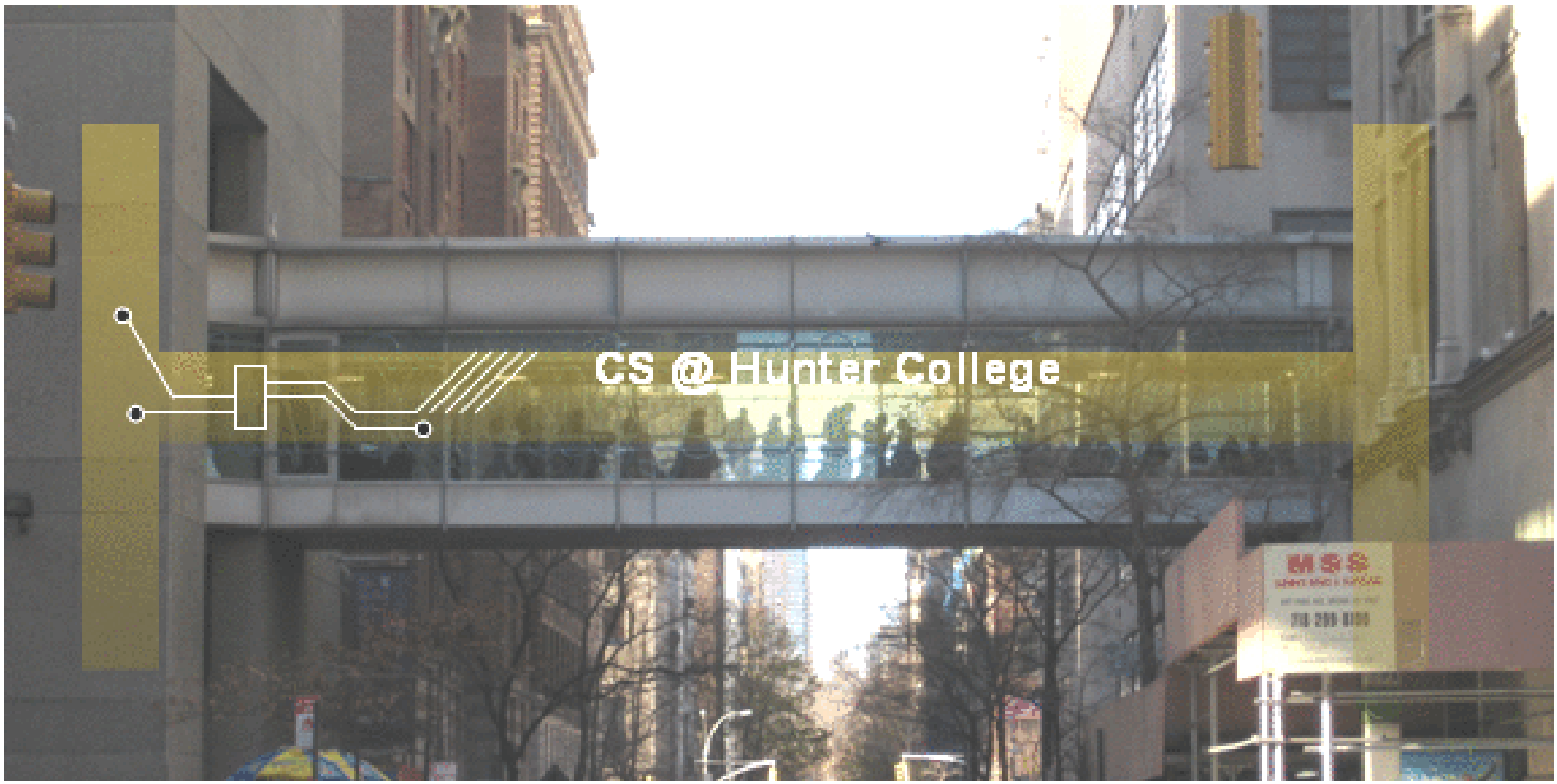
CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Frequently Asked Questions

From email.

Frequently Asked Questions

From email.

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

Frequently Asked Questions

From email.

- **Could you spend more time on circuits / logical expressions / truth tables / decisions?**
  Yes, we will start with that today.

Frequently Asked Questions

From email.

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

Frequently Asked Questions

From email.

- **Could you spend more time on circuits / logical expressions / truth tables / decisions?**
  Yes, we will start with that today.
- **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 email.

