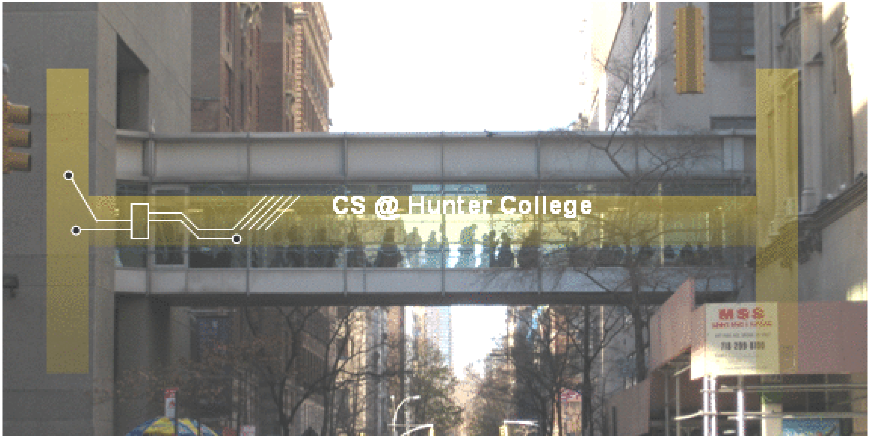
CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Announcements

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

  Today: Prof. Raffi Khatchadourian (software engineering)

Frequently Asked Questions

From lecture slips & recitation sections.

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?
  The registrar scheduled multiple classes for the same time slot. We are working with the dean's office to get this resolved.

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?
  The registrar scheduled multiple classes for the same time slot.
  We are working with the dean's office to get this resolved.
- I'm worried about the final since I've never taken a programming exam before. Help!

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?
  The registrar scheduled multiple classes for the same time slot. We are working with the dean's office to get this resolved.
- I'm worried about the final since I've never taken a programming exam before. Help!
  We understand. Starting today, we're going to spend the last 5 minutes of lecture on mock final exam questions.

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?
  The registrar scheduled multiple classes for the same time slot.
  We are working with the dean's office to get this resolved.
- I'm worried about the final since I've never taken a programming exam before. Help!
  We understand. Starting today, we're going to spend the last 5 minutes of lecture on mock final exam questions.
- I still don't get indices and the brackets. Could you spend more time on that?

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?
  The registrar scheduled multiple classes for the same time slot. We are working with the dean's office to get this resolved.
- I'm worried about the final since I've never taken a programming exam before. Help!
  We understand. Starting today, we're going to spend the last 5 minutes of lecture on mock final exam questions.
- 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).

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?
  The registrar scheduled multiple classes for the same time slot.
  We are working with the dean's office to get this resolved.
- I'm worried about the final since I've never taken a programming exam before. Help!
  We understand. Starting today, we're going to spend the last 5 minutes of lecture on mock final exam questions.
- 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).
- Could you spend more time on circuits/logical expressions/truth tables/decisions?

Frequently Asked Questions

From lecture slips & recitation sections.

- I have two finals scheduled at the same time. What do I do?
  The registrar scheduled multiple classes for the same time slot. We are working with the dean's office to get this resolved.
- I'm worried about the final since I've never taken a programming exam before. Help!
  We understand. Starting today, we're going to spend the last 5 minutes of lecture on mock final exam questions.
- 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).
- Could you spend more time on circuits/logical expressions/truth tables/decisions?
  We will do a bit today, but much more in the following weeks.

Today's Topics

- Recap: Logical Expressions & Circuits
- Accessing Formatted Data
- Preview: Functions
- Final Exam Overview

Recap: Logical Operators

and

in1		in2	*returns:*
`False`	`and`	`False`	`False`
`False`	`and`	`True`	`False`
`True`	`and`	`False`	`False`
`True`	`and`	`True`	`True`

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

Recap: Logical Operators

and

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

or

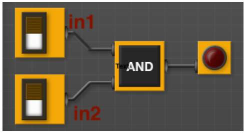
in1		in2	*returns:*
False	or	False	False
False	or	True	True
True	or	False	True
True	or	True	True

not

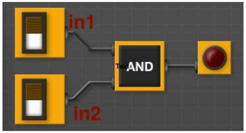
	in1	*returns:*
not	False	True
not	True	False

Logical Operators & Circuits

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



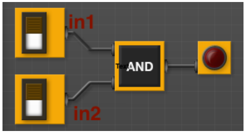
Logical Operators & Circuits



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

  Example: `in1 and in2`

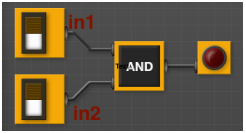
Logical Operators & Circuits



- 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.

