CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

CSci 127 (Hunter)

Lecture 6

3 March 2020 1 / 39

990

Announcements



• Each lecture includes a survey of computing research and tech in NYC.

Image: A matrix and a matrix

Today: Prof. Kelle Cruz (Astrophysics)

From lecture slips & recitation sections.

990

イロト イロト イモト イモト 二日

From lecture slips & recitation sections.

• Could you spend more time on circuits/logical expressions/truth tables/decisions?

996

From lecture slips & recitation sections.

• Could you spend more time on circuits/logical expressions/truth tables/decisions? We will start with that today, and more in the following weeks.

200

From lecture slips & recitation sections.

- Could you spend more time on circuits/logical expressions/truth tables/decisions? We will start with that today, and more in the following weeks.
- I still don't get indices and the brackets. Could you spend more time on that?

Sac

From lecture slips & recitation sections.

- Could you spend more time on circuits/logical expressions/truth tables/decisions? We will start with that today, and more in the following weeks.
- I still don't get indices and the brackets. Could you spend more time on that? *Yes, we will, since*
 - 1) it's fundamental, and
 - 2) the same ideas are used for accessing formatted data (today's topic).

From lecture slips & recitation sections.

- Could you spend more time on circuits/logical expressions/truth tables/decisions? We will start with that today, and more in the following weeks.
- I still don't get indices and the brackets. Could you spend more time on that? *Yes, we will, since*
 - 1) it's fundamental, and
 - 2) the same ideas are used for accessing formatted data (today's topic).
- I still don't get what is meant by input?

From lecture slips & recitation sections.

- Could you spend more time on circuits/logical expressions/truth tables/decisions? We will start with that today, and more in the following weeks.
- I still don't get indices and the brackets. Could you spend more time on that? *Yes, we will, since*
 - 1) it's fundamental, and
 - 2) the same ideas are used for accessing formatted data (today's topic).
- I still don't get what is meant by input? Input is data provided to a program each time it runs, it may change at each run. In this course we have used the input() function.

From lecture slips & recitation sections.

- Could you spend more time on circuits/logical expressions/truth tables/decisions? We will start with that today, and more in the following weeks.
- I still don't get indices and the brackets. Could you spend more time on that? Yes, we will, since
 1) it's fundamental, and
 - 2) the same ideas are used for accessing formatted data (today's topic).
- I still don't get what is meant by input? Input is data provided to a program each time it runs, it may change at each run. In this course we have used the input() function.
- Should I have received email for this course?

From lecture slips & recitation sections.

- Could you spend more time on circuits/logical expressions/truth tables/decisions? We will start with that today, and more in the following weeks.
- I still don't get indices and the brackets. Could you spend more time on that? Yes, we will, since
 1) it's fundamental, and
 - 2) the same ideas are used for accessing formatted data (today's topic).
- I still don't get what is meant by input? Input is data provided to a program each time it runs, it may change at each run. In this course we have used the input() function.
- Should I have received email for this course? Absolutely!!! We often send important communication by email. If you have not been receiving email from us weekly, please check your spam folder.

CSci 127 (Hunter)

Today's Topics



- Recap: Logical Expressions & Circuits
- Design: Cropping Images
- Accessing Formatted Data
- CS Survey: Astrophysics and astropy

200

Today's Topics



• Recap: Logical Expressions & Circuits

- Design: Cropping Images
- Accessing Formatted Data
- CS Survey: Astrophysics and astropy

200

Recap: Logical Operators

and

in1		in2	returns:
False	and	False	False
False	and	True	False
True	and	False	False
True	and	True	True

990

イロト イロト イモト イモト 二日

Recap: Logical Operators

and

in1		in2	returns:
False	and	False	False
False	and	True	False
True	and	False	False
True	and	True	True

or

in1		in2	returns:
False	or	False	False
False	or	True	True
True	or	False	True
True	or	True	True

999

<ロト < 回 ト < 回 ト < 回 ト < 回 ト - 回</p>

Recap: Logical Operators

and

in1		in2	returns:
False	and	False	False
False	and	True	False
True	and	False	False
True	and	True	True

or

in1		in2	returns:
False	or	False	False
False	or	True	True
True	or	False	True
True	or	True	True

not

	in1	returns:
not	False	True
not	True	False

1 3 March 2020 6/39

990



• Each logical operator (and, or, & not) can be used to join together expressions.

Image: A match a ma

200



• Each logical operator (and, or, & not) can be used to join together expressions.

Example: in1 and in2



 Each logical operator (and, or, & not) can be used to join together expressions.

Example: in1 and in2

 Each logical operator (and, or, & not) has a corresponding logical circuit that can be used to join together inputs.



 Each logical operator (and, or, & not) can be used to join together expressions.

Example: in1 and in2

 Each logical operator (and, or, & not) has a corresponding logical circuit that can be used to join together inputs.

Examples: Logical Circuit



CSci 127 (Hunter)

E 3 March 2020 8 / 39

990

< ロ ト < 回 ト < 三 ト < 三 ト</p>

Examples: Logical Circuit



(in1 and in2) and in3

CSci 127 (Hunter)

Lecture 6

<ロト < 回 ト < 巨 ト < 三 ト < 三 ト ミ の Q () 3 March 2020 8 / 39

More Circuit Examples

Examples from last lecture:



Draw a circuit that corresponds to each logical expression:

- o not(in1 or in2)
- (in1 or in2) and (in1 or in3)
- o (not(in1 and not in2)) or (in1 and (in2 and in3))

イロト イヨト イヨト イ

In Pairs or Triples:

Predict what the code will do:

```
x = 6
                      y = x \% 4
                      w = y^{**3}
                      z = w // 2
                      print(x,y,w,z)
                      x, y = y, w
                      print(x,y,w,z)
                      x = y / 2
print(x,y,w,z)
                     sports = ["Field Hockey", "Swimming", "Water Polo"]
                     mess = "Qoauxca BrletRce crcx qvBnga ocUxk"
                     result =
                     for i in range(len(mess)):
                                                 if i % 3 == 0:
                                                                              print(mess[i])
                                                                               result = result + mess[i]
                   print(sports[1], result)
 < □ > < □ > < 豆 > < 豆 > < 豆 > < 豆 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □
```

Python Tutor

```
x = 6
y = x % 4
w = y**3
z = w // 2
print(x,y,w,z)
x,y = y,w
print(x,y,w,z)
x = y / 2
print(x,y,w,z)
```

(Demo with pythonTutor)

200

Today's Topics



- Recap: Logical Expressions & Circuits
- Design: Cropping Images
- Accessing Formatted Data
- CS Survey: Astrophysics and astropy

3

Sar

In Pairs or Triples: Design Question

From Final Exam, Fall 2017, V4, #6.



Design an algorithm that reads in an image and displays the lower left corner of the image.

CSci 127 (Hunter)

Lecture 6

3 March 2020 13 / 39

In Pairs or Triples: Design Question

From Final Exam, Fall 2017, V4, #6.



Design an algorithm that reads in an image and displays the lower left corner of the image.

Input: Output: Process: (Brainstorm for a "To Do" list to accomplish this.)

CSci 127 (Hunter)

Lecture 6

3 March 2020 13 / 39

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

<ロト < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

How to approach this:

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

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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.

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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.

Jac.

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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:

Jac.

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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:
 - Import libraries.

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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:
 - Import libraries.
 - 2 Ask user for an image name.

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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:
 - Import libraries.
 - 2 Ask user for an image name.
 - ③ Read in image.

Sac
Design Question

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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 Import libraries.
- 2 Ask user for an image name.
- ③ Read in image.
- ④ Figure out size of image.

イロト イボト イヨト 一日

Design Question

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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 Import libraries.
- 2 Ask user for an image name.
- 3 Read in image.
- ④ Figure out size of image.
- 5 Make a new image that's half the height and half the width.

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 ◇◇◇

Design Question

Design a program that asks the user for an image and then display the upper left quarter of the image. (First, design the pseudocode, and if time, expand to a Python program.)

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 Import libraries.
- 2 Ask user for an image name.
- ③ Read in image.
- ④ Figure out size of image.
- 5 Make a new image that's half the height and half the width.
- O Display the new image.