- **Could you spend more time on circuits / logical expressions / truth tables / decisions?**
  Yes, we will start with that today.
- **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?**

Frequently Asked Questions

From email.

- **Could you spend more time on circuits / logical expressions / truth tables / decisions?**
  Yes, we will start with that today.
- **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.

Frequently Asked Questions

From email.

- **Could you spend more time on circuits / logical expressions / truth tables / decisions?**
  Yes, we will start with that today.
- **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.
- **Can we have more time for the Lab quiz?**

Frequently Asked Questions

From email.

- **Could you spend more time on circuits / logical expressions / truth tables / decisions?**
  Yes, we will start with that today.
- **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.
- **Can we have more time for the Lab quiz?**
  Yes! After a few of you have mentioned feeling too rushed for the lab quiz, I have increased the time to 20 minutes.

Frequently Asked Questions

From email.

- **Could you spend more time on circuits/logical expressions/truth tables/decisions?**
  Yes, we will start with that today.
- **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.
- **Can we have more time for the Lab quiz?**
  Yes! After a few of you have mentioned feeling too rushed for the lab quiz, I have increased the time to 20 minutes.
  Keep in mind that the final exam will be in the same format and it is also timed. You will have 2 hours for the final exam.

Today's Topics

- Recap: Logical Expressions & Circuits
- Design: Cropping Images
- Accessing Formatted Data

Today's Topics

- **Recap: Logical Expressions & Circuits**
- Design: Cropping Images
- Accessing Formatted Data

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

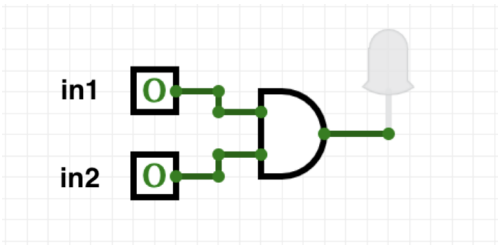
or

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

not

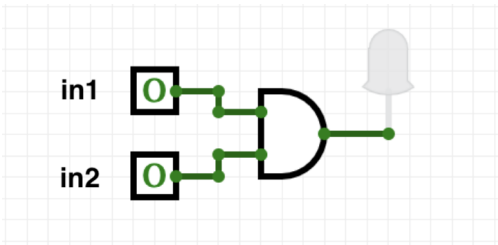
	in1	*returns:*
not	False	True
not	True	False

