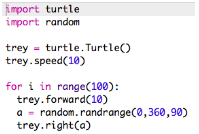
## Recap: Indefinite Loops & Random Numbers



```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Indefinite (while) loops allow you to repeat a block of code as long as a condition holds.
- Very useful for checking user input for correctness.
- Python's built-in random package has useful methods for generating random whole numbers and real numbers.

## Recap: Indefinite Loops & Random Numbers


```
import turtle
import random

trey = turtle.Turtle()
trey.speed(10)

for i in range(100):
  trey.forward(10)
  a = random.randrange(0,360,90)
  trey.right(a)
```

- Indefinite (while) loops allow you to repeat a block of code as long as a condition holds.
- Very useful for checking user input for correctness.
- Python's built-in random package has useful methods for generating random whole numbers and real numbers.
- To use, must include: `import random`.