CSci 127 (Hunter)

3 March 2020 14 / 39

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 ◇◇◇



Э

900



> import matplotlib.pyplot as plt import numpy as np

900



> import matplotlib.pyplot as plt import numpy as np

Ask user for an image name. 2

Sar



import matplotlib.pyplot as plt import numpy as np

Ask user for an image name.

inF = input('Enter file name: ')



import matplotlib.pyplot as plt import numpy as np

- 2 Ask user for an image name. inF = input('Enter file name: ')
- ③ Read in image.



import matplotlib.pyplot as plt import numpy as np

2 Ask user for an image name.

inF = input('Enter file name: ')

③ Read in image.

img = plt.imread(inF) #Read in image from inF



import matplotlib.pyplot as plt import numpy as np

2 Ask user for an image name.

inF = input('Enter file name: ')

③ Read in image.

img = plt.imread(inF) #Read in image from inF

4 Figure out size of image.

In Pairs or Triples: Design Question HUENTER The City University of New York The City University of New York

Import libraries.

import matplotlib.pyplot as plt import numpy as np

2 Ask user for an image name.

inF = input('Enter file name: ')

③ Read in image.

img = plt.imread(inF) #Read in image from inF

④ Figure out size of image.

height = img.shape[0] #Get height
width = img.shape[1] #Get width

In Pairs or Triples: Design Question HUONTER The City University of New York The City University of New York

Import libraries.

import matplotlib.pyplot as plt import numpy as np

2 Ask user for an image name.

inF = input('Enter file name: ')

③ Read in image.

img = plt.imread(inF) #Read in image from inF

- ④ Figure out size of image. height = img.shape[0] #Get height width = img.shape[1] #Get width
- Make a new image that's half the height and half the width.

< ロト < 同ト < ヨト < ヨト

In Pairs or Triples: Design Question HUCNTER The City University of New York Import libraries.

- import matplotlib.pyplot as plt import numpy as np
- 2 Ask user for an image name.

inF = input('Enter file name: ')

③ Read in image.

img = plt.imread(inF) #Read in image from inF

④ Figure out size of image.

height = img.shape[0] #Get height

width = img.shape[1] #Get width

Make a new image that's half the height and half the width. img2 = img[height//2:, :width//2] #Crop to lower left corner

Sac

In Pairs or Triples: Design Question The City University of New York The City Univers Import libraries. 1 import matplotlib.pyplot as plt import numpy as np 2 Ask user for an image name. inF = input('Enter file name: ') ③ Read in image. img = plt.imread(inF) #Read in image from inF ④ Figure out size of image. height = img.shape[0] #Get height width = img.shape[1] #Get width Make a new image that's half the height and half the width. 5 img2 = img[height//2:, :width//2] #Crop to lower left corner

6 Display the new image. plt.imshow(img2) #Load our new image into pyplot plt.show() #Show the image (waits until closed to continue) CSci 127 (Hunter)
Lecture 6
3 March 2020 15/39

Today's Topics



- Recap: Logical Expressions & Circuits
- Design: Cropping Images
- Accessing Formatted Data
- CS Survey: Astrophysics and astropy

3

	1	Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

• Common to have data structured in a spread sheet.

996

イロト イロト イモト イモト 一日

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Common to have data structured in a spread sheet.
- In the example above, we have the first line that says "Undergraduate".

3

Sac

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Common to have data structured in a spread sheet.
- In the example above, we have the first line that says "Undergraduate".
- Next line has the titles for the columns.

CSci 127 (Hunter)

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Common to have data structured in a spread sheet.
- In the example above, we have the first line that says "Undergraduate".
- Next line has the titles for the columns.
- Subsequent lines have a college and attributes about the college.

CSci 127 (Hunter)

Lecture 6

3 March 2020 17 / 39

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Common to have data structured in a spread sheet.
- In the example above, we have the first line that says "Undergraduate".
- Next line has the titles for the columns.
- Subsequent lines have a college and attributes about the college.
- Python has several ways to read in such data.

CSci 127 (Hunter)

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Common to have data structured in a spread sheet.
- In the example above, we have the first line that says "Undergraduate".
- Next line has the titles for the columns.
- Subsequent lines have a college and attributes about the college.
- Python has several ways to read in such data.
- We will use the popular Python Data Analysis Library (Pandas). CSci 127 (Hunter) Lecture 6



• We will use the popular Python Data Analysis Library (Pandas).

CSci 127 (Hunter)

Lecture 6

E ► 4 E ► E ∽ Q C 3 March 2020 18 / 39



- We will use the popular Python Data Analysis Library (Pandas).
- Open source and freely available (part of anaconda distribution).



- We will use the popular Python Data Analysis Library (Pandas).
- Open source and freely available (part of anaconda distribution).
- Already loaded on the machines in 1001E North.



- We will use the popular Python Data Analysis Library (Pandas).
- Open source and freely available (part of anaconda distribution).
- Already loaded on the machines in 1001E North.
- See end of Lab 6 for directions on downloading it to your home machine.

Image: A match a ma



- We will use the popular Python Data Analysis Library (Pandas).
- Open source and freely available (part of anaconda distribution).
- Already loaded on the machines in 1001E North.
- See end of Lab 6 for directions on downloading it to your home machine.
- To use, add to the top of your file:

import pandas as pd

CSci 127 (Hunter)

< ロト < 同ト < ヨト < ヨト

	1	Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

• Excel .xls files have much extra formatting.

990

	1	Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Excel .xls files have much extra formatting.
- The text file version is called **CSV** for comma separated values.

CSci 127 (Hunter)

Sac

イロト イポト イヨト イヨト 二日

	1	Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Excel .xls files have much extra formatting.
- The text file version is called **CSV** for comma separated values.
- Each row is a line in the file.

CSci 127 (Hunter)

3

Sac

	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- Excel .xls files have much extra formatting.
- The text file version is called **CSV** for comma separated values.
- Each row is a line in the file.
- Columns are separated by commas on each line.

CSci 127 (Hunter)

3

```
Source: https://en.wikipedia.org/wiki/Demographics of New York City.....
All population figures are consistent with present-day boundaries.,,,,,,
First census after the consolidation of the five boroughs,.....
.....
Year, Manhattan, Brooklyn, Oueens, Bronx, Staten Island, Total
1698,4937,2017,...727,7681
1771,21863,3623,.,2847,28423
1790,33131,4549,6159,1781,3827,49447
1800,60515,5740,6642,1755,4563,79215
1810,96373,8303,7444,2267,5347,119734
1820, 123706, 11187, 8246, 2782, 6135, 152056
1830,202589,20535,9049,3023,7082,242278
1840,312710,47613,14480,5346,10965,391114
1850, 515547, 138882, 18593, 8032, 15061, 696115
1860,813669,279122,32903,23593,25492,1174779
1870,942292,419921,45468,37393,33029,1478103
1880, 1164673, 599495, 56559, 51980, 38991, 1911698
1890,1441216,838547,87050,88908,51693,2507414
1900,1850093,1166582,152999,200507,67021,3437202
1910,2331542,1634351,284041,430980,85969,4766883
1920, 2284103, 2018356, 469042, 732016, 116531, 5620048
1930, 1867312, 2560401, 1079129, 1265258, 158346, 6930446
1940, 1889924, 2698285, 1297634, 1394711, 174441, 7454995
1950, 1960101, 2738175, 1550849, 1451277, 191555, 7891957
1960, 1698281, 2627319, 1809578, 1424815, 221991, 7781984
1970, 1539233, 2602012, 1986473, 1471701, 295443, 7894862
1980, 1428285, 2230936, 1891325, 1168972, 352121, 7071639
1990, 1487536, 2300664, 1951598, 1203789, 378977, 7322564
2000,1537195,2465326,2229379,1332650,443728,8008278
2010, 1585873, 2504700, 2230722, 1385108, 468730, 8175133
2015, 1644518, 2636735, 2339150, 1455444, 474558, 8550405
```

nycHistPop.csv

CSci 127 (Hunter)

3 March 2020 20 / 39

Sac

	1	Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

• To read in a CSV file: myVar = pd.read_csv("myFile.csv")

900

		Undergraduate	1
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- To read in a CSV file: myVar = pd.read_csv("myFile.csv")
- Pandas has its own type, DataFrame, that is perfect for holding a sheet of data.

