### CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

< ロ ト < 団 ト < 三 ト < 三 ト</p>

CSci 127 (Hunter)

Lecture 7

Э 19 March 2019 1 / 43

990

#### Announcements



• Guest Lecturer: Dr. Tiziana Ligorio

CSci 127 (Hunter)

Lecture 7

Э 19 March 2019 2 / 43

-

590

イロト イロト イヨト

## Today's Topics



- Recap: Slicing & Images
- Introduction to Functions
- NYC Open Data

Э

900

## Today's Topics



#### • Recap: Slicing & Images

- Introduction to Functions
- NYC Open Data

990

## In Pairs or Triples:

Review: predict what the code will do:

```
motto = "Mihi Cura Futuri"
l = len(motto)
for i in range(l):
    print(motto[i])
for j in range(l-1,-1,-1):
    print(motto[j])
```

```
import matplotlib.pyplot as plt
import numpy as np
img = plt.imread('csBridge')
plt.imshow(img)
plt.show()
height = img.shape[0]
width = ima.shape[1]
img2 = img[:height//2, :width//2]
plt.imshow(ima2)
plt.show()
```

Sac

### Python Tutor

```
motto = "Mihi Cura Futuri"
l = len(motto)
for i in range(l):
    print(motto[i])
for j in range(l-1,-1,-1):
    print(motto[j])
```

(Demo with pythonTutor)

999

#### Images

```
import matplotlib.pyplot as plt
import numpy as np
img = plt.imread('csBridge')
plt.imshow(img)
plt.show()
height = img.shape[0]
width = img.shape[1]
img2 = img[:height//2, :width//2]
plt.imshow(img2)
plt.show()
```

```
import matplotlib.pyplot as plt
import numpy as np
img = plt.imread('csBridge')
plt.imshow(img)
plt.show()
height = img.shape[0]
width = img.shape[1]
img2 = img[:height//2, :width//2]
plt.imshow(img2)
plt.show()
```



19 March 2019 8 / 43

3

Sac

```
import matplotlib.pyplot as plt
import numpy as np
img = plt.imread('csBridge')
plt.imshow(img)
plt.show()
height = img.shape[0]
width = img.shape[1]
img2 = img[:height//2, :width//2]
plt.imshow(img2)
plt.show()
```



19 March 2019 9 / 43

3

Sac

```
import matplotlib.pyplot as plt
import numpy as np
img = plt.imread('csBridge')
plt.imshow(img)
plt.show()
height = img.shape[0]
width = img.shape[1]
img2 = img[:height//2, :width//2]
plt.imshow(img2)
plt.show()
```



3

Sac



• How would you select the lower left corner?

3



• How would you select the lower left corner? img2 = img[height//2:, :width//2]



- How would you select the lower left corner? img2 = img[height//2:, :width//2]
- How would you select the upper right corner?



- How would you select the lower left corner? img2 = img[height//2:, :width//2]
- How would you select the upper right corner? img2 = img[:height//2, width//2:]

CSci 127 (Hunter)



- How would you select the lower left corner? img2 = img[height//2:, :width//2]
- How would you select the upper right corner? img2 = img[:height//2, width//2:]
- How would you select the lower right corner?

CSci 127 (Hunter)

イロト イポト イヨト イヨト

19 March 2019

10 / 43



- How would you select the lower left corner? img2 = img[height//2:, :width//2]
- How would you select the upper right corner? img2 = img[:height//2, width//2:]
- How would you select the lower right corner? img2 = img[height//2:, width//2:]

CSci 127 (Hunter)

イロト イポト イヨト イヨト

19 March 2019

10 / 43

### Today's Topics



- Recap: Slicing & Images
- Introduction to Functions
- NYC Open Data

Э

996

```
• Functions are a way to break code into pieces, that can be easily reused.
```

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
def main():
    print("Hello, World!")
if __name__ == "__main__":
    main()
```

イロト 不良 トイヨト イヨト ヨー のくや

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- Many languages require that all code must be organized with functions.

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
def main():
```

```
print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- Many languages require that all code must be organized with functions.
- The opening function is often called main()

Sac

<ロト < 団ト < 団ト < 団ト < 団ト = 三</p>

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
```

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- Many languages require that all code must be organized with functions.
- The opening function is often called main()
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis:

= nar

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
```

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- Many languages require that all code must be organized with functions.
- The opening function is often called main()
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")

Sac

イロト 不得 トイヨト イヨト 二日

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
```

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- Many languages require that all code must be organized with functions.
- The opening function is often called main()
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or define your own functions,

Sac

イロト 不得 トイヨト イヨト 二日

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
```

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- Many languages require that all code must be organized with functions.
- The opening function is often called main()
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or define your own functions, which are stored, until invoked or called.

Sac

"Hello, World!" with Functions

- #Name: your name here
  #Date: October 2017
  #This program, uses functions,
  # says hello to the world!
- def main():
   print("Hello, World!")

if \_\_name\_\_ == "\_\_main\_\_":
 main()

CSci 127 (Hunter)

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

### Python Tutor

#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!

def main():
 print("Hello, World!")

if \_\_name\_\_ == "\_\_main\_\_": main() (Demo with pythonTutor)

イロト 不良 トイヨト イヨト ヨー のくや

# In Pairs or Triples:

Predict what the code will do:

```
def totalWithTax(food,tip):
    total = 0
    tax = 0.0875
    total = food + food * tax
    total = total + tip
    return(total)
```

```
lunch = float(input('Enter lunch total: '))
lTip = float(input('Enter lunch tip:' ))
lTotal = totalWithTax(lunch, lTip)
print('Lunch total is', lTotal)
```

```
dinner= float(input('Enter dinner total: '))
dTip = float(input('Enter dinner tip:' ))
dTotal = totalWithTax(dinner, dTip)
print('Dinner total is', dTotal)
```

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

### Python Tutor

def totalWithTax(food,tip): total = 0 tax = 0.0875 total = food + food \* tax total = total + tip return(total)

lunch = float(input('Enter lunch total: '))
lTip = float(input('Enter lunch tip:' ))
lTotal = totalWithTax(lunch, lTip)
print('Lunch total is', lTotal)

dinner= float(input('Enter dinner total: '))
dTip = float(input('Enter dinner tip:' ))
dTotal = totalWithTax(dinner, dTip)
print('Dinner total is', dTotal)

#### (Demo with pythonTutor)

▲□▶ ▲□▶ ▲三▶ ▲三▶ 三三 - のの⊙

• Functions can have **input parameters**.