Logical Operators & Circuits



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

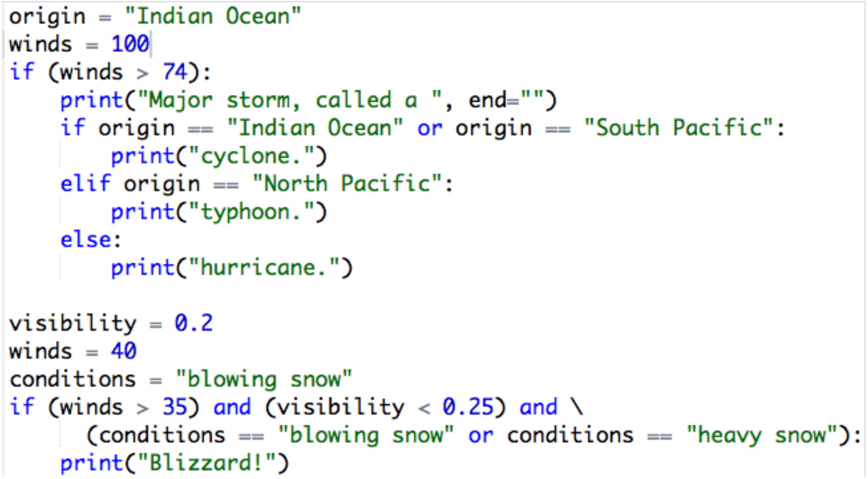
  Example: `in1 and in2`

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

  Example: see image.

Examples:

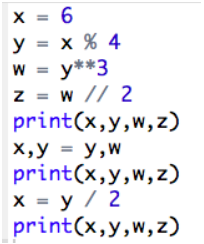
Examples from last lecture:



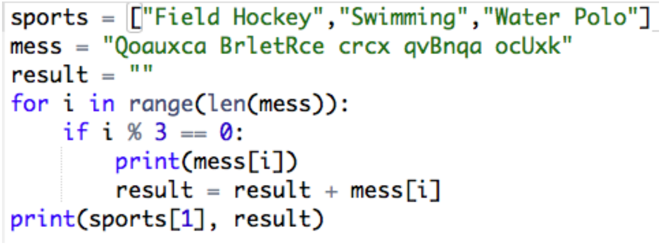
```
origin = "Indian Ocean"
winds = 100
if (winds > 74):
    print("Major storm, called a ", end="")
    if origin == "Indian Ocean" or origin == "South Pacific":
        print("cyclone.")
    elif origin == "North Pacific":
        print("typhoon.")
    else:
        print("hurricane.")

visibility = 0.2
winds = 40
conditions = "blowing snow"
if (winds > 35) and (visibility < 0.25) and \
      (conditions == "blowing snow" or conditions == "heavy snow"):
    print("Blizzard!")
```

In Pairs or Triples:

Predict what the code will do:



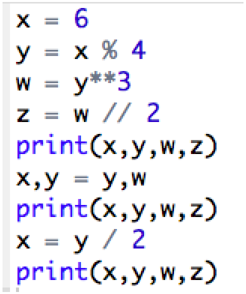
```python
x = 6
y = x % 4
w = y**3
z = w // 2
print(x,y,w,z)
x,y = y,w
print(x,y,w,z)
x = y / 2
print(x,y,w,z)
```



```python
sports = ["Field Hockey","Swimming","Water Polo"]
mess = "Qoauxca BrletRce crcx qvBnqa ocUxk"
result = ""
for i in range(len(mess)):
    if i % 3 == 0:
        print(mess[i])
        result = result + mess[i]
print(sports[1], result)
```

- And, design a program that asks the user for an image and then displays the upper left quarter of the image.
  (First, design the pseudocode. If time, expand to a Python program.)