CSci 127 (Hunter)

Lecture 6

3 March 2020 21 / 39

3

	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- To read in a CSV file: myVar = pd.read_csv("myFile.csv")
- Pandas has its own type, DataFrame, that is perfect for holding a sheet of data.
- Often abbreviated: df.

CSci 127 (Hunter)

3

	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

- To read in a CSV file: myVar = pd.read_csv("myFile.csv")
- Pandas has its own type, DataFrame, that is perfect for holding a sheet of data.
- Often abbreviated: df.
- It also has Series, that is perfect for holding a row or column of data.

CSci 127 (Hunter)

Example: Reading in CSV Files

Source: https://en.wikipedia.org/wiki/Demographice_of_New_York_City,,,,, All population figures are consistent with present-day boundaries.,,,,, First census after the consolidation of the five boroughs,,,,,

Year, Manhattan, Brooklyn, Queens, Bronx, Staten Island, Total 1698, 4937, 2017, ... 727, 7681 1771,21863,3623,,,2847,28423 1790, 33131, 4549, 6159, 1781, 3827, 49447 1800,60515,5740,6642,1755,4563,79215 1810,96373,8303,7444,2267,5347,119734 1820, 123706, 11187, 8246, 2782, 6135, 152056 1830, 202589, 20535, 9049, 3023, 7082, 242278 1840, 312710, 47613, 14480, 5346, 10965, 391114 1850,515547,138882,18593,8032,15061,696115 1860,813669,279122,32903,23593,25492,1174779 1870,942292,419921,45468,37393,33029,1478103 1880, 1164673, 599495, 56559, 51980, 38991, 1911698 1890,1441216,838547,87050,88908,51693,2507414 1900, 1850093, 1166582, 152999, 200507, 67021, 3437202 1910,2331542,1634351,284041,430980,85969,4766883 1920, 2284103, 2018356, 469042, 732016, 116531, 5620048 1930, 1867312, 2560401, 1079129, 1265258, 158346, 6930446 1940,1889924,2698285,1297634,1394711,174441,7454995 1950, 1960101, 2738175, 1550849, 1451277, 191555, 7891957 1960, 1698281, 2627319, 1809578, 1424815, 221991, 7781984 1970, 1539233, 2602012, 1986473, 1471701, 295443, 7894862 1980, 1428285, 2230936, 1891325, 1168972, 352121, 7071639 1990,1487536,2300664,1951598,1203789,378977,7322564 2000,1537195,2465326,2229379,1332650,443728,8008278 2010, 1585873, 2504700, 2230722, 1385108, 468730, 8175133 2015,1644518,2636735,2339150,1455444,474558,8550405

nycHistPop.csv

In Lab 6

CSci 127 (Hunter)

3 March 2020 22 / 39

<□▶ < □▶ < □▶ < 三▶ < 三▶ = 三 のへぐ
import matplotlib.pyplot as plt import pandas as pd

Source: https://en.wikipedia.org/wiki/Demographics_of_Mew_York_City,..., All population figures are consistent with present-day boundaries...... First census after the consolidation of the five boroughs,...,

Year, Manhattan, Brooklyn, Queens, Bronx, Staten Island, Total 1698, 4937, 2017, ... 727, 7681 1771,21863,3623,,,2847,28423 1790.33131.4549.6159.1781.3827.49447 1800,60515,5740,6642,1755,4563,79215 1810,96373,8303,7444,2267,5347,119734 1820, 123706, 11187, 8246, 2782, 6135, 152056 1830, 202589, 20535, 9049, 3023, 7082, 242278 1840, 312710, 47613, 14480, 5346, 10965, 391114 1850,515547,138882,18593,8032,15061,696115 1860,813669,279122,32903,23593,25492,1174779 1870,942292,419921,45468,37393,33029,1478103 1880, 1164673, 599495, 56559, 51980, 38991, 1911698 1890,1441216,838547,87050,88908,51693,2507414 1900, 1850093, 1166582, 152999, 200507, 67021, 343720 1910,2331542,1634351,284041,430980,85969,4766883 1920, 2284103, 2018356, 469042, 732016, 116531, 5620048 1930, 1867312, 2560401, 1079129, 1265258, 158346, 6930446 1940,1889924,2698285,1297634,1394711,174441,7454995 1950, 1960101, 2738175, 1550849, 1451277, 191555, 7891957 1960, 1698281, 2627319, 1809578, 1424815, 221991, 7781984 1970, 1539233, 2602012, 1986473, 1471701, 295443, 7894862 1980, 1428285, 2230936, 1891325, 1168972, 352121, 7071639 1990,1487536,2300664,1951598,1203789,378977,7322564 2000,1537195,2465326,2229379,1332650,443728,8008278 2010, 1585873, 2504700, 2230722, 1385108, 468730, 8175133 2015,1644518,2636735,2339150,1455444,474558,8550405

nycHistPop.csv

In Lab 6

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

import matplotlib.pyplot as plt import pandas as pd

pop = pd.read_csv('nycHistPop.csv', skiprows=5)

Source: https://en.wikipedia.org/wiki/Demographics_of_New_York_City,..., All population figures are consistent with present-day boundaries...... Pirst census after the consolidation of the five boroughs,....,

Year, Manhattan, Brooklyn, Queens, Bronx, Staten Island, Total 1698, 4937, 2017, ... 727, 7681 1771,21863,3623,,,2847,28423 1790.33131.4549.6159.1781.3827.49447 1800,60515,5740,6642,1755,4563,79215 1810,96373,8303,7444,2267,5347,119734 1820, 123706, 11187, 8246, 2782, 6135, 152056 1830, 202589, 20535, 9049, 3023, 7082, 242278 1840, 312710, 47613, 14480, 5346, 10965, 391114 1850,515547,138882,18593,8032,15061,696115 1860,813669,279122,32903,23593,25492,1174779 1870,942292,419921,45468,37393,33029,1478103 1880, 1164673, 599495, 56559, 51980, 38991, 1911698 1890,1441216,838547,87050,88908,51693,2507414 1900, 1850093, 1166582, 152999, 200507, 67021, 343720 1910,2331542,1634351,284041,430980,85969,4766883 1920, 2284103, 2018356, 469042, 732016, 116531, 5620048 1930, 1867312, 2560401, 1079129, 1265258, 158346, 6930446 1940,1889924,2698285,1297634,1394711,174441,7454995 1950, 1960101, 2738175, 1550849, 1451277, 191555, 7891957 1960, 1698281, 2627319, 1809578, 1424815, 221991, 7781984 1970, 1539233, 2602012, 1986473, 1471701, 295443, 7894862 1980, 1428285, 2230936, 1891325, 1168972, 352121, 7071639 1990,1487536,2300664,1951598,1203789,378977,7322564 2000,1537195,2465326,2229379,1332650,443728,8008278 2010, 1585873, 2504700, 2230722, 1385108, 468730, 8175133 2015,1644518,2636735,2339150,1455444,474558,8550405

nycHistPop.csv

In Lab 6

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

import matplotlib.pyplot as plt import pandas as pd

pop = pd.read_csv('nycHistPop.csv', skiprows=5)

pop.plot(x="Year")

plt.show()

Source: https://en.wikipedia.org/wiki/Demographics_of_Mew_York_City,..., All population figures are consistent with present-day boundaries...... First census after the consolidation of the five boroughs,...,