Logical Operators & Circuits



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

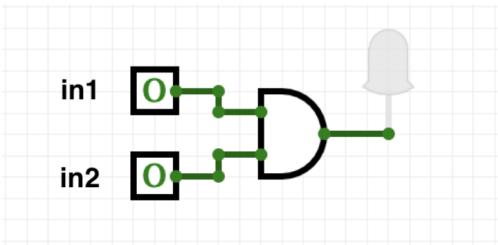
Logical Operators & Circuits



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

  Example: `in1 and in2`

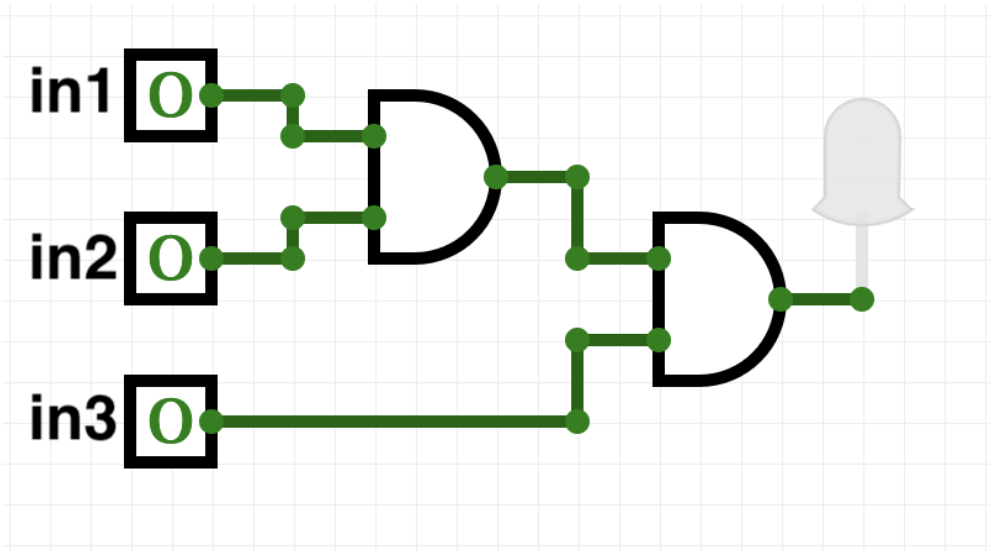
Logical Operators & Circuits



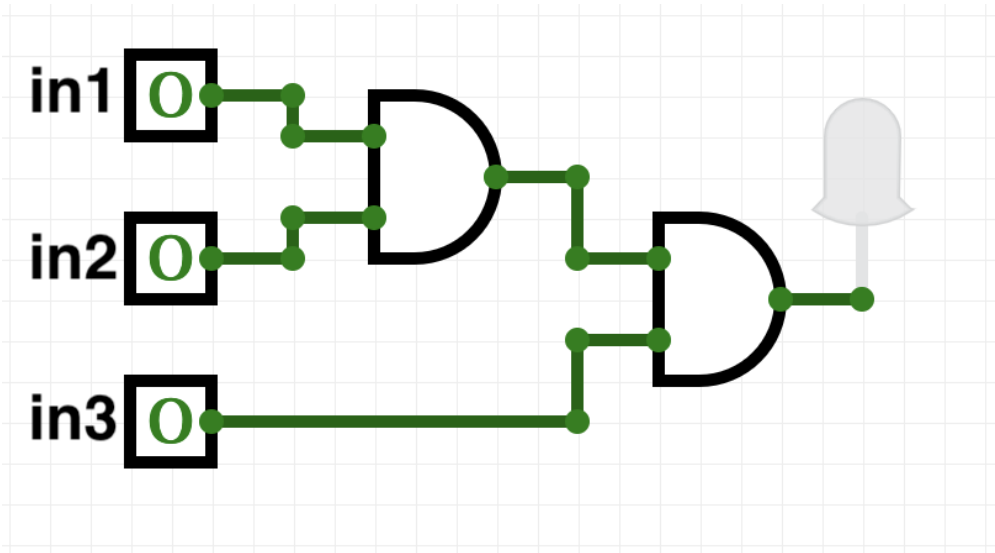
- Each logical operator (`and`, `or`, & `not`) can be used to join together expressions.

  Example: `in1 and in2`
- Each logical operator (`and`, `or`, & `not`) has a corresponding logical circuit that can be used to join together inputs.

Examples: Logical Circuit



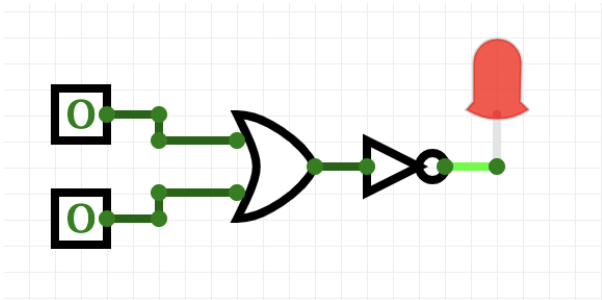
Examples: Logical Circuit

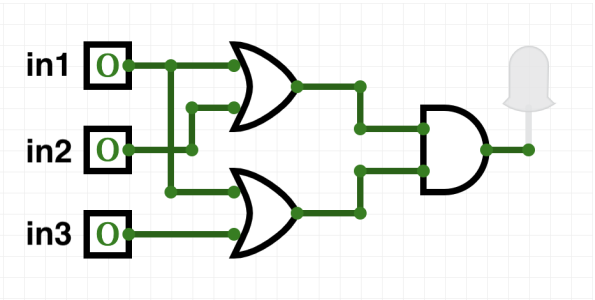


`(in1 and in2) and in3`

More Circuit Examples

Examples from last lecture:





Draw a circuit that corresponds to each logical expression:

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

Challenge:

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 qvBnqa ocUxk"
result = ""
for i in range(len(mess)):
    if i % 3 == 0:
        print(mess[i])
        result = result + mess[i]
print(sports[1], result)
```

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)

Today's Topics

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

Challenge: 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.

Challenge: 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.)*

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 tim expand to a Python program.)

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 tim expand to a Python program.)

How to approach this:

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

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 tim 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

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 tim 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

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 tim 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

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 tim 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.

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 tim 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.

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 tim 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.

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 tim 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.

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 tim 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.

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 tim 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.
6. Display the new image.

In Pairs or Triples: Design Question

1. Import libraries.

In Pairs or Triples: Design Question

1. Import libraries.

