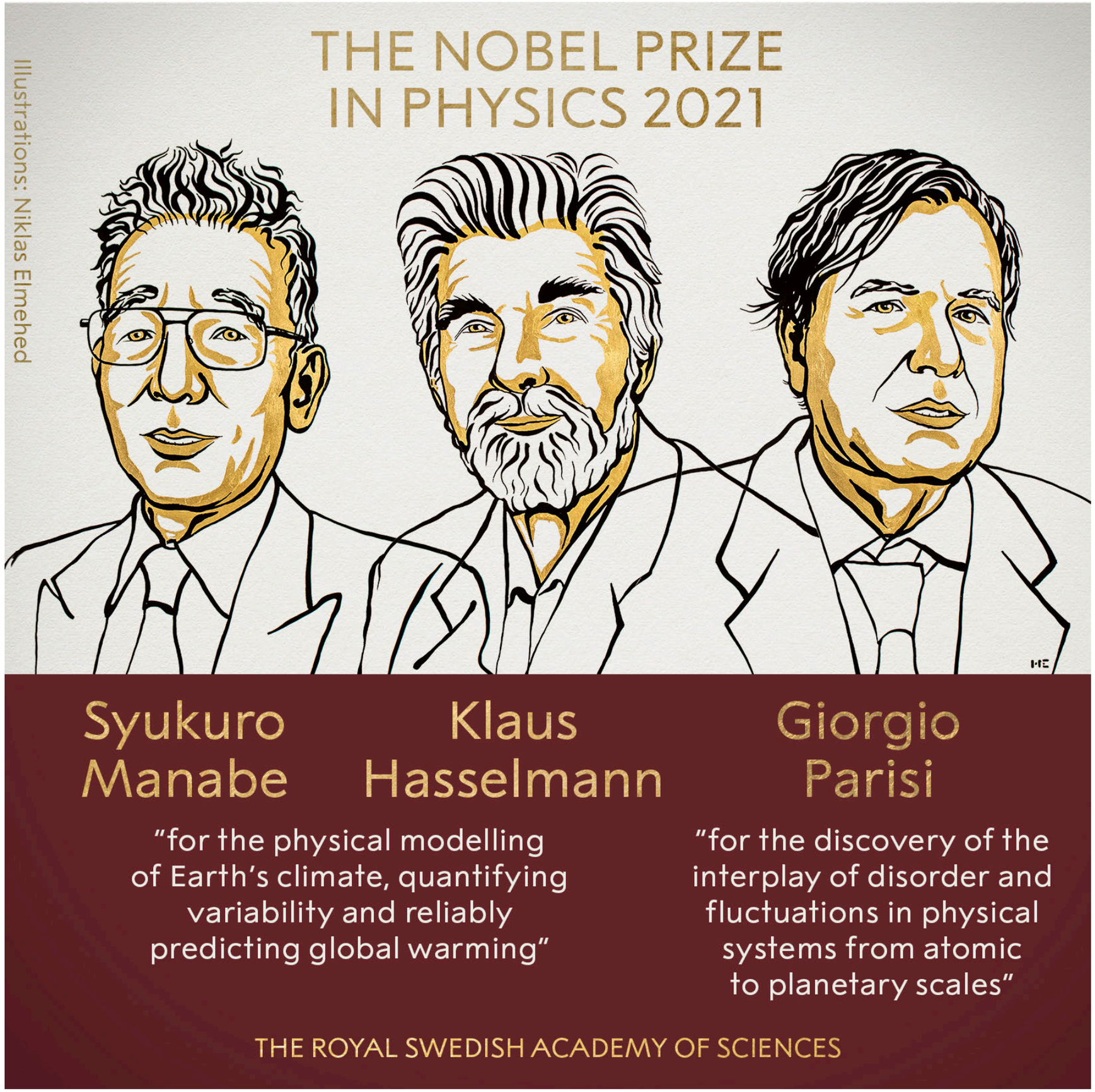
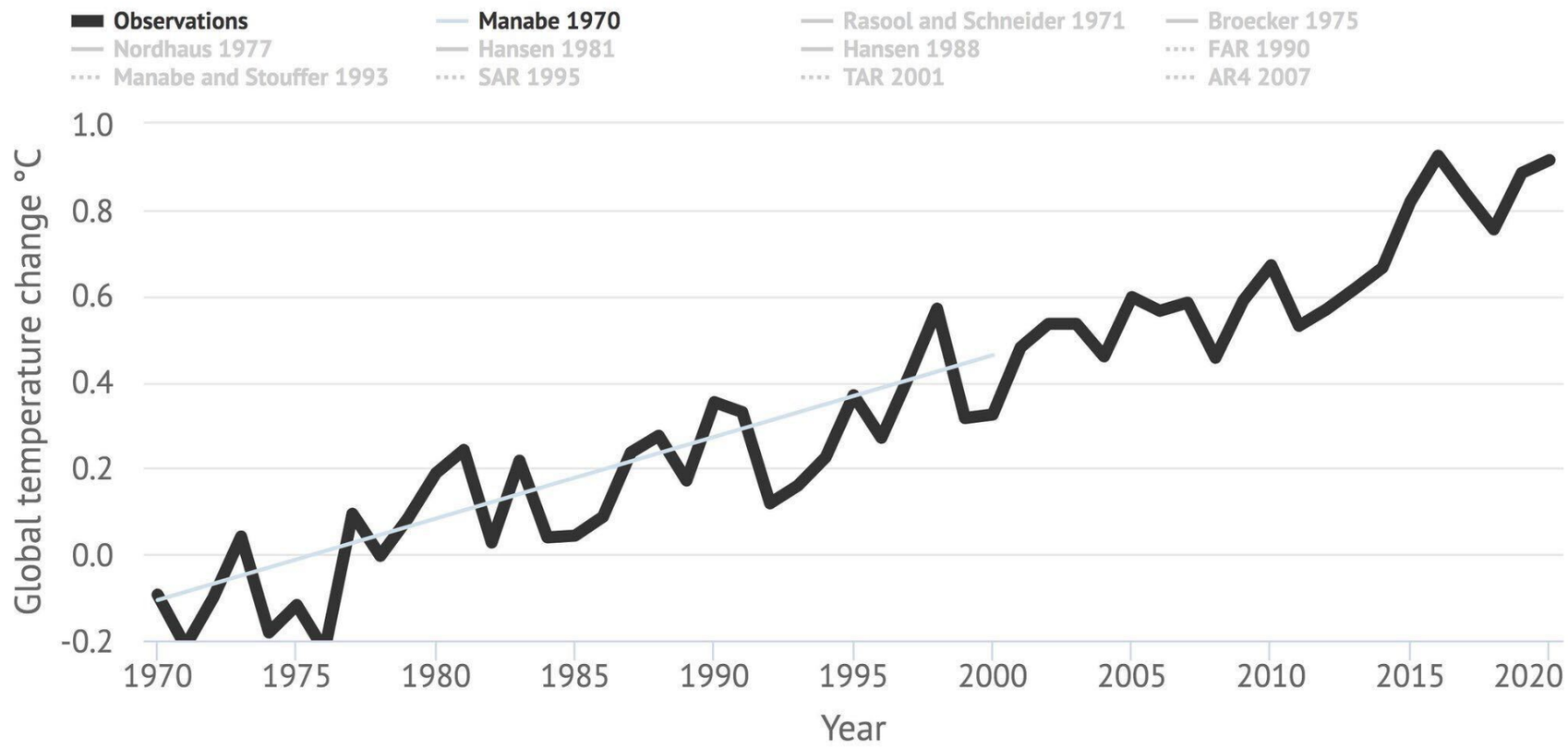
ASTR 119: Session 4
Basic Python Programming