Year, Manhattan, Brooklyn, Queens, Bronx, Staten Island, Total 1698, 4937, 2017, ... 727, 7681 1771,21863,3623,,,2847,28423 1790.33131.4549.6159.1781.3827.49447 1800,60515,5740,6642,1755,4563,79215 1810,96373,8303,7444,2267,5347,119734 1820, 123706, 11187, 8246, 2782, 6135, 152056 1830, 202589, 20535, 9049, 3023, 7082, 242278 1840, 312710, 47613, 14480, 5346, 10965, 391114 1850,515547,138882,18593,8032,15061,696115 1860,813669,279122,32903,23593,25492,1174779 1870,942292,419921,45468,37393,33029,1478103 1880, 1164673, 599495, 56559, 51980, 38991, 1911698 1890,1441216,838547,87050,88908,51693,2507414 1900, 1850093, 1166582, 152999, 200507, 67021, 343720 1910,2331542,1634351,284041,430980,85969,4766883 1920, 2284103, 2018356, 469042, 732016, 116531, 562004 1930, 1867312, 2560401, 1079129, 1265258, 158346, 6930446 1940,1889924,2698285,1297634,1394711,174441,7454995 1950, 1960101, 2738175, 1550849, 1451277, 191555, 7891957 1960, 1698281, 2627319, 1809578, 1424815, 221991, 7781984 1970, 1539233, 2602012, 1986473, 1471701, 295443, 7894862 1980, 1428285, 2230936, 1891325, 1168972, 352121, 7071639 1990,1487536,2300664,1951598,1203789,378977,7322564 2000,1537195,2465326,2229379,1332650,443728,8008278 2010, 1585873, 2504700, 2230722, 1385108, 468730, 8175133 2015,1644518,2636735,2339150,1455444,474558,8550405

nycHistPop.csv

In Lab 6

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

import matplotlib.pyplot as plt import pandas as pd

pop = pd.read_csv('nycHistPop.csv', skiprows=5)

Year, Manhattan, Brooklyn, Queens, Bronx, Staten Island, Total 1698, 4937, 2017, ... 727, 7681 1771,21863,3623,,,2847,28423 1790.33131.4549.6159.1781.3827.49447 1800,60515,5740,6642,1755,4563,79215 1810,96373,8303,7444,2267,5347,119734 1820,123706,11187,8246,2782,6135,152056 1830, 202589, 20535, 9049, 3023, 7082, 242278 1840, 312710, 47613, 14480, 5346, 10965, 391114 1850,515547,138882,18593,8032,15061,696115 1860,813669,279122,32903,23593,25492,1174779 1870,942292,419921,45468,37393,33029,1478103 1880, 1164673, 599495, 56559, 51980, 38991, 1911698 1890,1441216,838547,87050,88908,51693,2507414 1900, 1850093, 1166582, 152999, 200507, 67021, 343720 1910,2331542,1634351,284041,430980,85969,4766883 1920, 2284103, 2018356, 469042, 732016, 116531, 5620048 1930, 1867312, 2560401, 1079129, 1265258, 158346, 6930446 1940,1889924,2698285,1297634,1394711,174441,7454995 1950, 1960101, 2738175, 1550849, 1451277, 191555, 7891957 1960, 1698281, 2627319, 1809578, 1424815, 221991, 7781984 1970, 1539233, 2602012, 1986473, 1471701, 295443, 7894862 1980, 1428285, 2230936, 1891325, 1168972, 352121, 7071639 1990,1487536,2300664,1951598,1203789,378977,7322564 2000,1537195,2465326,2229379,1332650,443728,8008278 2010, 1585873, 2504700, 2230722, 1385108, 468730, 8175133 2015,1644518,2636735,2339150,1455444,474558,8550405

nycHistPop.csv

In Lab 6



3 March 2020 22 / 39

イロト イボト イヨト イヨト 二日

Series in Pandas



• Series can store a column or row of a DataFrame.

3 March 2020 23 / 39

Series in Pandas



- Series can store a column or row of a DataFrame.
- Example: pop["Manhattan"] is the Series corresponding to the column of Manhattan data.

CSci 127 (Hunter)

Lecture 6

3 3 March 2020 23 / 39

Sac

Series in Pandas



- Series can store a column or row of a DataFrame.
- Example: pop["Manhattan"] is the Series corresponding to the column of Manhattan data.

```
● Example:

print("The largest number living in the Bronx is",

pop["Bronx"].max())

CSci 127 (Hunter) Lecture 6 3 March 2020 23/39
```



Predict what the following will do:

• print("Queens:", pop["Queens"].min())

590

< □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > <



Predict what the following will do:

```
o print("Queens:", pop["Queens"].min())
```

```
o print("S I:", pop["Staten Island"].mean())
```

Sac



Predict what the following will do:

- print("Queens:", pop["Queens"].min())
- print("S I:", pop["Staten Island"].mean())
- print("S I:", pop["Staten Island"].std())

CSci 127 (Hunter)

3 March 2020 24 / 39

3



Predict what the following will do:

- print("Queens:", pop["Queens"].min())
- print("S I:", pop["Staten Island"].mean())
- print("S I:", pop["Staten Island"].std())
- o pop.plot.bar(x="Year")

CSci 127 (Hunter)

3 March 2020 24 / 39

3



Predict what the following will do:

- o print("Queens:", pop["Queens"].min())
- print("S I:", pop["Staten Island"].mean())
- o print("S I:", pop["Staten Island"].std())
- op.plot.bar(x="Year")
- o pop.plot.scatter(x="Brooklyn", y= "Total")

CSci 127 (Hunter)

3 3 March 2020 24 / 39

Sar



Predict what the following will do:

- print("Queens:", pop["Queens"].min())
- print("S I:", pop["Staten Island"].mean())
- print("S I:", pop["Staten Island"].std())
- o pop.plot.bar(x="Year")
- pop.plot.scatter(x="Brooklyn", y= "Total")
- pop["Fraction"] = pop["Bronx"]/pop["Total"]

CSci 127 (Hunter)

3

Predict what the following will do:

• print("Queens:", pop["Queens"].min())



Э

990

Predict what the following will do:

● print("Queens:", pop["Queens"].min())

Minimum value in the column with label "Queens".



3

< ロ ト < 団 ト < 三 ト < 三 ト</p>

Predict what the following will do:

- print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean())



Э

イロト イボト イヨト イヨト

990



- print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- o print("S I:", pop["Staten Island"].mean())
 Average of values in the column "Staten Island".

Image: A math and A





Predict what the following will do:

- print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean()) Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std())

< □ > < 同 >



Predict what the following will do:

- o print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean())
 Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std()) Standard deviation of values in the column "Staten Island".

< □ > < 同 >



Predict what the following will do:

- o print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean()) Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std()) Standard deviation of values in the column "Staten Island".
- op.plot.bar(x="Year")



Predict what the following will do:

- print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean()) Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std()) Standard deviation of values in the column "Staten Island".
- pop.plot.bar(x="Year")
 Bar chart with x-axis "Year".



Predict what the following will do:

- print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean()) Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std())
 Standard deviation of values in the column "Staten
 Island".
- pop.plot.bar(x="Year") Bar chart with x-axis "Year".
- pop.plot.scatter(x="Brooklyn", y= "Total")

3 March 2020 25 / 39



Predict what the following will do:

- o print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean()) Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std()) Standard deviation of values in the column "Staten Island".
- pop.plot.bar(x="Year") Bar chart with x-axis "Year".
- pop.plot.scatter(x="Brooklyn", y= "Total")
 Scatter plot of Brooklyn versus Total values.



Predict what the following will do:

- o print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean()) Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std())
 Standard deviation of values in the column "Staten
 Island".
- pop.plot.bar(x="Year") Bar chart with x-axis "Year".
- pop.plot.scatter(x="Brooklyn", y= "Total")
 Scatter plot of Brooklyn versus Total values.
- pop["Fraction"] = pop["Bronx"]/pop["Total"]



Predict what the following will do:

- print("Queens:", pop["Queens"].min())
 Minimum value in the column with label "Queens".
- print("S I:", pop["Staten Island"].mean())
 Average of values in the column "Staten Island".
- print("S I :", pop["Staten Island"].std()) Standard deviation of values in the column "Staten Island".
- pop.plot.bar(x="Year") Bar chart with x-axis "Year".
- pop.plot.scatter(x="Brooklyn", y= "Total")
 Scatter plot of Brooklyn versus Total values.
- pop["Fraction"] = pop["Bronx"]/pop["Total"] New column with the fraction of population that lives in the Bronx.