```
import matplotlib.pyplot as plt
import numpy as np
```

In Pairs or Triples: Design Question

1. Import libraries.

   ```
   import matplotlib.pyplot as plt
   import numpy as np
   ```

2. Ask user for an image name.

In Pairs or Triples: Design Question

1. Import libraries.

```
import matplotlib.pyplot as plt
import numpy as np
```

2. Ask user for an image name.

```
inF = input('Enter file name: ')
```

In Pairs or Triples: Design Question

1. 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.

In Pairs or Triples: Design Question

1. 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
   ```

In Pairs or Triples: Design Question

1. 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.

In Pairs or Triples: Design Question

1. 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

1. 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
   ```
5. Make a new image that's half the height and half the width.

In Pairs or Triples: Design Question

1. 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
   ```
5. Make a new image that's half the height and half the width.
   ```
   img2 = img[height//2:, :width//2] #Crop to lower left corner
   ```

In Pairs or Triples: Design Question

1. 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
   ```
5. Make a new image that's half the height and half the width.
   ```
   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)
   ```

Today's Topics

Recap: Logical Expressions & Circuits

Design: Cropping Images

Accessing Formatted Data

CS Survey: Astrophysics and astropy

Structured Data

Common to have data structured in a spread sheet.

Structured Data

Common to have data structured in a spread sheet.

In the example above, we have the rst line that says \Undergraduate".

Structured Data

Common to have data structured in a spread sheet.

In the example above, we have the rst line that says \Undergraduate".

Next line has the titles for the columns.

Structured Data

Common to have data structured in a spread sheet.

In the example above, we have the rst line that says \Undergraduate".

Next line has the titles for the columns.

Subsequent lines have a college and attributes about the college.

Structured Data

Common to have data structured in a spread sheet.

In the example above, we have the rst 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

Common to have data structured in a spread sheet.

In the example above, we have the rst 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

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

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

Open source and freely available (part of anaconda distribution).

See Lab 1 for directions on downloading it to your home machine.

Structured Data

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

Open source and freely available (part of anaconda distribution).

See Lab 1 for directions on downloading it to your home machine.

If you can't install on your computer, it is supported in https://repl.it/

Structured Data

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

Open source and freely available (part of anaconda distribution).

See Lab 1 for directions on downloading it to your home machine.

If you can't install on your computer, it is supported in https://repl.it/

To use, add to the top of your program:

```
import pandas as pd
```

CSV Files

Excel .xls files have much extra formatting.

CSV Files

Excel .xls files have much extra formatting.

The text file version is called CSV for comma separated values.

CSV Files

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

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

nycHistPop.csv

Reading in CSV Files

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

Reading in CSV Files

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

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

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

nycHistPop.csv

In Lab 6

Example: Reading in CSV Files

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

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)
```

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()
```

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()
```

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())

Challenge:

Predict what the following will do:

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

Challenge:

Predict what the following will do:

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

Challenge:

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())
```

Challenge:

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")
```

Challenge:

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")
```

Challenge:

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"]
```

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())
Average of values 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())
```
Average of values 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())
Average of values in the column \Staten Island".

print("S I :", pop["Staten Island"].std())
Standard deviation of values 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())
```
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")
```

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())
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".

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())
```
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")
```

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())
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.

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())
```
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"]
```

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())
```
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.

Challenge:

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

cunyF2016.csv

Challenge:

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:

cunyF2016.csv

Challenge:

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.

cunyF2016.csv

Challenge:

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.

cunyF2016.csv

Challenge:

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.

cunyF2016.csv

Challenge:

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.

cunyF2016.csv

Challenge:

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:

cunyF2016.csv

Challenge:

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.

cunyF2016.csv

Challenge:

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
   ```

cunyF2016.csv

Challenge:

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.

cunyF2016.csv

Challenge:

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)
   ```

Challenge:

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.

Challenge:

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.scatter(x="Full-time",y="Part-time")
   ```

Challenge:

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.scatter(x="Full-time",y="Part-time")
   ```
4. Display plot.