Outline

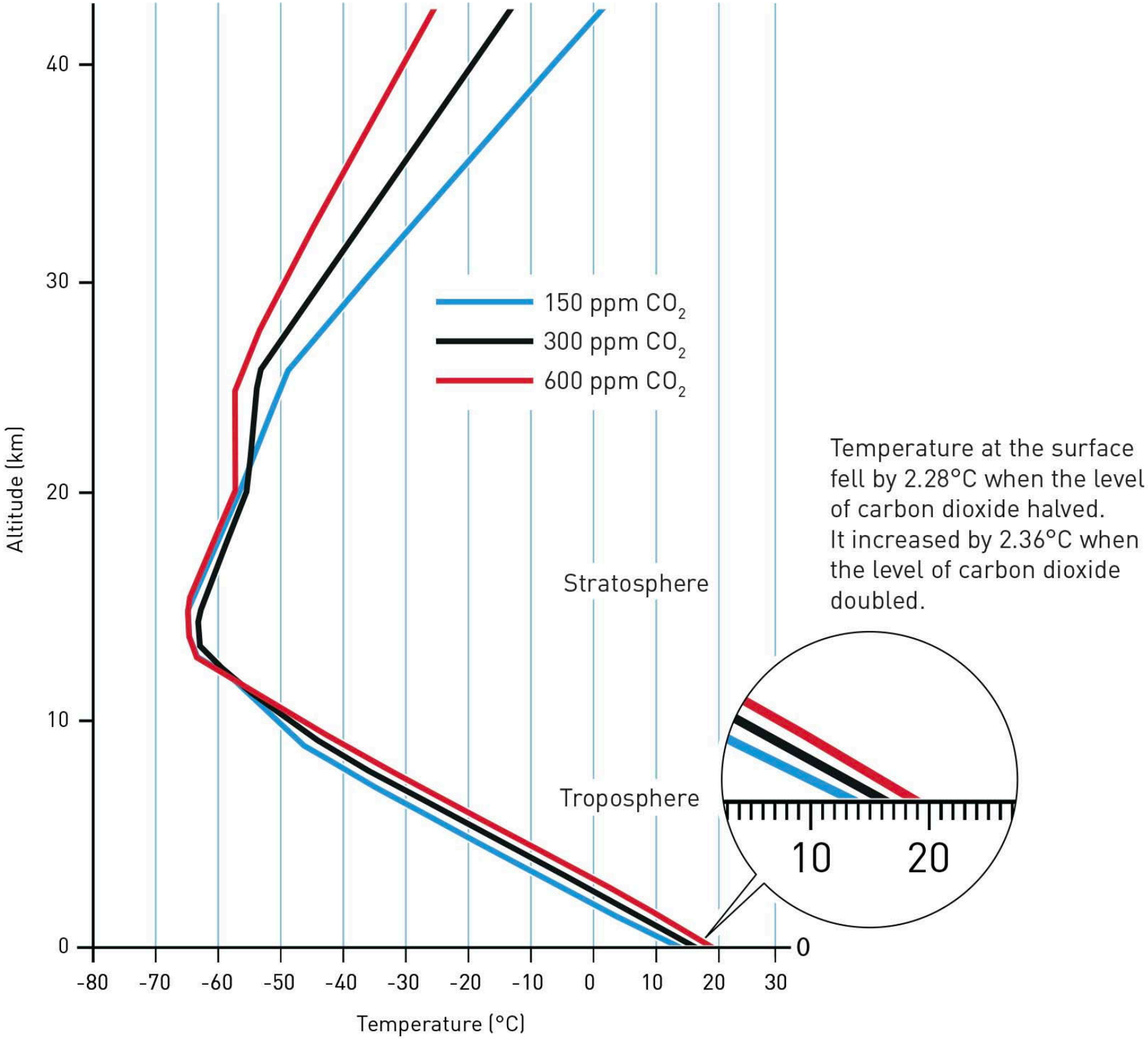
1) ~~Visualization of the Day~~ Nobel Prize in Physics
2) Reminder: Homework Due Oct 15
3) Dictionaries
4) Exceptions
5) Modules
6) Getting Help
7) Floating Point Numbers
8) Numpy
9) Saving Your Work


THE NOBEL PRIZE
IN PHYSICS 2021
Illustrations: Niklas Elmehed
Syukuro Manabe
Klaus Hasselmann
Giorgio Parisi
"for the physical modelling of Earth's climate, quantifying variability and reliably predicting global warming"
"for the discovery of the interplay of disorder and fluctuations in physical systems from atomic to planetary scales"
THE ROYAL SWEDISH ACADEMY OF SCIENCES

Models of global climate have accurately predicted the rise of global temperatures since 1970.