< = > < = > < = > < = >

		Undergraduate		
College	Full-time	Part-time	Total	
Baruch	11,288	3,922	15,210	
Brooklyn	10,198	4,208	14,406	
City	10,067	3,250	13,317	
Hunter	12,223	4,500	16,723	
John Jay	9,831	2,843	12,674	
Lehman	6,600	4,720	11,320	
Medgar Evers	4,760	2,059	6,819	
NYCCT	10,912	6,370	17,282	
Queens	11,693	4,633	16,326	
Staten Island	9,584	2,948	12,532	
York	5,066	3,192	8,258	

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Sar

イロト 不得 トイラト イラト 二日

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
Yark	5,066	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

Sar

イロト 不得 トイラト イラト 二日

	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,087	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5.066	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

1 Include pandas & pyplot libraries.

Sar

イロト 不得 トイラト イラト 二日

	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,087	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

1 Include pandas & pyplot libraries.

Read in the CSV file.

	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

1 Include pandas & pyplot libraries.

- Read in the CSV file.
- 3 Set up a scatter plot.

	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,066	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

1 Include pandas & pyplot libraries.

- 2 Read in the CSV file.
- ③ Set up a scatter plot.
- ④ Display plot.

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

		Undergraduate		
College	Full-time	Part-time	Total	
Baruch	11,288	3,922	15,210	
Brooklyn	10,198	4,208	14,406	
City	10,067	3,250	13,317	
Hunter	12,223	4,500	16,723	
John Jay	9,831	2,843	12,674	
Lehman	6,600	4,720	11,320	
Medgar Evers	4,760	2,059	6,819	
NYCCT	10,912	6,370	17,282	
Queens	11,693	4,633	16,326	
Staten Island	9,584	2,948	12,532	
York	5,068	3,192	8,258	

cunyF2016.csv

Solution:

Sac

イロト イボト イヨト イヨト 二日

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

		Undergraduate		
College	Full-time	Part-time	Total	
Baruch	11,288	3,922	15,210	
Brooklyn	10,198	4,208	14,406	
City	10,067	3,250	13,317	
Hunter	12,223	4,500	16,723	
John Jay	9,831	2,843	12,674	
Lehman	6,600	4,720	11,320	
Medgar Evers	4,760	2,059	6,819	
NYCCT	10,912	6,370	17,282	
Queens	11,693	4,633	16,326	
Staten Island	9,584	2,948	12,532	
York	5,068	3,192	8,258	

cunyF2016.csv

Solution:

1 Include pandas & pyplot libraries.

Sac

イロト イボト イヨト イヨト 二日

Write a complete Python program that
reads in the file, cunyF2016.csv, and
produces a scatter plot of full-time ver-
sus part-time enrollment.

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,068	3,192	8,258

cunyF2016.csv

Solution:

 Include pandas & pyplot libraries. import matplotlib.pyplot as plt import pandas as pd

< □ > < 同 > < 臣 > < 臣 > 三 = - の < ⊙

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,405
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,068	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

- Include pandas & pyplot libraries. import matplotlib.pyplot as plt import pandas as pd
- Read in the CSV file.

Sac

イロト イボト イヨト 一日

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,405
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,068	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

- Include pandas & pyplot libraries. import matplotlib.pyplot as plt import pandas as pd
- Pread in the CSV file. pop=pd.read_csv('cunyF2016.csv', skiprows=1)
- 3 Set up a scatter plot.

Sac

イロト イボト イヨト イヨト 二日
In Pairs or Triples

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,068	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

- Include pandas & pyplot libraries. import matplotlib.pyplot as plt import pandas as pd
- Pread in the CSV file. pop=pd.read_csv('cunyF2016.csv',skiprows=1)
- 3 Set up a scatter plot. pop.plot.scatter(x="Full-time",y="Part-time")
- ④ Display plot.

200

イロト イボト イヨト イヨト 二日

In Pairs or Triples

		Undergraduate	
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten Island	9,584	2,948	12,532
York	5,068	3,192	8,258

cunyF2016.csv

Write a complete Python program that reads in the file, cunyF2016.csv, and produces a scatter plot of full-time versus part-time enrollment.

Solution:

- Include pandas & pyplot libraries. import matplotlib.pyplot as plt import pandas as pd
- Pread in the CSV file. pop=pd.read_csv('cunyF2016.csv', skiprows=1)
- 3 Set up a scatter plot. pop.plot.scatter(x="Full-time",y="Part-time")
- ④ Display plot. plt.show()

Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

Rain in	Australia			
Date	Location	MinTemp	MaxTemp	Rainfall
12/1/08	Albury	13.4	22.9	0.6
5/22/15	BadgerysCree	11	15.6	1.6
3/17/11	BadgerysCree	18.1	25.8	16.6
7/27/10	Cobar	5.3	17.2	0
9/5/10	Moree	12.1	19.8	23.4
1/23/12	CoffsHarbour	20	24.4	28
7/15/11	Moree	2.8	19	0
1/28/10	Newcastle	22.2	28	0
12/2/15	Moree	20.1	32	4.8

AustraliaRain.csv

Rain in	Australia			
Date	Location	MinTemp	MaxTemp	Rainfall
12/1/08	Albury	13.4	22.9	0.6
5/22/15	BadgerysCree	11	15.6	1.6
3/17/11	BadgerysCree	18.1	25.8	16.6
7/27/10	Cobar	5.3	17.2	0
9/5/10	Moree	12.1	19.8	23.4
1/23/12	CoffsHarbour	20	24.4	28
7/15/11	Moree	2.8	19	0
1/28/10	Newcastle	22.2	28	0
12/2/15	Moree	20.1	32	4.8

AustraliaRain.csv

Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at each location:

Rain in	Australia			
Date	Location	MinTemp	MaxTemp	Rainfall
12/1/08	Albury	13.4	22.9	0.6
5/22/15	BadgerysCree	11	15.6	1.6
3/17/11	BadgerysCree	18.1	25.8	16.6
7/27/10	Cobar	5.3	17.2	0
9/5/10	Moree	12.1	19.8	23.4
1/23/12	CoffsHarbour	20	24.4	28
7/15/11	Moree	2.8	19	0
1/28/10	Newcastle	22.2	28	0
12/2/15	Moree	20.1	32	4.8

AustraliaRain.csv

Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at each location:

 Import libraries. import pandas as pd

200

イロト イボト イヨト イヨト 二日

Rain in	Australia			
Date	Location	MinTemp	MaxTemp	Rainfall
12/1/08	Albury	13.4	22.9	0.6
5/22/15	BadgerysCree	11	15.6	1.6
3/17/11	BadgerysCree	18.1	25.8	16.6
7/27/10	Cobar	5.3	17.2	0
9/5/10	Moree	12.1	19.8	23.4
1/23/12	CoffsHarbour	20	24.4	28
7/15/11	Moree	2.8	19	0
1/28/10	Newcastle	22.2	28	0
12/2/15	Moree	20.1	32	4.8

AustraliaRain.csv

Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at each location:

- Import libraries. import pandas as pd
- 2 Read in the CSV file. rain = pd.read_csv('AustraliaRain.csv', skiprows=1)

Sac

イロト イボト イヨト イヨト 二日

Rain in .	Australia			
Date	Location	MinTemp	MaxTemp	Rainfall
12/1/08	Albury	13.4	22.9	0.6
5/22/15	BadgerysCree	11	15.6	1.6
3/17/11	BadgerysCree	18.1	25.8	16.6
7/27/10	Cobar	5.3	17.2	0
9/5/10	Moree	12.1	19.8	23.4
1/23/12	CoffsHarbour	20	24.4	28
7/15/11	Moree	2.8	19	0
1/28/10	Newcastle	22.2	28	0
12/2/15	Moree	20.1	32	4.8

AustraliaRain.csv

Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at each location:

- Import libraries. import pandas as pd
- 2 Read in the CSV file. rain = pd.read_csv('AustraliaRain.csv', skiprows=1)
- ③ Group the data by location averages.
 groupAvg =
 rain.groupby('Location').mean()

▲□▶ ▲□▶ ▲ヨ▶ ▲ヨ▶ - ヨ - ∽0.00

Rain in .	Australia			
Date	Location	MinTemp	MaxTemp	Rainfall
12/1/08	Albury	13.4	22.9	0.6
5/22/15	BadgerysCree	11	15.6	1.6
3/17/11	BadgerysCree	18.1	25.8	16.6
7/27/10	Cobar	5.3	17.2	0
9/5/10	Moree	12.1	19.8	23.4
1/23/12	CoffsHarbour	20	24.4	28
7/15/11	Moree	2.8	19	0
1/28/10	Newcastle	22.2	28	0
12/2/15	Moree	20.1	32	4.8