```
def totalWithTax(food,tip):
    total = 0
    tax = 0.0875
    total = food + food * tax
    total = total + tip
    return(total)
lunch = float(input('Enter lunch tip:' ))
lTotal = totalWithTax(lunch, lTip)
print('Lunch total is', lTotal)
dinner= float(input('Enter dinner total: '))
dTotal = totalWithTax(dinner, dTip)
print('Dinner total is', dTotal)
```

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

```
def totalWithTax(food,tip):
    total = 0
    tax = 0.0875
    total = food + food * tax
    total = total + tip
    return(total)
lunch = float(input('Enter lunch tip:' ))
lTip = float(input('Enter lunch tip:' ))
lTotal = totalWithTax(lunch, lTip)
print('Lunch total is', lTotal)
dinner= float(input('Enter dinner total: '))
dTip = float(input('Enter dinner tip:' ))
dTotal = totalWithTax(dinner, dTip)
print('Dinner total is', dTotal)
```

- Functions can have input parameters.
- Surrounded by parentheses, both in the function definition, and in the function call (invocation).

19 March 2019 17 / 43

200

イロト 不得 トイヨト イヨト 二日

```
def totalWithTax(food,tip):
    total = 0
    tax = 0.0875
    total = food + food * tax
    total = total + tip
    return(total)
```

```
lunch = float(input('Enter lunch total: '))
ITip = float(input('Enter lunch tip:'))
ITotal = totalWithTax(lunch, ITip)
print('Lunch total is', lTotal)
```

```
dinner= float(input('Enter dinner total: '))
dTip = float(input('Enter dinner tip:' ))
dTotal = totalWithTax(dinner, dTip)
print('Dinner total is', dTotal)
```

- Functions can have input parameters.
- Surrounded by parentheses, both in the function definition, and in the function call (invocation).
- The "placeholders" in the function definition: **formal parameters**.

Sac

```
def totalWithTax(food,tip):
    total = 0
    tax = 0.0875
    total = food + food * tax
    total = total + tip
    return(total)
```

```
lunch = float(input('Enter lunch total: '))
ITip = float(input('Enter lunch tip:'))
ITotal = totalWithTax(lunch, ITip)
print('Lunch total is', ITotal)
```

```
dinner= float(input('Enter dinner total: '))
dTip = float(input('Enter dinner tip:' ))
dTotal = totalWithTax(dinner, dTip)
print('Dinner total is', dTotal)
```

- Functions can have input parameters.
- Surrounded by parentheses, both in the function definition, and in the function call (invocation).
- The "placeholders" in the function definition: **formal parameters**.
- The ones in the function call: actual parameters

Sac

```
def totalWithTax(food,tip):
    total = 0
    tax = 0.0875
    total = food + food * tax
    total = total + tip
    return(total)
```

```
lunch = float(input('Enter lunch total: '))
ITip = float(input('Enter lunch tip:'))
ITotal = totalWithTax(lunch, ITip)
print('Lunch total is', lTotal)
```

```
dinner= float(input('Enter dinner total: '))
dTip = float(input('Enter dinner tip:' ))
dTotal = totalWithTax(dinner, dTip)
print('Dinner total is', dTotal)
```

- Functions can have input parameters.
- Surrounded by parentheses, both in the function definition, and in the function call (invocation).
- The "placeholders" in the function definition: formal parameters.
- The ones in the function call: actual parameters
- Functions can also return values to where it was called.

200

イロト 不得 トイヨト イヨト 二日



- Functions can have input parameters.
- Surrounded by parentheses, both in the function definition, and in the function call (invocation).
- The "placeholders" in the function definition: **formal parameters**.
- The ones in the function call: actual parameters.
- Functions can also return values to where it was called.

Sac

# In Pairs or Triples:

Circle the actual parameters and underline the formal parameters:

```
def prob4():
    verse = "jam tomorrow and jam yesterday,"
    print("The rule is.")
    c = mystery(verse)
    w = enigma(verse.c)
    print(c,w)
def mystery(v):
    print(v)
    c = v.count("jam")
    return(c)
def enigma(v,c):
    print("but never", v[-1])
    for i in range(c):
        print("jam")
    return("day.")
prob4()
                                            ◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○
  CSci 127 (Hunter)
                              Lecture 7
```

19 March 2019

19 / 43

# In Pairs or Triples:

Circle the actual parameters and underline the formal parameters:



20 / 43
## In Pairs or Triples:

Predict what the code will do:

```
def prob4():
    verse = "jam tomorrow and jam yesterday,"
    print("The rule is,")
    c = mystery(verse)
    w = enigma(verse,c)
    print(c,w)
def mystery(v):
    print(v)
    c = v.count("jam")
    return(c)
def enigma(v,c):
    print("but never", v[-1])
    for i in range(c):
        print("jam")
    return("day.")
prob4()
```

CSci 127 (Hunter)

19 March 2019 21 / 43

#### Python Tutor

```
def prob4();
    verse = "iam tomorrow and iam vesterday."
    print("The rule is,")
   c = mystery(verse)
   w = enigma(verse,c)
    print(c.w)
def mystery(v):
    print(v)
   c = v.count("jam")
    return(c)
def enigma(v,c):
    print("but never", v[-1])
    for i in range(c):
        print("jam")
    return("day.")
prob4()
```

#### (Demo with pythonTutor)

イロト 不良 トイヨト イヨト ヨー のくや

In Pairs or Triples:

Predict what the code will do:

```
#Greet loop example
```

```
def greetLoop(person):
    print("Greetings")
    for i in range(5):
        print("Hello", person)
```

```
greetLoop("Thomas")
```

```
# From "Teaching with Python" by John Zelle
def happy():
    print("Happy Birthday to you!")
def sing(P):
    happy()
    print("Happy Birthday dear " + P + "!")
    happy()
sing("Fred")
sing("Thomas")
```

sing("Hunter")

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

#### Python Tutor

#Greet loop example

```
def greetLoop(person):
    print("Greetings")
    for i in range(5):
        print("Hello", person)
```

greetLoop("Thomas")

# From "Teaching with Python" by John Zelle

def happy():
 print("Happy Birthday to you!")

def sing(P):

happy() happy() print("Happy Birthday dear " + P + "!") happy()

sing("Fred")
sing("Thomas")
sing("Hunter")

#### (Demo with pythonTutor)

▲□▶ ▲□▶ ▲ヨ▶ ▲ヨ▶ ヨ - のの⊙

### In Pairs or Triples:

```
Fill in the missing code:
```

```
def monthString(monthNum):
    Takes as input a number, monthNum, and
    returns the corresponding month name as a string.
    Example: monthStrina(1) returns "January".
    Assumes that input is an integer ranging from 1 to 12
    monthString = ""
    *******
    ### FILL IN YOUR CODE HERE
                                 ###
    ### Other than your name above, ###
    ### this is the only section
                                 ###
    ### you change in this program. ###
    *****
    return(monthString)
def main():
    n = int(input('Enter the number of the month: '))
    mString = monthString(n)
    print('The month is'. mString)
                                                         イロト イポト イヨト イヨト 二日
```

```
CSci 127 (Hunter)
```

19 March 2019 25 / 43

200

def monthString(monthNum):

Takes as input a number, monthNum, and returns the corresponding month name as a string. Example: monthString(1) returns "January". Assumes that input is an integer ranging from 1 to 12

monthString = ""

#### 

return(monthString)

#### def main():

n = int(input('Enter the number of the month: '))
nString = monthString(n)
print('The month is', mString)

#### (Demo with IDLE)

CSci 127 (Hunter)

19 March 2019 26 / 43

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
def main():
    print("Hello, World!")
if __name__ == "__main__":
    main()
```

• Functions are a way to break code into pieces, that can be easily reused.

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
```

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis:

3

Sac

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!
```

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")

Sac

#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or define your own functions,

Sac

#Name: your name here
#Date: October 2017
#This program, uses functions,
# says hello to the world!

```
def main():
    print("Hello, World!")
```

```
if __name__ == "__main__":
    main()
```

- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or define your own functions, which are stored, until invoked or called.

Sac

#### Today's Topics



- Recap: Slicing & Images
- Introduction to Functions
- NYC Open Data

3

590

## Open Data for All New Yorkers

Where can you find public Wi-Fi in your neighborhood? What kind of tree is in front of your office? Learn about where you live, work, eat, shop and play using NYC Open Data.

Search Open Data for things like 311, Buildings, Crime



• Freely available source of data.

# Open Data for All New Yorkers

Where can you find public Wi-Fi in your neighborhood? What kind of tree is in front of your office? Learn about where you live, work, eat, shop and play using NYC Open Data.

Search Open Data for things like 311, Buildings, Crime



- Freely available source of data.
- Maintained by the NYC data analytics team.

 $\exists \rightarrow$ 

# Open Data for All New Yorkers

Where can you find public Wi-Fi in your neighborhood? What kind of tree is in front of your office? Learn about where you live, work, eat, shop and play using NYC Open Data.

Search Open Data for things like 311, Buildings, Crime



- Freely available source of data.
- Maintained by the NYC data analytics team.
- We will use several different ones for this class.

 $\exists \rightarrow$ 

# Open Data for All New Yorkers

Where can you find public Wi-Fi in your neighborhood? What kind of tree is in front of your office? Learn about where you live, work, eat, shop and play using NYC Open Data.



Search Open Data for things like 311, Buildings, Crime

- Freely available source of data.
- Maintained by the NYC data analytics team.
- We will use several different ones for this class.
- Will use pandas, pyplot & folium libraries to analyze, visualize and map the data.

CSci 127 (Hunter)

19 March 2019 29 / 43

- 4 同下 4 国下 4 国



Where can you find public Wi-Fi in your neighborhood? What kind of tree is in front of your office? Learn about where you live, work, eat, shop and play using NYC Open Data.



Search Open Data for things like 311, Buildings, Crime

- Freely available source of data.
- Maintained by the NYC data analytics team.
- We will use several different ones for this class.
- Will use pandas, pyplot & folium libraries to analyze, visualize and map the data.
- Lab 7 covers accessing and downloading NYC OpenData datasets.

CSci 127 (Hunter)



(Demo OpenData movie.)

3

Sac

## NYC OpenData

Home Data About ~ Learn

#### Film Permits

Permits are generally required when asserting the exclusive use of city property, like a sidewalk, a street, or a park. See http://www1.nyc.gov/site/mome/permits/when-permit-required.page

EventID :	EventType :	StartDateTi :	EndDateTime :	EnteredOn ↓ :	EventAg :	ParkingHeld :	Borou
455063	Shooting Permit	12/06/2018 07:00	12/06/2018 09:00	12/05/2018 12:36	Mayor's Offic	STARR AVENUE b	Queens
454967	Shooting Permit	12/06/2018 07:00	12/06/2018 05:00	12/04/2018 09:11	Mayor's Offic	EAGLE STREET be	Brooklyn
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 07:00	12/04/2018 05:44	Mayor's Offic	SOUTH OXFORD	Brooklyn
454920	Shooting Permit	12/06/2018 10:00	12/06/2018 11:59	12/04/2018 03:28	Mayor's Offic	13 AVENUE betw	Queens
454914	Shooting Permit	12/06/2018 08:00	12/06/2018 11:00	12/04/2018 03:05	Mayor's Offic	ELDERT STREET b	Brooklyn
454909	Shooting Permit	12/05/2018 08:00	12/05/2018 06:00	12/04/2018 02:45	Mayor's Offic	ELDERT STREET b	Brooklyn
454905	Shooting Permit	12/06/2018 07:00	12/06/2018 10:00	12/04/2018 02:17	Mayor's Offic	35 STREET betwe	Queens