Models of global climate have taught us about the direct connection between carbon dioxide atmospheric abundance and surface temperatures on Earth (this model is from 1967).



Temperature at the surface fell by 2.28°C when the level of carbon dioxide halved. It increased by 2.36°C when the level of carbon dioxide doubled.

We can celebrate the accomplishments of these scientists and also be critical of inequities in the system by which they are selected.

Nobel laureates are not representative of the global community of physicists. The vast majority of Nobel laureates in Physics are white, cis men.

Homework, due Oct 7, 9:50am

1) Write all the example programs from Session 2 and 3 and make a **folder** called astr-119-hw-2 **in your astr-119 repository** that contains all the python files named as follows:

   a) variables_and_loops.py: The program on slide 12 of Session 2 PDF.
   b) while.py : The program on slide 14 of Session 2 PDF.
   c) operators.py : The program on slide 15 of Session 2 PDF.
   d) functions.py : The program on slide 9 of Session 3 PDF.
   e) if_else_control.py : The program on slide 11 of Session 3 PDF.
   f) data_types.py : The program on slide 13 of Session 3 PDF.
   g) data_types_continued.py : The program on slide 15 of Session 3 PDF.
   h) python_arrays.py : The program on slide 17 of Session 3 PDF.
   i) dictionaries.py : The program on slide 19 of Session 3 PDF.
   j) exceptions.py : The program on slide 21 of Session 3 PDF.
   k) useful_modules.py : The program on slide 25 of Session 3 PDF.

2) Create an issue for your repository and tag your TA, asking them to grade your homework.

3) Your TA will branch your code and email you commented version of the code and a grade. To get the full grade possible, all the programs will need to reproduce the functionality shown in the slides.

Python Dictionaries

```
1  #define a dictionary data structure
2
3  #dictionaries have key : value for the elements
4  example_dict = {
5      "class"          :   "Astr 119",
6      "prof"           :   "Brant",
7      "awesomeness"    :   10
8  }
9  print(type(example_dict))   #will say dict
10
11 #get a value via key
12 course = example_dict["class"]
13 print(course)
14
15 #change a value via key
16 example_dict["awesomeness"] += 1     #increase awesomeness
17
18 #print the dictionary
19 print(example_dict)
20
21 #print dictionary element by element
22 for x in example_dict.keys():
23     print(x,example_dict[x])
24
```

Python Dictionaries

1. brant@pewter.lan: /Users/brant/Desktop/classes/astr_119/Session 2 (bash)

```
[18:42:18][brant@pewter:~/Desktop/classes/astr_119/Session 2]$ python3 python_dictionaries.py
<class 'dict'>
Astr 119
{'class': 'Astr 119', 'prof': 'Brant', 'awesomeness': 11}
class Astr 119
prof Brant
awesomeness 11
[18:42:21][brant@pewter:~/Desktop/classes/astr_119/Session 2]$
```

Python Exceptions

```
1  #python exceptions let you deal with
2  #unexpected results
3
4  try:
5      print(a)    #this will throw an exception since a is not defined
6  except:
7      print("a is not defined!")
8
9  #there are specific errors to help with cases
10 try:
11     print(a)    #this will throw an exception since a is not defined
12 except NameError:
13     print("a is still not defined!")
14 except:
15     print("Something else went wrong.")
16
17 #this will break our program
18 #since a is not defined
19 print(a)
```