AustraliaRain.csv

Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at each location:

- Import libraries. import pandas as pd
- 2 Read in the CSV file. rain = pd.read_csv('AustraliaRain.csv', skiprows=1)
- ③ Group the data by location averages. groupAvg = rain.groupby('Location').mean()
- ④ Print the average rainfall at each location. print(groupAvg['Rainfall'])

Sac

イロット (日本) (日本)



Rain in	Australia			
Date	Location	MinTemp	MaxTemp	Rainfall
12/1/08	Albury	13.4	22.9	0.6
5/22/15	BadgerysCree	11	15.6	1.6
3/17/11	BadgerysCree	18.1	25.8	16.6
7/27/10	Cobar	5.3	17.2	0
9/5/10	Moree	12.1	19.8	23.4
1/23/12	CoffsHarbour	20	24.4	28
7/15/11	Moree	2.8	19	0
1/28/10	Newcastle	22.2	28	0
12/2/15	Moree	20.1	32	4.8

AustraliaRain.csv

Adelaide	1.572185
Albany	2.255073
Albury	1.925710
AliceSprings	0.869355
BadgerysCreek	2.207925
Ballarat	1.688830
Bendigo	1.621452
Brisbane	3.160536
Cairns	5.765317
Canberra	1.735038
Cobar	1.129262
CoffsHarbour	5.054592
Dartmoor	2.148554

Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at each location:

- Import libraries. import pandas as pd
- 2 Read in the CSV file. rain = pd.read_csv('AustraliaRain.csv', skiprows=1)
- ③ Group the data by location averages. groupAvg = rain.groupby('Location').mean()
- ④ Print the average rainfall at each location. print(groupAvg['Rainfall'])



Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at one location, e.g. Moree:

	MILIETO	MaxTemp	Bainfell
Abury	18.4	22.9	- 0,
BadgerysCree	11	15.6	1.1
BadgerysCree	18.1	25.8	16.1
Cobar	5.3	17.2	
Monee	12.1	19.8	23.4
CoffilHerbour	20	24.4	2
Moree	2.8	19	
Newcastle	22.2	28	
Moree	20.1	32	43
	Albury BadgerysCree BadgerysCree Cobar Moree CoffsHarbour Moree Newcaste Moree	Abury 13.4 BadgerysCree 11 BadgerysCree 18.1 Cohar 5.3 Moree 12.1 Coffairfartour 20 Moree 2.8 Moree 22.1 Moree 20.1	Albury 17.4 22.9 BadgerysCree 11 15.6 BadgerysCree 18.1 25.8 Cohar 5.3 17.2 Mone 12.1 19.8 Cohihiertour 20 24.4 None 2.8 19 Nerocafe 22.2 28

AustraliaRain.csv

< □ > < □ > < 豆 > < 豆 > < 豆 > < 豆 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □



Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at one location, e.g. Moree:

Date	Location	MinTemp	MaxTemp	Beinfell
12/1/08	Abury	18.4	22.9	0.
5/22/15	BadgerysCree	11	15.6	1.1
3/17/11	BadgerysCree	18.1	25.0	16.1
7/27/10	Cobar	5.3	17.2	
9,5(10	Monee	12.1	19.8	23.
1/23/12	CoffilHerbour	20	24.4	23
7/15/11	Moree	2.8	19	
1/28/10	Newcastle	22.2	28	
12/2/15	Moree	20.1	32	43

AustraliaRain.csv

- Import libraries. import pandas as pd
- 2 Read in the CSV file. rain = pd.read_csv('AustraliaRain.csv', skiprows=1)
- ③ Group the data by location get averages for group Moree. MoreeAvg = rain.groupby(['Location']).get_group('Moree').mean()

< ロ > < 同 > < 三 > < 三 > < 三 > < ○ < ○ </p>



Sometimes you have **recurring values in a column** and you want to examine the data for a particular value.

For example, to find the average rainfall at one location, e.g. Moree:

Date	Location	MinTemp	MaxTemp	Beinfell
12/1/08	Abury	18.4	22.9	0.
5/22/15	BadgerysCree	11	15.6	1.1
3/17/11	BadgerysCree	18.1	25.0	16.1
7/27/10	Cobar	5.3	17.2	
9,5(10	Monee	12.1	19.8	23.
1/23/12	CoffilHerbour	20	24.4	23
7/15/11	Moree	2.8	19	
1/28/10	Newcastle	22.2	28	
12/2/15	Moree	20.1	32	43

AustraliaRain.csv

- Import libraries. import pandas as pd
- 2 Read in the CSV file. rain =

pd.read_csv('AustraliaRain.csv',skiprows=1)

- ③ Group the data by location get averages for group Moree. MoreeAvg = rain.groupby(['Location']).get_group('Moree').mean()
- ④ Print the average rainfall. print(MoreeAvg['Rainfall'])

Today's Topics



- Recap: Logical Expressions & Circuits
- Design: Cropping Images
- Accessing Formatted Data
- CS Survey: Astrophysics and astropy

3

Sar

CS Survey: Prof. Cruz, Astrophysics



Hunter College - Department of Physics and Astronomy American History Museum - Department of Astrophysics http://www.hunter.cuny.edu/physics/faculty/cruz/kelle-cruz



CSci 127 (Hunter)

イロト イポト イヨト イヨ

Design Challenge

Stars						
Temperature (K)	Luminosity(L/Lo)	Radius(R/Ro)	Absolute magnitude(Mv)	Star type	Star color	Spectral Class
3068	0.0024	0.17	16.12	Brown Dwarf	Red	M
25000	0.056	0.0084	10.58	White Dwarf	Blue White	В
2650	0.00069	0.11	17.45	Brown Dwarf	Red	M
11790	0.00015	0.011	12.59	White Dwarf	Yellowish White	F
15276	1136	7.2	-1.97	Main Sequence	Blue-white	В
5800	0.81	0.9	5.05	Main Sequence	yellow-white	F
16500	0.013	0.014	11.89	White Dwarf	Blue White	в
3192	0.00362	0.1967	13.53	Red Dwarf	Red	м
6380	1.35	0.98	2.93	Main Sequence	yellow-white	F
3834	272000	1183	-9.2	Hypergiant	Red	M
3628	0.0055	0.393	10.48	Red Dwarf	Red	м
3749	550000	1648	-8.05	Hypergiant	Orange	м

On your Lecture Slip, design an algorithm that:

- Prints the luminosity of the brightest star.
- Prints the temperature in Kelvin (K) of the coldest star.
- Prints the temperature in Fahrenheit of the coldest star. New: astropy.units will seamlessly convert!!!
- Prints the average radius of a Hypergiant

CSci 127 (Hunter)