CSci 127 (Hunter)

Lecture 7

・ロト ・日 ・ ・ ヨ ・ ・ ヨ ・ うへぐ

19 March 2019 31 / 43

#### CSci 127 (Hunter)

## Example: OpenData Film Permits

NYC OpenData

Home Data About -> Learn -> Alerts Contact Us Blog Q Sign In

Film Pern	nits							2)	1.2		C. Find in	n this Datase	et.
Permits are street, or a p	generally required park. See http://ww	d when asserting th w1.nyc.gov/site/mo	e exclusive use of ome/permits/when-	М	ore Views F	ilter Visuali	ize Export	Discuss E	mbed About				
EventID :	EventType i	StartDateTi	EndDateTime	EnteredOn 4	EventAg	ParkingHeld :	Borou i	Com i	Police i	Categ i	SubC i	Count i	ZipCo i
455063	Shooting Permit	12/06/2018 07:00	12/06/2018 09:00	12/05/2018 12:36	Mayor's Offic	STARR AVENUE b	Queens	2	108	Television	Episodic s	United Sta	11101
454967	Shooting Permit	12/06/2018 07:00	12/06/2018 05:00	12/04/2018 09:11	Mayor's Offic	EAGLE STREET be	Brooklyn	1	94	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/05/2018 07:00	12/04/2018 05:44	Mayor's Offic	SOUTH OXFORD	Brooklyn	2, 6	76, 88	Still Photo	Not Applic	United Sta	11217, 11
454920	Shooting Permit	12/06/2018 10:00	12/05/2018 11:59	12/04/2018 03:28	Mayor's Offic	13 AVENUE betw	Queens	1, 3, 7	109, 7, 90	Film	Feature	United Sta	10002, 11
454914	Shooting Permit	12/06/2018 08:00	12/05/2018 11:00	12/04/2018 03:05	Mayor's Offic	ELDERT STREET b	Brooklyn	4, 5	104, 75, 83	Television	Episodic s	United Sta	11207, 11
454909	Shooting Permit	12/05/2018 08:00	12/05/2018 06:00	12/04/2018 02:45	Mayor's Offic	ELDERT STREET b	Brooklyn	4	83	Television	Episodic s	United Sta	11237
454905	Shooting Permit	12/06/2018 07:00	12/05/2018 10:00	12/04/2018 02:17	Mayor's Offic	35 STREET betwe	Queens	1	114	Television	Cable-epis	United Sta	11101, 11

Lecture 7

• What's the most popular street for filming?

#### CSci 127 (Hunter)

Example: OpenData Film Permits

NYC OpenData

Home Data About - Learn - Alerts Contact Us Blog Q Sign In

Film Pern	nits				2)	1.9		🔍 Find in	h this Datase	st.			
Permits are street, or a p	generally required bark. See http://ww	when asserting th w1.nyc.gov/site/mo	e exclusive use of me/permits/when-	city property, like -permit-required.pa	a sidewalk, a 🖺 age	1		Ν	fore Views F	ilter Visual	ze Export	Discuss Er	nbed About
EventID :	EventType :	StartDateTi	EndDateTime	EnteredOn 4	EventAg	ParkingHeld i	Borou i	Com i	Police i	Categ i	SubC i	Count i	ZipCo i
455063	Shooting Permit	12/06/2018 07:00	12/06/2018 09:00	12/05/2018 12:36	Mayor's Offic	STARR AVENUE b	Queens	2	108	Television	Episodic s	United Sta	11101
454967	Shooting Permit	12/06/2018 07:00	12/06/2018 05:00	12/04/2018 09:11	Mayor's Offic	EAGLE STREET be	Brooklyn	1	94	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/05/2018 07:00	12/04/2018 05:44	Mayor's Offic	SOUTH OXFORD	Brooklyn	2, 6	76, 88	Still Photo	Not Applic	United Sta	11217, 11
454920	Shooting Permit	12/06/2018 10:00	12/05/2018 11:59	12/04/2018 03:28	Mayor's Offic	13 AVENUE betw	Queens	1, 3, 7	109, 7, 90	Film	Feature	United Sta	10002, 11
454914	Shooting Permit	12/06/2018 08:00	12/05/2018 11:00	12/04/2018 03:05	Mayor's Offic	ELDERT STREET b	Brooklyn	4, 5	104, 75, 83	Television	Episodic s	United Sta	11207, 11
454909	Shooting Permit	12/05/2018 08:00	12/05/2018 06:00	12/04/2018 02:45	Mayor's Offic	ELDERT STREET b	Brooklyn	4	83	Television	Episodic s	United Sta	11237
454905	Shooting Permit	12/06/2018 07:00	12/06/2018 10:00	12/04/2018 02:17	Mayor's Offic	35 STREET betwe	Queens	1	114	Television	Cable-epis	United Sta	11101, 11

- What's the most popular street for filming?
- What's the most popular borough?

-----

19 March 2019 32 / 43

▲□▶ ▲□▶ ▲三▶ ▲三▶ 三三 ろの⊙

#### CSci 127 (Hunter)

#### Example: OpenData Film Permits

NYC OpenData

Ciles Describe

Home Data About -> Learn -> Alerts Contact Us Blog Q Sign In

. . . . . . . . .