Python Exceptions

1. brant@pewter.lan: /Users/brant/Desktop/classes/astr_119/Session 2 (bash)

```
[21:21:27][brant@pewter:~/Desktop/classes/astr_119/Session 2]$ python3 python_exceptions.py
a is not defined!
a is still not defined!
Traceback (most recent call last):
  File "python_exceptions.py", line 19, in <module>
    print(a)
NameError: name 'a' is not defined
[21:21:36][brant@pewter:~/Desktop/classes/astr_119/Session 2]$
```

Python Modules

Define a module

test_module.py

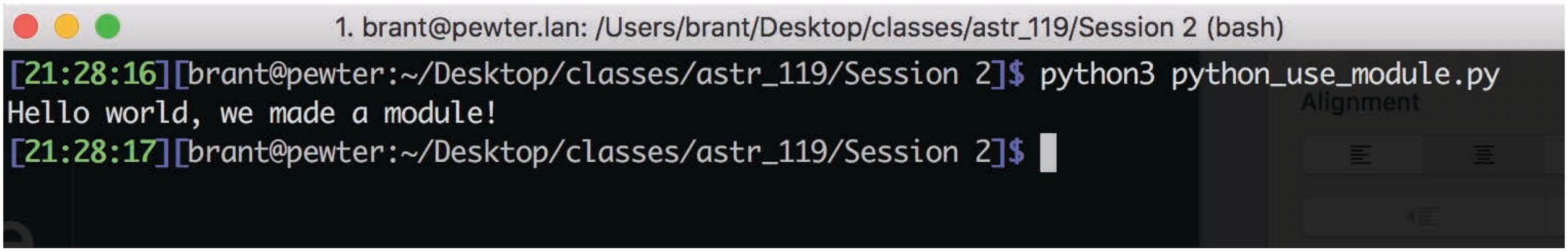
```
1  def hello_world():
2      print("Hello world, we made a module!")
```

Use a module

python_use_module.py

```
1  import test_module as tm
2
3  tm.hello_world()
4
```

Python Modules



Useful Modules

```python
import numpy as np                      #the Numpy library
import matplotlib.pyplot as plt         #Matplotlib's pyplot

import sys                              #gives access to a C-like sys library
import os                               #gives access to operating system


print(sys.argv)     #prints any command line arguments, incl program name
print(os.getcwd())  #prints the current working directory
```

```
1. brant@pewter.lan: /Users/brant/Desktop/classes/astr_119/Session 2 (bash)
[21:40:32][brant@pewter:~/Desktop/classes/astr_119/Session 2]$ python3 python_useful_modules.py
['python_useful_modules.py']
/Users/brant/Desktop/classes/astr_119/Session 2
[21:40:34][brant@pewter:~/Desktop/classes/astr_119/Session 2]$
```

https://docs.python.org/3/library/

Getting Help

```
1  import os
2
3  help(os)
```

1. brant@pewter.lan: /Users/brant/Desktop/classes/astr_119/Session 2 (less)

```
Help on module os:

NAME
    os - OS routines for NT or Posix depending on what system we're on.

MODULE REFERENCE
    https://docs.python.org/3.6/library/os

    The following documentation is automatically generated from the Python
    source files.  It may be incomplete, incorrect or include features that
    are considered implementation detail and may vary between Python
    implementations.  When in doubt, consult the module reference at the
    location listed above.

DESCRIPTION
    This exports:
      - all functions from posix or nt, e.g. unlink, stat, etc.
      - os.path is either posixpath or ntpath
      - os.name is either 'posix' or 'nt'
      - os.curdir is a string representing the current directory (always '.')
      - os.pardir is a string representing the parent directory (always '..')
```

Bits, Information, and Precision

We've seen a reference to "bits" in terms of the precision or information content of different data types. What are bits, and how do they encode information and precision?

1. brant@pewter.lan: /Users/brant/Desktop/classes/astr_119/Session 2 (bash)

```
[18:15:34][brant@pewter:~/Desktop/classes/astr_119/Session 2]$ python3 data_types.py
<class 'int'>
<class 'numpy.ndarray'>
<class 'numpy.int64'>
<class 'float'>
<class 'float'>
<class 'numpy.ndarray'>
<class 'numpy.float64'>
[18:15:53][brant@pewter:~/Desktop/classes/astr_119/Session 2]$
```

A bit is a single binary digit (0 or 1). A byte is 8 bits.