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`)

Design Question

And, 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.)

Design Question

And, 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.

Design Question

And, 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.

Design Question

And, 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.

Design Question

And, 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:

Design Question

And, 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. Ask user for an image name.

Design Question

And, 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. Ask user for an image name.
  2. Read in image.

Design Question

And, 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. Ask user for an image name.
    2. Read in image.
    3. Figure out size of image.

Design Question

And, 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. Ask user for an image name.
    2. Read in image.
    3. Figure out size of image.
    4. Make a new image that’s half the height and half the width.

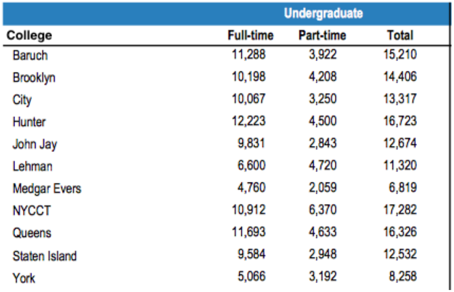
Design Question

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

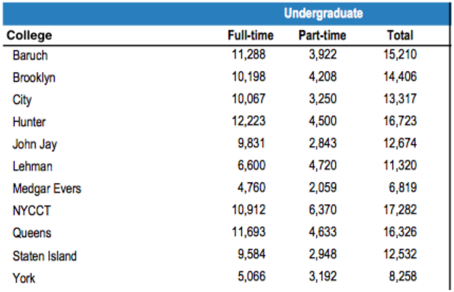
Structured Data



	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.

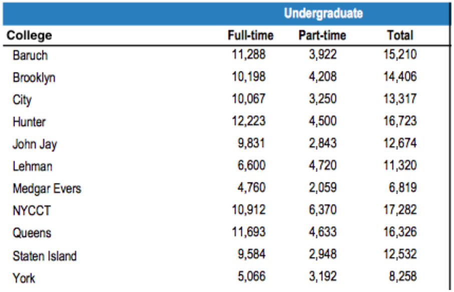
Structured Data



	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".

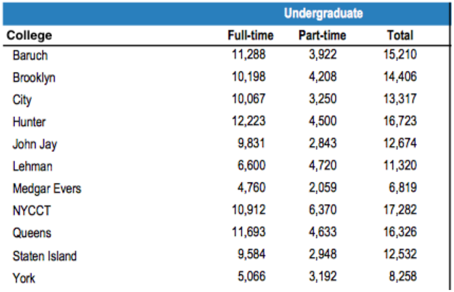
Structured Data



	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.

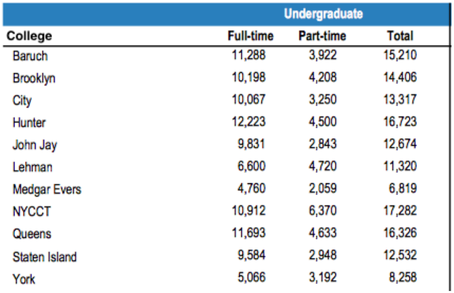
Structured Data



	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.

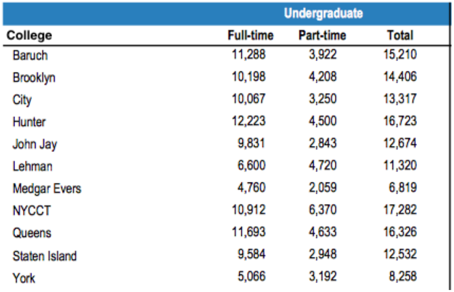
Structured Data



	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.

Structured Data



	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**).

Structured Data

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

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

Structured Data

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

Structured Data

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

- 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.

Structured Data

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

- 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.