Luminosity(L/Lo)	Radius(R/Ro)	Absolute magnitude(Mv)	Star type	Star color	Spectral Class
0.0024	0.17	16.12	Brown Dwarf	Red	м
0.056	0.0084	10.58	White Dwarf	Blue White	В
0.00069	0.11	17.45	Brown Dwarf	Red	M
0.00015	0.011	12.59	White Dwarf	Yellowish White	F
1136	7.2	-1.97	Main Sequence	Blue-white	В
0.81	0.9	5.05	Main Sequence	yellow-white	F
0.013	0.014	11.89	White Dwarf	Blue White	В
0.00362	0.1967	13.53	Red Dwarf	Red	м
1.35	0.98	2.93	Main Sequence	yellow-white	F
272000	1183	-9.2	Hypergiant	Red	M
0.0055	0.393	10.48	Red Dwarf	Red	M
550000	1648	-8.05	Hypergiant	Orange	M
	Luminosity(L/Lo) 0.0024 0.056 0.00059 0.00015 1136 0.81 0.0132 1.33 272000 0.0055 550000	Luminosity(L/Lo) Radius(R/Re) 0.0024 0.17 0.056 0.0084 0.0005 0.11 1.06 7.9 0.013 0.014 0.0032 0.1967 1.35 0.98 277000 1188 0.0055 0.389 550000 1648	Luminosity(L/Lo) Radius(P/Ro) Absolute magnitude(Mv) 0.0024 0.17 16.12 0.056 0.0084 10.58 0.00045 0.011 17.45 0.00059 0.111 17.45 0.00015 0.011 12.59 1115 2 1.50 0.013 0.14 1.88 0.0052 0.167 1353 1.35 0.98 2.93 27000 118 9.42 500000 1648 -8.65	Luminosity(L/Lo) Radius(#/Ro) Absolute magnitude(Mv) Star type 0.0024 0.17 16.12 Brown Dwarf 0.0056 0.0084 10.58 Withe Dwarf 0.00056 0.011 17.45 Brown Dwarf 0.0005 0.011 12.59 Withe Dwarf 119 7.1 5 Drown Dwarf 0.0015 0.012 5.9 Mithe Dwarf 0.013 0.014 1.18 Withe Dwarf 0.013 0.014 1.18 Withe Dwarf 1.35 0.98 2.93 Main Sequence 272000 138 -9.2 Hypergint 0.0055 0.393 10.48 Red Dwarf 550000 1649 -8.05 Hypergint	Luminosity(L/Lo) Radius(R/Ro) Absolute magnitude(Mr) Star color 0.0024 0.17 16.12 Brown Dwarf Red 0.0056 0.0084 10.58 White Dwarf Red 0.00069 0.11 17.45 Brown Dwarf Red 0.00015 0.011 12.59 White Dwarf Red 1.05 0.015 0.011 12.59 White Dwarf Weilwich White 0.013 0.014 1.89 White Dwarf Blue White Value White 0.013 0.014 1.89 White Dwarf Blue White Second 0.013 0.014 1.89 White Dwarf Blue White Second Second 1.35 0.98 2.35 Main Sequence yellow-white Second Second

• Libraries: pandas and astropy

999

Stars						
Temperature (K)	Luminosity(L/Lo)	Radius(R/Ro)	Absolute magnitude(Mv)	Star type	Star color	Spectral Class
3068	0.0024	0.17	16.12	Brown Dwarf	Red	м
25000	0.056	0.0084	10.58	White Dwarf	Blue White	В
2650	0.00069	0.11	17.45	Brown Dwarf	Red	M
11790	0.00015	0.011	12.59	White Dwarf	Yellowish White	F
15276	1136	7.2	-1.97	Main Sequence	Blue-white	В
5800	0.81	0.9	5.05	Main Sequence	yellow-white	F
16500	0.013	0.014	11.89	White Dwarf	Blue White	в
3192	0.00362	0.1967	13.53	Red Dwarf	Red	м
6380	1.35	0.98	2.93	Main Sequence	yellow-white	F
3834	272000	1183	-9.2	Hypergiant	Red	м
3628	0.0055	0.393	10.48	Red Dwarf	Red	M
3749	550000	1648	-8.05	Hypergiant	Orange	м

- Libraries: pandas and astropy
- Process:
 - Print max of 'Luminosity' column

Luminosity(L/Lo)	Radius(R/Ro)	Absolute magnitude(Mv)	Star type	Star color	Spectral Class
0.0024	0.17	16.12	Brown Dwarf	Red	м
0.056	0.0084	10.58	White Dwarf	Blue White	В
0.00069	0.11	17.45	Brown Dwarf	Red	M
0.00015	0.011	12.59	White Dwarf	Yellowish White	F
1136	7.2	-1.97	Main Sequence	Blue-white	В
0.81	0.9	5.05	Main Sequence	yellow-white	F
0.013	0.014	11.89	White Dwarf	Blue White	В
0.00362	0.1967	13.53	Red Dwarf	Red	м
1.35	0.98	2.93	Main Sequence	yellow-white	F
272000	1183	-9.2	Hypergiant	Red	M
0.0055	0.393	10.48	Red Dwarf	Red	м
550000	1648	-8.05	Hypergiant	Orange	M
	Luminosity(L/Lo) 0.0024 0.056 0.00059 0.00015 1136 0.013 0.00362 1.33 272000 0.0055 550000	Luminosity(L/Lo) Radius(R/Ro) 0.0024 0.77 0.0056 0.0084 0.00009 0.11 1.136 7.2 0.81 0.9 0.013 0.014 0.0032 0.1967 1.35 0.98 272000 1188 0.0055 0.393 550000 16484	Luminosity(L/Lo) Radius(R/Ko) Absolute magnitude(Mv) 0.024 0.17 16.12 0.056 0.0084 10.58 0.00069 0.11 17.45 0.00015 0.011 17.45 0.0015 0.011 17.45 0.013 0.014 1.18 0.013 0.014 1.18 0.013 0.98 2.93 1.35 0.98 2.93 272000 1183 -9.2 0.0055 0.393 10.48 550000 1648 -8.40	Luminosity(L/Lo) Radius(#/Ro) Absolute magnitude(Mv) Star type 0.056 0.0084 10.58 White Dwarf 0.0009 0.11 17.45 Brown Dwarf 0.00015 0.011 17.45 Brown Dwarf 0.00015 0.011 17.45 Brown Dwarf 1.136 7.2 -1.97 Main Sequence 0.013 0.014 1.189 White Dwarf 0.0032 0.1967 13.53 Red Dwarf 1.35 0.98 2.93 Main Sequence 0.0032 0.1967 13.53 Red Dwarf 1.35 0.98 2.93 Main Sequence 272000 1183 -9.2 Hypergiant 0.0055 0.333 10.48 Red Dwarf 0.5000 1648 -8.05 Hypergiant	Luminosity(I/Lo) Radius(R/Ro) Absolute magnitude(Mv) Star color 0.0024 0.17 15.12 frown Dwarf Red 0.056 0.0084 10.58 White Dwarf Blue White 0.0009 0.11 17.45 frown Dwarf Red 0.00015 0.011 12.59 White Dwarf Velowish White 1136 7.2 -1.97 Main Sequence Velow-white 0.013 0.014 1.89 White Dwarf Blue White 0.133 0.014 1.89 White Dwarf Blue White 1.35 0.98 2.93 Main Sequence Velow-white 1.35 0.98 2.93 Main Sequence Velow-white 272000 1183 -9.2 Hypergiant Red 0.0055 0.393 10.48 Red Dwarf Red 550000 1648 -8.08 Hypergiant Orage

• Libraries: pandas and astropy

Process:

- Print max of 'Luminosity' column
- > Prints min of 'Temperature' column and store it in temp variable

200

イロト イボト イヨト イヨト 二日

Stars						
Temperature (K)	Luminosity(L/Lo)	Radius(R/Ro)	Absolute magnitude(Mv)	Star type	Star color	Spectral Class
3068	0.0024	0.17	16.12	Brown Dwarf	Red	м
25000	0.056	0.0084	10.58	White Dwarf	Blue White	В
2650	0.00069	0.11	17.45	Brown Dwarf	Red	M
11790	0.00015	0.011	12.59	White Dwarf	Yellowish White	F
15276	1136	7.2	-1.97	Main Sequence	Blue-white	В
5800	0.81	0.9	5.05	Main Sequence	yellow-white	F
16500	0.013	0.014	11.89	White Dwarf	Blue White	в
3192	0.00362	0.1967	13.53	Red Dwarf	Red	м
6380	1.35	0.98	2.93	Main Sequence	yellow-white	F
3834	272000	1183	-9.2	Hypergiant	Red	м
3628	0.0055	0.393	10.48	Red Dwarf	Red	M
3749	550000	1648	-8.05	Hypergiant	Orange	м

• Libraries: pandas and astropy

• Process:

- Print max of 'Luminosity' column
- Prints min of 'Temperature' column and store it in temp variable
- Use **astropy to convert** temp variable to Fahrenheit and print