Fillineen	i ii us					27	1.9		G Find in				
Permits are street, or a	generally required park. See http://ww	when asserting th w1.nyc.gov/site/mo	e exclusive use of me/permits/when-	м	ore Views	ilter Visuali	ze Export	Discuss En	nbed About				
EventID i	EventType i	StartDateTi i	EndDateTime	EnteredOn $\downarrow$ i	EventAg i	ParkingHeld :	Borou	Com i	Police i	Categ i	SubC i	Count i	ZipCo i
455063	Shooting Permit	12/06/2018 07:00	12/06/2018 09:00	12/05/2018 12:36	Mayor's Offic	STARR AVENUE b	Queens	2	108	Television	Episodic s	United Sta	11101
454967	Shooting Permit	12/06/2018 07:00	12/06/2018 05:00	12/04/2018 09:11	Mayor's Offic	EAGLE STREET be	Brooklyn	1	94	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 07:00	12/04/2018 05:44	Mayor's Offic	SOUTH OXFORD	Brooklyn	2, 6	76, 88	Still Photo	Not Applic	United Sta	11217, 11
454920	Shooting Permit	12/06/2018 10:00	12/06/2018 11:59	12/04/2018 03:28	Mayor's Offic	13 AVENUE betw	Queens	1, 3, 7	109, 7, 90	Film	Feature	United Sta	10002, 11
454914	Shooting Permit	12/06/2018 08:00	12/05/2018 11:00	12/04/2018 03:05	Mayor's Offic	ELDERT STREET b	Brooklyn	4, 5	104, 75, 83	Television	Episodic s	United Sta	11207, 11
454909	Shooting Permit	12/05/2018 08:00	12/05/2018 06:00	12/04/2018 02:45	Mayor's Offic	ELDERT STREET b	Brooklyn	4	83	Television	Episodic s	United Sta	11237
454905	Shooting Permit	12/06/2018 07:00	12/05/2018 10:00	12/04/2018 02:17	Mayor's Offic	35 STREET betwe	Queens	1	114	Television	Cable-epis	United Sta	11101, 11

- What's the most popular street for filming?
- What's the most popular borough?
- How many TV episodes were filmed?

Lecture 7

19 March 2019 32 / 43

	N	YC Ope	enData		Home	e Deta About	- Learn -	Aarts	Contact Us	Blog C	Sign	in.		
Film Perr Permits an street, or a	Tim Permis Time are permis inspection was needed by any section of the collision of the solution of the soluti													
Depetti i	Event?ype i	StortDeteTL i	EndDeteTime 1	EnteredOr + 1	EventAg. 1	ParkingHeld 1	Borea i	Com. i	Pelos. 1	Catego 1	SubC. 1	Count	ZipCo. :	
455063	Shooting Permit	12/06/2018 07:00	12/06/2018 09:00	12/05/2018 12:36	Meyor's Offic	STARRAVENUE h	Queens	2	108	Television	Ephodic s.,	United Sta	11101	
454962	Shaoting Fermit	12/06/2018 07:05	12/06/2018 05:00	12/56/2018 09:11	Mayors Offic	EAGLE STREET DO	Bracklys		64	Television	Episodic s	United Sta	11222	
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 07:00	12/04/2018/05:44	Meyer's Offic	SOUTH ORPORD	Braoklyn	2.6	75,88	Sil Photo	Not Applic	United Sta.,	11217, 11	
454920	Shooting Fermit	12/06/2018 12:05	12/06/2018 11:59	12/04/2018 03:28.	Mayor's Offic	13 AVENUE betw	Queens	1.2.7	108, 7, 90	Film	Feature	United Sta	10002, 11	
454915	Shooting Permit	12/06/2018 08:00	13/06/2018 11:00	12/04/2018 03:05	Mayers offic	ELDERT STREET D.,	Braokys	4.5	104, 25, 83	Television	tpisodic s.,	United Sta.,	11207, 11	
454909	Shooting Permit	12/05/2018 08:00	12/05/2018 05:00	12/04/2018 02:45	Meyor's Offic	ELDERT STREET 6	Drooklyn	4	13	Television	Ephodic s.,	United Sta	11237	
454905	Shooting Fermit	12/06/2018 07:05	13/06/2018 10:00	12/56/2018 02:17	Mayors Offic	26 STREET DOWN.	Queens		114	Television	Cable-spis	United Sta	11101, 11	

• Download the data as a CSV file and store on your computer.

990

	N	YC Ope	Home	Data About	- Learn -	Alerts	Contact Us	Blog C	Sign	in			
Film Perr Permits an street, or a	Tim Permis Tim Pe												
Depetti i	Event?ype i	StortDeteTL i	EndDeteTime 1	EnteredOr + 1	EventAg. i	Parkingheid 1	Borea i	Com. i	Pelos. 1	Catego 1	SubC. 1	Count	ZipCo i
455063	Shooting Permit	12/06/2018 07:00	12/06/2018 09:00	12/05/2018 12:36	Meyor's Offic	STARR. AVENUE 6	Queens	2	108	Television	Ephodic s.,	United Sta	11101
454962	Shooting Fermit	12/06/2018 07:05	13/06/2018 05:00	12/04/2018 09:11	Mayors Offic	DAGLE STREET DO.	Brooklys		64	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 07:00	12/04/2018/05:44	Meyer's Offic	SOUTH ORPORD	Braoklyn	2.6	75,88	Sil Photo	Not Applic	United Sta.,	11217, 11
454920	Shooting Fermit	12/06/2018 12:05	12/06/2018 11:59	12/04/2018 03:28.	Mayor's Offic	13 AVENUE betw	Queens	1.2.7	108, 7, 90	Film	Feature	United Sta	10002, 11
454934	Shaoting Permit	12/06/2018 08:00	13/06/2018 11:00	12/14/2018 03:05	stayers offic	ELDERT STREET D	Braokys	4.5	104, 25, 83	Television	tpisodic s	United Sta	11207, 11
454909	Shooting Permit	12/05/2018 08:00	12/05/2018 05:00	12/04/2018 02:45	Meyor's Offic	ELOERT STREET 6	Drooklyn	4	13	Television	Ephodic s.,	United Sta	11237
454905	Shooting Fermit	12/06/2018 07:05	13/06/2018 10:00	12/56/2018 02:17	Mayors Offic	25 STREET DODAR.	Queens		114	Television	Cable-spis	United Sta	11105, 11

• Download the data as a CSV file and store on your computer.