Structured Data

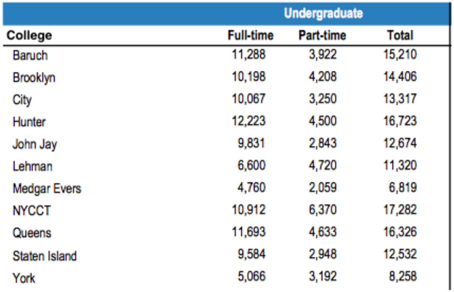
pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

- 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
```

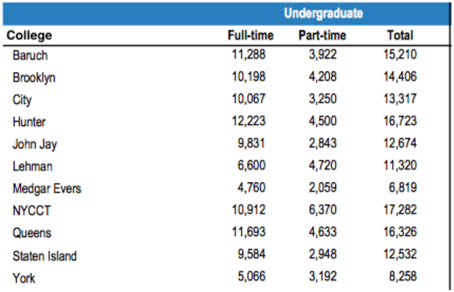
CSV Files



	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.

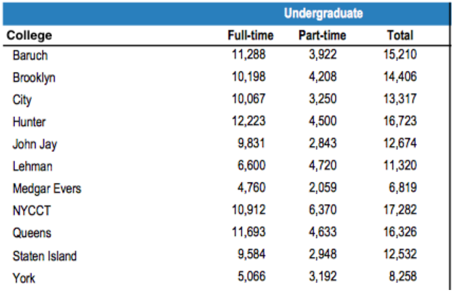
CSV Files



	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.

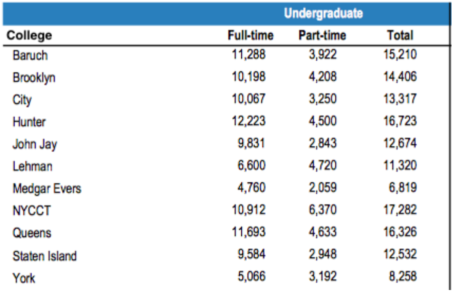
CSV Files



	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.

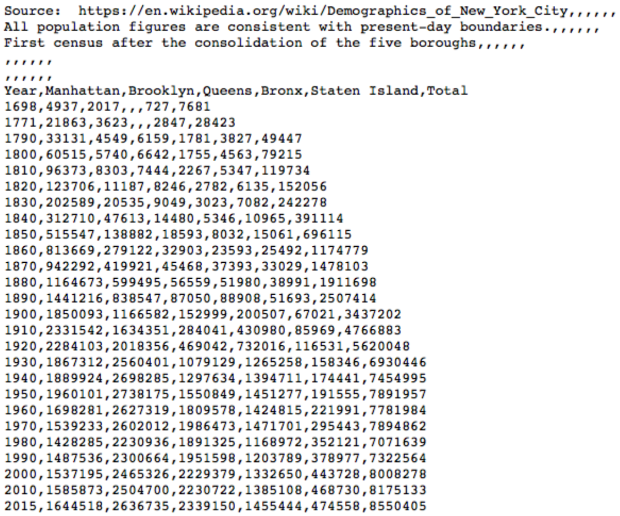
CSV Files



	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.

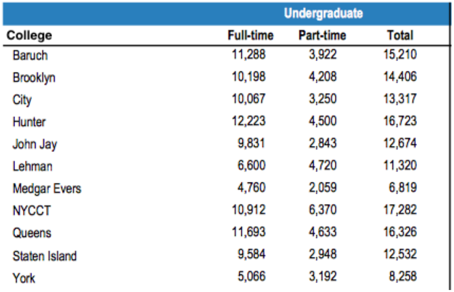
CSV Files



```
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,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`

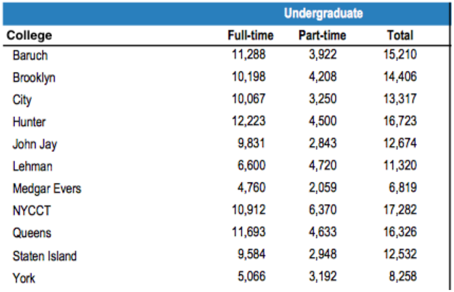
Reading in CSV Files



	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")`

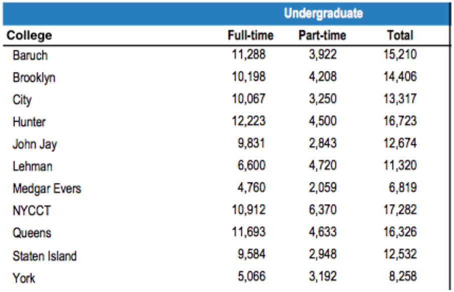
Reading in CSV Files



	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.