Challenge:

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.scatter(x="Full-time",y="Part-time")
   ```
4. Display plot.
   ```
   plt.show()
   ```

groupby()

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

AustraliaRain.csv

groupby()

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 :

AustraliaRain.csv

groupby()

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 :

1. Import libraries.
   ```
   import pandas as pd
   ```

AustraliaRain.csv

groupby()

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 :

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

AustraliaRain.csv

groupby()

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 :

AustraliaRain.csv

1. 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.
   ```
   groupAvg = rain.groupby('Location')
   ```

groupby()

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 :

AustraliaRain.csv

1. 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.
   ```
   groupAvg = rain.groupby('Location')
   ```
4. Print the average rainfall at each location.
   ```
   print(groupAvg['Rainfall'].mean())
   ```

groupby()

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 :

AustraliaRain.csv

1. 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.
   ```
   groupAvg = rain.groupby('Location')
   ```
4. Print the average rainfall at each location.
   ```
   print(groupAvg['Rainfall'].mean())
   ```

groupby()

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. Albury :

AustraliaRain.csv

groupby()

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. Albury :

AustraliaRain.csv

1. 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 data for group Albury.
   ```
   AlburyAvg = rain.groupby('Location').get_group('Albury')
   ```

groupby()

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. Albury :

AustraliaRain.csv

1. 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 data for group Albury.
   ```
   AlburyAvg = rain.groupby('Location').get_group('Albury')
   ```
4. Print the average rainfall in Albury.
   ```
   print(AlburyAvg['Rainfall'].mean())
   ```

groupby()

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. Albury :

AustraliaRain.csv

1. 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 data for group Albury.
   ```
   AlburyAvg = rain.groupby('Location').get_group('Albury')
   ```
4. Print the average rainfall in Albury.
   ```
   print(AlburyAvg['Rainfall'].mean())
   ```

Design Challenge

Design an algorithm that:

- Prints the luminosity of the brightest star.
- Prints the temperature of the coldest star.
- Prints the average radius of a Hypergiant.

Design Challenge - Solution

Libraries: pandas

Design Challenge - Solution

Libraries: pandas

Process:

- Print max of 'Luminosity' column

Design Challenge - Solution

Libraries: pandas

Process:

- Print max of 'Luminosity' column
- Print min of 'Temperature' column

Design Challenge - Solution

Libraries: pandas

Process:

- Print max of 'Luminosity' column
- Print min of 'Temperature' column
- groupby 'Star Type' and take averages, then print max of 'Radius' column

Design Challenge - Solution

Libraries: pandas

Process:

- Print max of 'Luminosity' column
- Print min of 'Temperature' column
- groupby 'Star Type' and take averages, then print max of 'Radius' column
- OR groupby 'Star Type' and get group 'Hypergiant' to print average 'Radius'

Design Challenge - Code

Libraries: pandas

```
import pandas as pd
stars = pd.read_csv('Stars.csv')
```

Design Challenge - Code

Libraries: pandas

```
import pandas as pd
stars = pd.read_csv('Stars.csv')
```

Process:

- Print max of 'Luminosity' column

```
print(stars['Luminosity(L/Lo)'].max())
```

Design Challenge - Code

Libraries: pandas

```
import pandas as pd
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
  ```
  print( stars['Temperature (K)'].min())
  ```

Design Challenge - Code

Libraries: pandas

```
import pandas as pd
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
  ```
  print( stars['Temperature (K)'].min())
  ```
- groupby 'Star Type' and take averages, then print max of 'Radius' column
  ```
  print(stars.groupby('Star type')          n
  .mean()['Radius(R/Ro)'].max())
  ```

Design Challenge - Code

Libraries: pandas

```
import pandas as pd
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
  ```
  print( stars['Temperature (K)'].min())
  ```
- OR groupby 'Star Type' and get group 'Hypergiant' to print average 'Radius'
  ```
  print(stars.groupby('Star type') n
  .get_group('Hypergiant').mean()['Radius(R/Ro)'])
  ```

Recap

Recap: Logical Expressions & Circuits

Recap

Recap: Logical Expressions & Circuits

Accessing Formatted Data:

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

Recap

- Recap: Logical Expressions & Circuits
- Accessing Formatted Data:
  - Pandas library has elegant solutions for accessing & analyzing structured data.
  - Can manipulate individual columns or rows ('Series').

Recap

- 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.