#### • Python program:

```
#CSci 127 Teaching Staff
#March 2019
#OpenData Film Permits
#Import pandas for reading and analyzing CSV data:
import pandas as pd
csvFile = "filmPermits.csv" #Name of the CSV file
tickets = pd.read_csv(csvFile)#Read in the file to a dataframe
```

CSci 127 (Hunter)

19 March 2019 33 / 43

Sac

イロト 不得 トイヨト イヨト 二日

	N	<b>YC</b> Оре		Hom	Data About	- Learn -	Alerts	Contact Us	Blog C	Sign	in		
Film Perr Permits an street, or a	Tim Permits mening any permits mening and a set of the set of th												
Depetti i	Event?ype i	StortDetell_ i	EndbateTime 1	EnteredOr + 1	EventAg. i	Parkingheid 1	Borea i	Con. i	Polos. 1	Centry. 1	SubC. 1	Count	ZipCo. i
455063	Shooting Permit	12/06/2018 07:08	12/06/2018 19:00	12/05/2018 12:36	Meyor's Offic	STARR. AVENUE 6	Queero	2	108	Television	Ephodic s.,	United Sta	11101
454962	Shaoting Fermit	12/06/2018 07:05	12/06/2018 25:00	12/56/2018 09:11	Mayors Offic	EAGLE STREET DO.	Braoklys		64	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 17:00	12/04/2018/05:44	Meyer's Offic	SOUTH ORPORD	Brooklyn	2.6	75,88	Sil Phote	Not Applic	United Sta.,	11217, 11
454920	Shooting Fermit	12/06/2218 12:00	12/06/2018 11:59	12/04/2018 03:28.	Mayor's Offic	13 AVENUE betw	Queens	1.2.7	108, 7, 90	Film	Feature	United Sta	10002, 11
454955	Shooting Permit	12/06/2018 08:00	13/06/2018 11:00	12/04/2018 03:05	Mayers offic	ELDERT STREET D.,	Braoklys	4.5	104, 25, 83	Television	tpisodic s.,	United Sta.,	11207, 11
454909	Shooting Permit	12/05/2018 08:08	12/05/2018 05:00	12/04/2018 02:45	Meyor's Offic	ELOERT STREET 6	Drooklyn	4	83	Television	Ephodic s.,	United Sta	11237
454905	Shaoting Fermit	12/06/2018 07:05	12/06/2018 10:00	12/56/2018 02:17	Mayors Offic	35 STREET between	Queens		114	Television	Cable-epis	United Sta	11105, 11

• Download the data as a CSV file and store on your computer.

#### • Python program:

```
#CSci 127 Teaching Staff
#March 2019
#OpenData Film Permits
```

```
#Import pandas for reading and analyzing CSV data:
import pandas as pd
csvFile = "filmPermits.csv" #Name of the CSV file
tickets = pd.read_csv(csvFile)#Read in the file to a dataframe
print(tickets) #Print out the dataframe
```

CSci 127 (Hunter)

19 March 2019 34 / 43

	N	YC Ope	Ham	Data About	- Learn -	Alerts	Contact Us	Blog C	Sign	n			
Film Perr Permits an street, or a	Tim Permis III i III IIII Contract when anothing the exclusion case of the property line is obtained, as all IIII IIIII Contract methods are provided page to the second case of the property independence of the property												
Depetto i	Event?ype :	StortDetell_ i	EndDeteTime 1	EnteredOr + 1	EventAg. 1	Parkingheid 1	Borea i	Com. i	Polos. 1	Centry. 1	subC. I	Count. 1	ZipCo i
455063	Shooting Fermit	12/06/2018 07:08	12/06/2018 09:00	12/05/2018 12:35	Mayor's Offic	STARRAVENUE h	Queens	2	108	Television	Ephodic s	United Sta	11101
454962	Shacking Fermit	12/06/2018 07:05	12/06/2018 05:00	12/56/2018 09:11	Mayors Offic	EAGLE STREET DO.	Bracklys		64	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 07:00	12/04/2018/05:44	Meyer's Offic	SOUTH ORPORD	Braoklyn	2.6	75,88	Sil Phote	Not Applic	United Sta.,	11217, 11
454920	Shooting Fermit	12/06/2218 12:00	12/06/2018 11:59	12/04/2018 03:28.	Mayor's Offic	13 AVENUE betw	Queens	1.2.7	108, 7, 90	Film	Feature	United Sta	10002, 11
454954	Shooting Permit	12/06/2018 08:00	13/06/2018 11:00	12/14/2018 03:05	stayers offic	ELDERT STREET D	Braokys	4.5	104, 25, 83	Television	tpisodic s.,	United Sta.,	11202, 11
454909	Shooting Fermit	12/05/2018 08:08	12/05/2018 05:00	12/04/2018 02:45	Meyor's Offic	ELOERT STREET 6	Drooklyn	4	83	Television	Ephodic s.,	United Sta	11237
454905	Shacking Fermit	12/06/2018 07:05	12/06/2018 10:00	12/56/2018 02:17	Mayors Offic	35 STREET between	Queens		114	Television	Cable-epis	United Sta	11101, 11

• Download the data as a CSV file and store on your computer.