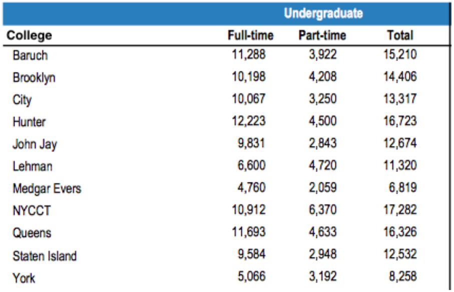
Reading in CSV Files



	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`.

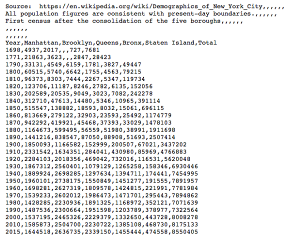
Reading in CSV Files



	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.

Example: Reading in CSV Files



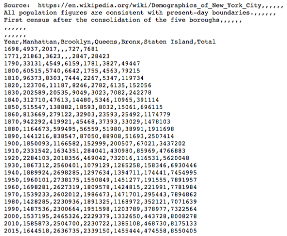
```
Source:  https://en.wikipedia.org/wiki/Demographics_of_New_York_City,,,,,,
All population figures are consistent with present-day boundaries.,,,,,,
First census after the consolidation of the five boroughs,,,,,,
,,,,,,
,,,,,,
Year,Manhattan,Brooklyn,Queens,Bronx,Staten Island,Total
1698,4937,2017,,,727,7681
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

Example: Reading in CSV Files

```
import matplotlib.pyplot as plt
import pandas as pd
```



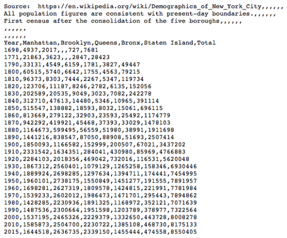
```
Source:  https://en.wikipedia.org/wiki/Demographics_of_New_York_City,,,,,,
All population figures are consistent with present-day boundaries.,,,,,,
First census after the consolidation of the five boroughs,,,,,,
,,,,,,
,,,,,,
Year,Manhattan,Brooklyn,Queens,Bronx,Staten Island,Total
1698,4937,2017,,,727,7681
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

Example: Reading in CSV Files

```
import matplotlib.pyplot as plt
import pandas as pd

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



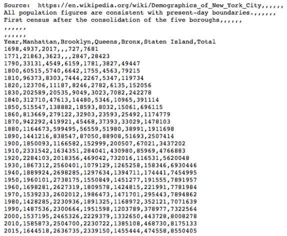
```
Source:  https://en.wikipedia.org/wiki/Demographics_of_New_York_City,,,,,,
All population figures are consistent with present-day boundaries.,,,,,,
First census after the consolidation of the five boroughs,,,,,,
,,,,,,
,,,,,,
Year,Manhattan,Brooklyn,Queens,Bronx,Staten Island,Total
1698,4937,2017,,,727,7681
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

Example: Reading in CSV Files

```
import matplotlib.pyplot as plt
import pandas as pd

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

pop.plot(x="Year")
plt.show()
```



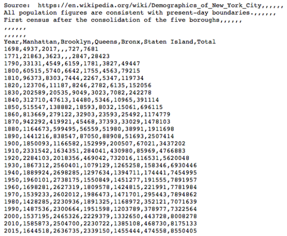
```
Source:  https://en.wikipedia.org/wiki/Demographics_of_New_York_City,,,,,,
All population figures are consistent with present-day boundaries.,,,,,,
First census after the consolidation of the five boroughs,,,,,,
,,,,,,
,,,,,,
Year,Manhattan,Brooklyn,Queens,Bronx,Staten Island,Total
1698,4937,2017,,,727,7681
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

Example: Reading in CSV Files

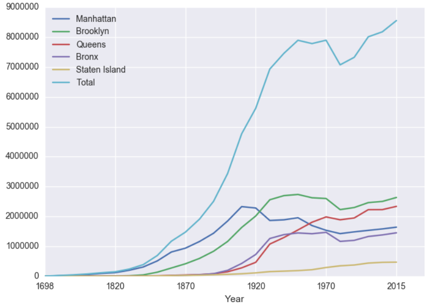
```
import matplotlib.pyplot as plt
import pandas as pd