イロト イポト イヨト イヨト 二日

Stars						
Temperature (K)	Luminosity(L/Lo)	Radius(R/Ro)	Absolute magnitude(Mv)	Star type	Star color	Spectral Class
3068	0.0024	0.17	16.12	Brown Dwarf	Red	м
25000	0.056	0.0084	10.58	White Dwarf	Blue White	В
2650	0.00069	0.11	17.45	Brown Dwarf	Red	M
11790	0.00015	0.011	12.59	White Dwarf	Yellowish White	F
15276	1136	7.2	-1.97	Main Sequence	Blue-white	В
5800	0.81	0.9	5.05	Main Sequence	yellow-white	F
16500	0.013	0.014	11.89	White Dwarf	Blue White	в
3192	0.00362	0.1967	13.53	Red Dwarf	Red	м
6380	1.35	0.98	2.93	Main Sequence	yellow-white	F
3834	272000	1183	-9.2	Hypergiant	Red	м
3628	0.0055	0.393	10.48	Red Dwarf	Red	M
3749	550000	1648	-8.05	Hypergiant	Orange	м

- Libraries: pandas and astropy
- Process:
 - Print max of 'Luminosity' column
 - Prints min of 'Temperature' column and store it in temp variable
 - Use astropy to convert temp variable to Fahrenheit and print
 - groupby 'Star Type' and take averages, then print max of 'Radius' column

CSci 127 (Hunter)

▲□▶ ▲□▶ ▲ヨ▶ ▲ヨ▶ - ヨ - ∽0.00

Stars						
Temperature (K)	Luminosity(L/Lo)	Radius(R/Ro)	Absolute magnitude(Mv)	Star type	Star color	Spectral Class
3068	0.0024	0.17	16.12	Brown Dwarf	Red	м
25000	0.056	0.0084	10.58	White Dwarf	Blue White	В
2650	0.00069	0.11	17.45	Brown Dwarf	Red	M
11790	0.00015	0.011	12.59	White Dwarf	Yellowish White	F
15276	1136	7.2	-1.97	Main Sequence	Blue-white	В
5800	0.81	0.9	5.05	Main Sequence	yellow-white	F
16500	0.013	0.014	11.89	White Dwarf	Blue White	в
3192	0.00362	0.1967	13.53	Red Dwarf	Red	м
6380	1.35	0.98	2.93	Main Sequence	yellow-white	F
3834	272000	1183	-9.2	Hypergiant	Red	м
3628	0.0055	0.393	10.48	Red Dwarf	Red	M
3749	550000	1648	-8.05	Hypergiant	Orange	м

- Libraries: pandas and astropy
- Process:
 - Print max of 'Luminosity' column
 - > Prints min of 'Temperature' column and store it in temp variable
 - Use astropy to convert temp variable to Fahrenheit and print
 - groupby 'Star Type' and take averages, then print max of 'Radius' column
 - OR groupby 'Star Type' and get group 'Hypergiant' to print average 'Radius'

CSci 127 (Hunter)

< □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > <

Design Challenge - Code

 Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')

< □ > < 同 > < 臣 > < 臣 > 三 = - の < ⊙

Design Challenge - Code

- Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')
- Process:
 - Print max of 'Luminosity' column print(stars['Luminosity(L/Lo)'].max())

Design Challenge - Code

- Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')
- Process:
 - Print max of 'Luminosity' column print(stars['Luminosity(L/Lo)'].max())
 - Prints min of 'Temperature' column and store it in temp variable minTempK = stars['Temperature (K)'].min() print(minTempK)

Design Challenge - Code

- Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')
- Process:
 - Print max of 'Luminosity' column print(stars['Luminosity(L/Lo)'].max())
 - Prints min of 'Temperature' column and store it in temp variable minTempK = stars['Temperature (K)'].min() print(minTempK)
 - b Use astropy to convert temp variable to Fahrenheit and print
 KUnit = minTempK * u.K
 print(KUnit.to(u.imperial.deg_F, equivalencies = \
 u.temperature()))

Design Challenge - Code

- Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')
- Process:
 - Print max of 'Luminosity' column print(stars['Luminosity(L/Lo)'].max())
 - Prints min of 'Temperature' column and store it in temp variable minTempK = stars['Temperature (K)'].min() print(minTempK)
 - > Use astropy to convert temp variable to Fahrenheit and print
 KUnit = minTempK * u.K
 print(KUnit.to(u.imperial.deg_F, equivalencies = \
 u.temperature()))
 - groupby 'Star Type' and take averages, then print max of 'Radius' column

Design Challenge - Code

- Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')
- Process:
 - Print max of 'Luminosity' column print(stars['Luminosity(L/Lo)'].max())
 - Prints min of 'Temperature' column and store it in temp variable minTempK = stars['Temperature (K)'].min() print(minTempK)
 - > Use astropy to convert temp variable to Fahrenheit and print KUnit = minTempK * u.K print(KUnit.to(u.imperial.deg_F, equivalencies = \ u.temperature()))
 - groupby 'Star Type' and take averages, then print max of 'Radius' column

```
print(stars.groupby(['Star type'])\
.mean()['Radius(R/Ro)'].max())
```

```
CSci 127 (Hunter)
```

Design Challenge - Code

- Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')
- Process:
 - Print max of 'Luminosity' column print(stars['Luminosity(L/Lo)'].max())
 - Prints min of 'Temperature' column and store it in temp variable minTempK = stars['Temperature (K)'].min() print(minTempK)
 - Use astropy to convert temp variable to Fahrenheit and print
 KUnit = minTempK * u.K
 print(KUnit.to(u.imperial.deg_F, equivalencies = \
 u.temperature()))

Design Challenge - Code

- Libraries: pandas and astropy import pandas as pd import astropy.units as u stars = pd.read_csv('Stars.csv')
- Process:
 - Print max of 'Luminosity' column print(stars['Luminosity(L/Lo)'].max())
 - Prints min of 'Temperature' column and store it in temp variable minTempK = stars['Temperature (K)'].min() print(minTempK)
 - Use astropy to convert temp variable to Fahrenheit and print
 KUnit = minTempK * u.K
 print(KUnit.to(u.imperial.deg_F, equivalencies = \
 u.temperature()))
 - OR groupby 'Star Type' and get group 'Hypergiant' to print average 'Radius'

.get_group('Hypergiant').mean()['Radius(R/Ro)'])

CSci 127 (Hunter)



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



990

- On lecture slip, write down a topic you wish we had spent more time (and why).
- Recap: Logical Expressions & Circuits



Sac

イロト イボト イヨト イヨト

- On lecture slip, write down a topic you wish we had spent more time (and why).
- Recap: Logical Expressions & Circuits
- Accessing Formatted Data:
 - Pandas library has elegant solutions for accessing & analyzing structured data.



3

Sac

イロト イヨト イヨト

- On lecture slip, write down a topic you wish we had spent more time (and why).
- Recap: Logical Expressions & Circuits
- Accessing Formatted Data:
 - Pandas library has elegant solutions for accessing & analyzing structured data.
 - Can manipulate individual columns or rows ('Series').



3

イロト イヨト イヨト

- On lecture slip, write down a topic you wish we had spent more time (and why).
- Recap: Logical Expressions & Circuits
- Accessing Formatted Data:
 - Pandas library has elegant solutions for accessing & analyzing structured data.
 - Can manipulate individual columns or rows ('Series').
 - Has useful functions for the entire sheet ('DataFrame') such as plotting.



イロト イロト イヨト イヨト

we had spent more time (and why).

On lecture slip, write down a topic you wish

- Recap: Logical Expressions & Circuits
- Accessing Formatted Data:
 - Pandas library has elegant solutions for accessing & analyzing structured data.
 - Can manipulate individual columns or rows ('Series').
 - Has useful functions for the entire sheet ('DataFrame') such as plotting.
- Pass your lecture slips to the aisles for the UTAs to collect.



イロト イロト イヨト イヨト

Practice Quiz & Final Questions





• 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:

CSci 127 (Hunter)

Lecture 6

3 March 2020 38 / 39





- 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;

CSci 127 (Hunter)

3 March 2020 38 / 39





- 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

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.

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).

CSci 127 (Hunter)

< ロト < 同ト < ヨト < 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: Unix commands!

CSci 127 (Hunter)

Writing Boards



• Return writing boards as you leave...

CSci 127 (Hunter)

Lecture 6

3 March 2020 39 / 39

Э

990

< ロ ト < 団 ト < 三 ト < 三 ト</p>