#### • Python program:

```
#CSci 127 Teaching Staff
#March 2019
#OpenData Film Permits
```

```
#Import pandas for reading and analyzing CSV data:
import pandas as pd
csvFile = "filmPermits.csv" #Name of the CSV file
tickets = pd.read_csv(csvFile)#Read in the file to a dataframe
print(tickets) #Print out the dataframe
print(tickets["ParkingHeld"]) #Print out streets (multiple times)
```

CSci 127 (Hunter)

19 March 2019 35 / 43

	N	<b>VC</b> Ope	enData		Home	Data About	- Learn -	Alerts	Contact Us	Blog	Signi	•	
Film Perr Permits are street, or a	nits generally require park. See http://ww	d when asserting th wrl.nyc.gow'site/inc	te exclusive use of	city property, like permit-required.p	a sidewalk, a 🗉 age	1		D.	f ⇒ ore Views	ili B 🗆	Q. Find in ize Deport	this Datase Discuss	c nbed About
Depetto i	Event?ype :	StortDetell_ i	EndDeteTime 1	EnteredOr + 1	EventAg. :	Parkingheid 1	Borea i	Con. i	Polos. 1	Centry. 1	SubC. 1	Count. 1	ZipCo. :
455063	Shooting Fermit	12/06/2018 07:08	12/06/2018 09:00	12/05/2018 12:36	Meyor's Offic	STARR. AVENUE 6	Queero	2	108	Television	tphodic s	United Sta	11101
454962	Shacking Fermit	12/06/2018 07:05	12/06/2018 05:00	12/56/2018 09:11	Mayors Offic	EAGLE STREET DO.	Braoklys		64	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 07:00	12/04/2018/05:44	Meyer's Offic.	SOUTH ORPORD	Brooklyn	2.6	75,88	Sil Phote	Not Applic	United Sta	11217, 11
454920	Shooting Fermit	12/06/2218 12:00	12/06/2018 11:59	12/04/2018 03:28.	Mayor's Offic	13 AVENUE betw	Queens	1.2.7	108, 7, 90	Film	Feature	United Sta	10002,11
454915	Shaoting Permit	12/06/2018 08:00	13/06/2018 11:00	12/04/2018 03:05	Mayers offic	ELDERT STREET D.,	Braoklys	4.5	104, 25, 83	Television	tpisodic s.,	United Sta	11207, 11
454909	Shooting Fermit	12/05/2018 08:00	12/05/2018 05:00	12/04/2018 02:45	Mayor's Offic	ELDERT STREET 6	Drooklyn	4	83	Television	Ephodic s.,	United Sta	11237
454905	Shacking Fermit	12/06/2018 07:05	12/06/2018 10:00	12/56/2018 02:17	Mayors Offic	35 STREET between	Queens		114	Television	Cable-epit	United Sta	11101, 11

• Download the data as a CSV file and store on your computer.

#### • Python program:

```
#CSci 127 Teaching Staff
#March 2019
#OpenData Film Permits
#Import pandas for reading and analyzing CSV data:
import pandas as pd
csvFile = "filmPermits.csv" #Name of the CSV file
tickets = pd.read_csv(csvFile)#Read in the file to a dataframe
print(tickets)  #Print out streets (multiple times)
print(tickets["ParkingHeld"].value_counts()) #Print out streets & number of times used
```

19 March 2019 36 / 43

	N	<b>VC</b> Ope	enData		Home	Data About	- Learn -	Alerts	Contact Us	Blog	Signi	•	
Film Perr Permits are street, or a	nits generally require park. See http://ww	d when asserting th wrl.nyc.gow'site/inc	te exclusive use of	city property, like permit-required.p	a sidewalk, a 🗉 age	1		D.	f ⇒ ore Views	ili B 🗆	Q. Find in ize Deport	this Datase Discuss	c nbed About
Depetto i	Event?ype :	StortDetell_ i	EndDeteTime 1	EnteredOr + 1	EventAg. :	Parkingheid 1	Borea i	Com. i	Polos. 1	Centry. 1	SubC. 1	Count. 1	ZipCo. :
455063	Shooting Fermit	12/06/2018 07:08	12/06/2018 09:00	12/05/2018 12:36	Meyor's Offic	STARR. AVENUE 6	Queero	2	108	Television	tphodic s	United Sta	11101
454962	Shacking Fermit	12/06/2018 07:05	12/06/2018 05:00	12/56/2018 09:11	Mayors Offic	EAGLE STREET DO.	Braoklys		64	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/06/2018 07:00	12/04/2018/05:44	Meyer's Offic.	SOUTH ORPORD	Brooklyn	2.6	75,88	Sil Phote	Not Applic	United Sta	11217, 11
454920	Shooting Fermit	12/06/2218 12:00	12/06/2018 11:59	12/04/2018 03:28.	Mayor's Offic	13 AVENUE betw	Queens	1.2.7	108, 7, 90	Film	Feature	United Sta	10002,11
454915	Shaoting Permit	12/06/2018 08:00	13/06/2018 11:00	12/04/2018 03:05	Mayers offic	ELDERT STREET D.,	Braoklys	4.5	104, 25, 83	Television	tpisodic s.,	United Sta	11207, 11
454909	Shooting Fermit	12/05/2018 08:00	12/05/2018 05:00	12/04/2018 02:45	Mayor's Offic	ELDERT STREET 6	Drooklyn	4	83	Television	Ephodic s.,	United Sta	11237
454905	Shacking Fermit	12/06/2018 07:05	12/06/2018 10:00	12/56/2018 02:17	Mayors Offic	35 STREET between	Queens		114	Television	Cable-epit	United Sta	11101, 11

• Download the data as a CSV file and store on your computer.