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

pop.plot(x="Year")
plt.show()
```



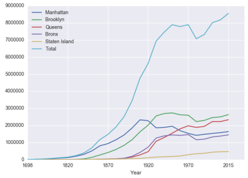
```
Source:  https://en.wikipedia.org/wiki/Demographics_of_New_York_City,,,,,,
All population figures are consistent with present-day boundaries.,,,,,,
First census after the consolidation of the five boroughs,,,,,,
,,,,,,
,,,,,,
Year,Manhattan,Brooklyn,Queens,Bronx,Staten Island,Total
1698,4937,2017,,,727,7681
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

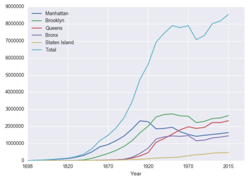


Series in Pandas



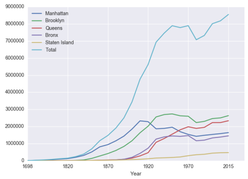
- Series can store a column or row of a DataFrame.

Series in Pandas



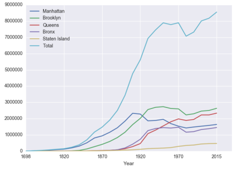
- Series can store a column or row of a DataFrame.
- Example: `pop["Manhattan"]` is the Series corresponding to the column of Manhattan data.

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())`

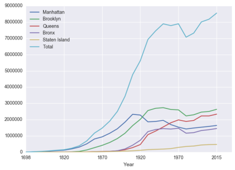
In Pairs or Triples



Predict what the following will do:

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

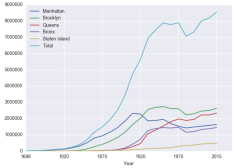
In Pairs or Triples



Predict what the following will do:

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

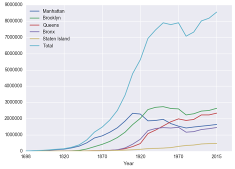
In Pairs or Triples



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())`

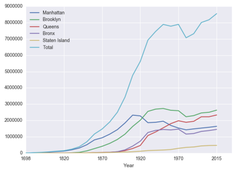
In Pairs or Triples



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())`
- `pop.plot.bar(x="Year")`

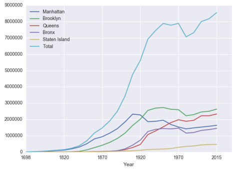
In Pairs or Triples



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())`
- `pop.plot.bar(x="Year")`
- `pop.plot.scatter(x="Brooklyn", y="Total")`

In Pairs or Triples



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())`
- `pop.plot.bar(x="Year")`
- `pop.plot.scatter(x="Brooklyn", y="Total")`
- `pop["Fraction"] = pop["Bronx"]/pop["Total"]`

CS Survey Talk

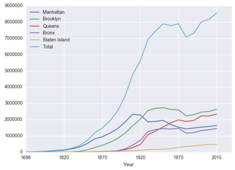
Prof. Raffi Khatchadourian

Department of Computer Science
Hunter College & the Graduate Center
Software Engineering

Solutions

Predict what the following will do:

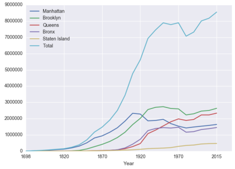
- `print("Queens:", pop["Queens"].min())`



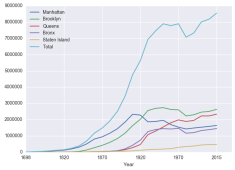
Solutions

Predict what the following will do:

- `print("Queens:", pop["Queens"].min())`
  Minimum value in the column with label "Queens".



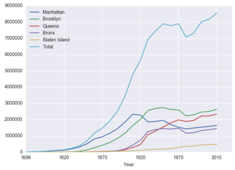
Solutions



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())`

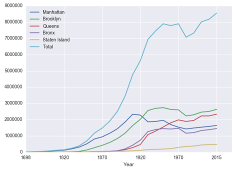
Solutions



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())`
  Minimum value in the column "Staten Island".

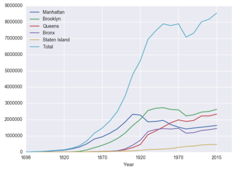
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`

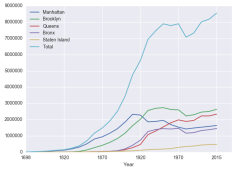
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`
  Minimum value in the column "Staten Island".

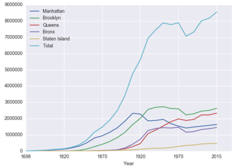
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`
  Minimum value in the column "Staten Island".
- `pop.plot.bar(x="Year")`

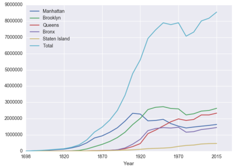
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`
  Minimum value in the column "Staten Island".
- `pop.plot.bar(x="Year")`
  Bar chart with x-axis "Year".

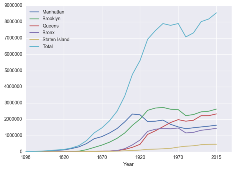
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`
  Minimum value in the column "Staten Island".
- `pop.plot.bar(x="Year")`
  Bar chart with x-axis "Year".
- `pop.plot.scatter(x="Brooklyn", y= "Total")`

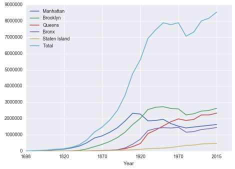
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`
  Minimum value 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.

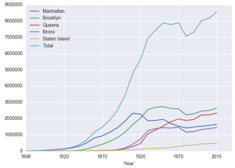
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`
  Minimum value 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"]`

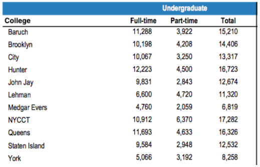
Solutions



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())`
  Minimum value in the column "Staten Island".
- `print("S I :", pop["Staten Island"].std())`
  Minimum value 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.

In Pairs or Triples

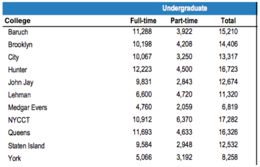


College	Undergraduate		
	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten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

In Pairs or Triples



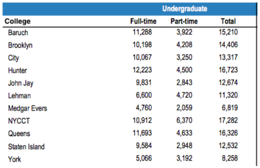
College	Undergraduate Full-time	Undergraduate Part-time	Undergraduate Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	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:

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	5,066	3,192	8,258

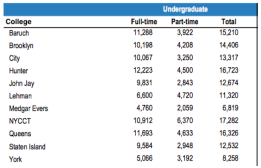
`cunyF2016.csv`

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.*

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	5,066	3,192	8,258

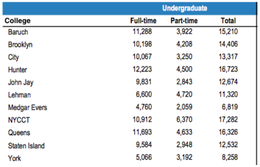
`cunyF2016.csv`

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.*

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	5,066	3,192	8,258

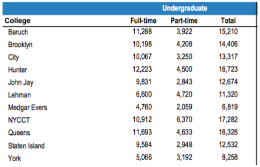
`cunyF2016.csv`

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.*

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	5,066	3,192	8,258

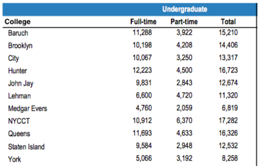
`cunyF2016.csv`

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.*

In Pairs or Triples



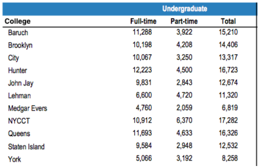
	Undergraduate		
College	Full-time	Part-time	Total
Baruch	11,288	3,922	15,210
Brooklyn	10,198	4,208	14,406
City	10,067	3,250	13,317
Hunter	12,223	4,500	16,723
John Jay	9,831	2,843	12,674
Lehman	6,600	4,720	11,320
Medgar Evers	4,760	2,059	6,819
NYCCT	10,912	6,370	17,282
Queens	11,693	4,633	16,326
Staten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