#### • Python program:

```
#CSci 127 Teaching Staff
#March 2019
#OpenData Film Permits
#Import pandas for reading and analyzing CSV data:
import pandas as pd
csvFile = "filmPermits.csv" #Name of the CSV file
tickets = pd.read_csv(csvFile)#Read in the file to a dataframe
print(tickets) #Print out the dataframe
print(tickets["ParkingHeld"]) #Print out streets (multiple times)
print(tickets["ParkingHeld"].value_counts()) #Print out streets & number of times used
print(tickets["ParkingHeld"].value_counts()):10]) #Print 10 most popular
```

Sac

NYC OpenData

Home Data About -> Learn -> Alerts Contact Us Blog Q Sign In

Film Permits     B     #													t nbed About
EventID :	EventType i	StartDateTi	EndDateTime	EnteredOn 4	EventAg i	ParkingHeld i	Borou i	Com i	Police i	Categ i	SubC i	Count i	ZipCo i
455063	Shooting Permit	12/06/2018 07:00	12/06/2018 09:00	12/05/2018 12:36	Mayor's Offic	STARR AVENUE b	Queens	2	108	Television	Episodic s	United Sta	11101
454967	Shooting Permit	12/06/2018 07:00	12/06/2018 05:00	12/04/2018 09:11	Mayor's Offic	EAGLE STREET be	Brooklyn	1	94	Television	Episodic s	United Sta	11222
454941	Shooting Permit	12/06/2018 07:00	12/05/2018 07:00	12/04/2018 05:44	Mayor's Offic	SOUTH OXFORD	Brooklyn	2, 6	76, 88	Still Photo	Not Applic	United Sta	11217, 11
454920	Shooting Permit	12/06/2018 10:00	12/05/2018 11:59	12/04/2018 03:28	Mayor's Offic	13 AVENUE betw	Queens	1, 3, 7	109, 7, 90	Film	Feature	United Sta	10002, 11
454914	Shooting Permit	12/06/2018 08:00	12/05/2018 11:00	12/04/2018 03:05	Mayor's Offic	ELDERT STREET b	Brooklyn	4, 5	104, 75, 83	Television	Episodic s	United Sta	11207, 11
454909	Shooting Permit	12/05/2018 08:00	12/05/2018 06:00	12/04/2018 02:45	Mayor's Offic	ELDERT STREET b	Brooklyn	4	83	Television	Episodic s	United Sta	11237
454905	Shooting Permit	12/06/2018 07:00	12/05/2018 10:00	12/04/2018 02:17	Mayor's Offic	35 STREET betwe	Queens	1	114	Television	Cable-epis	United Sta	11101, 11

#### Can approach the other questions in the same way:

- What's the most popular street for filming?
- What's the most popular borough?
- How many TV episodes were filmed?

CSci 127 (Hunter)

イロト 不得 トイヨト イヨト ヨー のくや



Design an algorithm that finds the closest collision. (Sample NYC OpenData collision data file on back of lecture slip.)

CSci 127 (Hunter)

Lecture 7

19 March 2019 39 / 43

900

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

• Create a "To Do" list of what your program has to accomplish.

200

イロト 不得 トイヨト イヨト 二日

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.
- Don't worry if you don't know how to do all the items you write down.

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.
- Don't worry if you don't know how to do all the items you write down.

• Example:

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.
- Don't worry if you don't know how to do all the items you write down.
- Example:
  - 1 Find data set (great place to look: NYC OpenData).
Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.
- Don't worry if you don't know how to do all the items you write down.
- Example:
  - I Find data set (great place to look: NYC OpenData).
  - 2 Ask user for current location.

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.
- Don't worry if you don't know how to do all the items you write down.
- Example:
  - I Find data set (great place to look: NYC OpenData).
  - 2 Ask user for current location.
  - ③ Open up the CSV file.

イロト 不得 トイヨト イヨト ヨー のくや

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.
- Don't worry if you don't know how to do all the items you write down.
- Example:
  - I Find data set (great place to look: NYC OpenData).
  - 2 Ask user for current location.
  - ③ Open up the CSV file.
  - 4 Check distance to each to user's location.

▲□▶ ▲□▶ ▲三▶ ▲三▶ 三三 ろの⊙

Design an algorithm that uses NYC OpenData collision data and computes the closest collision to the location the user provides.

How to approach this:

- Create a "To Do" list of what your program has to accomplish.
- Read through the problem, and break it into "To Do" items.
- Don't worry if you don't know how to do all the items you write down.
- Example:
  - I Find data set (great place to look: NYC OpenData).
  - 2 Ask user for current location.
  - ③ Open up the CSV file.
  - ④ Check distance to each to user's location.
  - 5 Save the location with the smallest distance.

イロト 不得 トイヨト イヨト ヨー のくや

• On lecture slip, write down a topic you wish we had spent more time (and why).





= nar





• On lecture slip, write down a topic you wish we had spent more time (and why).

• Functions are a way to break code into pieces, that can be easily reused.

3

Sac





- On lecture slip, write down a topic you wish we had spent more time (and why).
- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis:





- On lecture slip, write down a topic you wish we had spent more time (and why).
- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")

CSci 127 (Hunter)

19 March 2019 41 / 43





- On lecture slip, write down a topic you wish we had spent more time (and why).
- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or define your own functions,





- On lecture slip, write down a topic you wish we had spent more time (and why).
- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or define your own functions, which are stored, until invoked or called.





- On lecture slip, write down a topic you wish we had spent more time (and why).
- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or **define** your own functions, which are stored, until invoked or called.
- Accessing Formatted Data: NYC OpenData





- On lecture slip, write down a topic you wish we had spent more time (and why).
- Functions are a way to break code into pieces, that can be easily reused.
- You call or invoke a function by typing its name, followed by any inputs, surrounded by parenthesis: Example: print("Hello", "World")
- Can write, or **define** your own functions, which are stored, until invoked or called.
- Accessing Formatted Data: NYC OpenData
- Pass your lecture slips to the aisles for the UTAs to collect.



• Since you must pass the final exam to pass the course, we end every lecture with final exam review.



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - write as much you can for 60 seconds;



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.

CSci 127 (Hunter)

19 March 2019 42 / 43



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.
- Past exams are on the webpage (under Final Exam Information).

CSci 127 (Hunter)



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.
- Past exams are on the webpage (under Final Exam Information).
- Theme: Functions!

CSci 127 (Hunter)

19 March 2019 42 / 43

イロト 不得下 イヨト イヨト



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.
- Past exams are on the webpage (under Final Exam Information).
- Theme: Functions! Starting with Summer 18, #4.

CSci 127 (Hunter)

## Writing Boards



• Return writing boards as you leave...

CSci 127 (Hunter)

Lecture 7

19 March 2019 43 / 43

3

990

<ロト <回ト < 回ト < 回ト