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	5,066	3,192	8,258

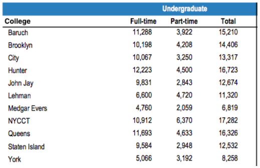
`cunyF2016.csv`

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.*

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	5,066	3,192	8,258

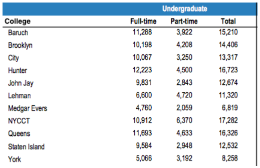
`cunyF2016.csv`

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.*
```
import matplotlib.pyplot as plt
import pandas as pd
```

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	5,066	3,192	8,258

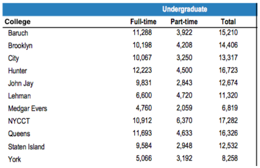
`cunyF2016.csv`

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.*
   ```
   import matplotlib.pyplot as plt
   import pandas as pd
   ```
2. *Read in the CSV file.*

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	5,066	3,192	8,258

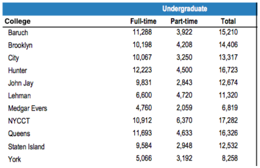
`cunyF2016.csv`

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.*
   ```
   import matplotlib.pyplot as plt
   import pandas as pd
   ```
2. *Read in the CSV file.*
   ```
   pop=pd.read_csv('cunyF2016.csv',skiprows=1)
   ```
3. *Set up a scatter plot.*

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	5,066	3,192	8,258

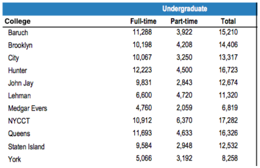
`cunyF2016.csv`

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

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	5,066	3,192	8,258

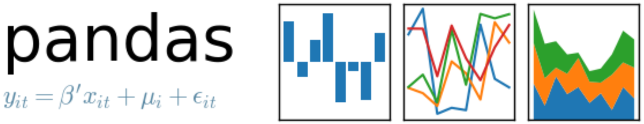
`cunyF2016.csv`

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.*
   ```
   import matplotlib.pyplot as plt
   import pandas as pd
   ```
2. *Read in the CSV file.*
   ```
   pop=pd.read_csv(’cunyF2016.csv’,skiprows=1)
   ```
3. *Set up a scatter plot.*
   ```
   pop.plot(x="Full-time",y="Part-time")
   ```
4. *Display plot.*
   ```
   plt.show()
   ```

Recap



- Pandas library has elegant solutions for accessing & analyzing structured data.

Recap

pandas

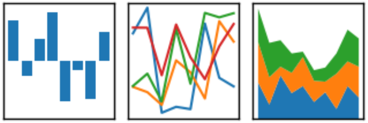
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

- Pandas library has elegant solutions for accessing & analyzing structured data.
- Can manipulate individual columns or rows ('Series').

Recap

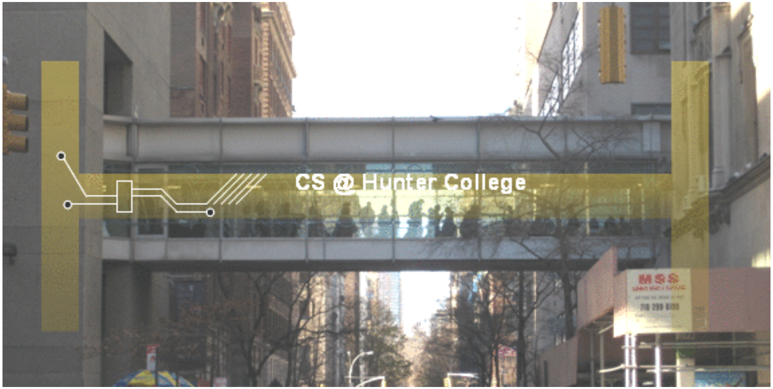
pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



- 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.

Lecture Slips



- On-line lecture slips: `tinyurl.com/yc6j6ubr`

Final Prep

- Starting today, the last 5 minutes of lecture will be on mock final exam questions.

Final Prep

- Starting today, the last 5 minutes of lecture will be on mock final exam questions.
- Pull out a sheet of paper, and do as much as you can before class ends.

Final Prep

- Starting today, the last 5 minutes of lecture will be on mock final exam questions.
- Pull out a sheet of paper, and do as much as you can before class ends.
- Will discuss solutions next lecture.