Floating Point Numbers

A bit is a single binary digit (0 or 1). A byte is 8 bits.

During the first Gulf War, the timing in the reference clocks of Patriot missile systems were 24 bit precision floating point numbers that represented 0.1 second increments.

"0.1" is irrational in binary bits, so clipping at 24 bits led to small timing errors that accrued over time.

After 100 hours of operations, clocks were off by 0.34s. Scud missiles travel ~500 meters in this time, so you get the picture.

http://ta.twi.tudelft.nl/users/vuik/wi211/disasters.html

Floating Point Numbers

A bit is a single binary digit (0 or 1). A byte is 8 bits.

In 1996, on its first launch an uncrewed Ariane 5 rocket failed after launch. The rocket and cargo cost $500 million alone, with the program costing $7 billion.

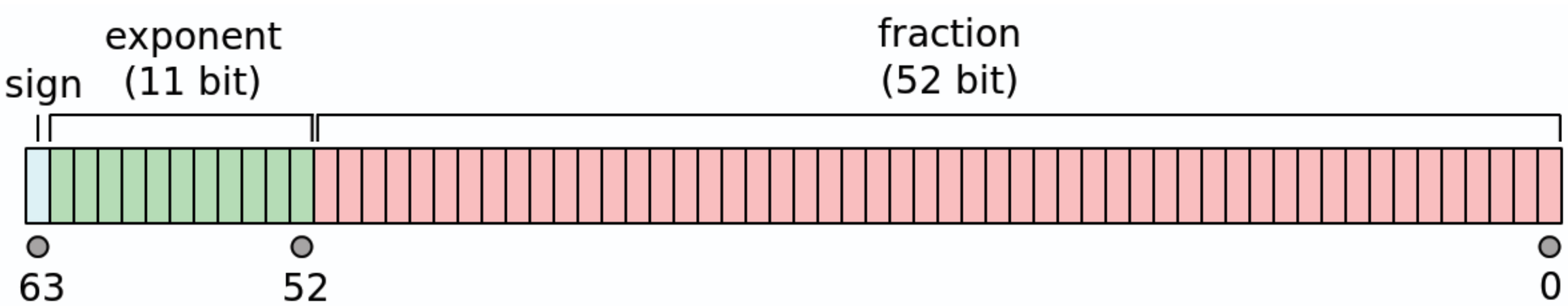
The cause of the failure was the conversion of a 64-bit floating point number to a 16-bit signed integer.

http://ta.twi.tudelft.nl/users/vuik/wi211/disasters.html

Double Precision

A bit is a single binary digit (0 or 1). A byte is 8 bits.

A we call 4 byte variables "single" precision and 8 byte variables "double" precision. A float64 holds eight bytes:



$$\text{value} = (-1)^{\text{sign}} \left(1 + \sum_{i=1}^{52} b_{52-i} 2^{-i}\right) \times 2^{e-1023}$$

wikipedia

Double Precision

A bit is a single binary digit (0 or 1). A byte is 8 bits.

A float64 holds eight bytes:

A float64 has 52 binary digits that represent its significand. Unless the bits in the significand are all zero, an implicit bit allows for 53 binary digits.

The precision is therefore ~ $\log_{10}(2^{53})$ ~ 15.955 decimals.

IEEE 754 standard specifies 15 or 17 decimals of precision, depending on the conversion.

Numpy

NumPy

Scipy.org

NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

NumPy is licensed under the BSD license, enabling reuse with few restrictions.

About NumPy
License
Code of Conduct
Old array packages

Getting Started

- Getting NumPy
- Installing the SciPy Stack
- NumPy and SciPy documentation page
- NumPy Tutorial
- NumPy for MATLAB© Users
- NumPy functions by category
- NumPy Mailing List

For more information on the SciPy Stack (for which NumPy provides the fundamental array data structure), see scipy.org.

http://www.numpy.org/

Using pip to install numpy

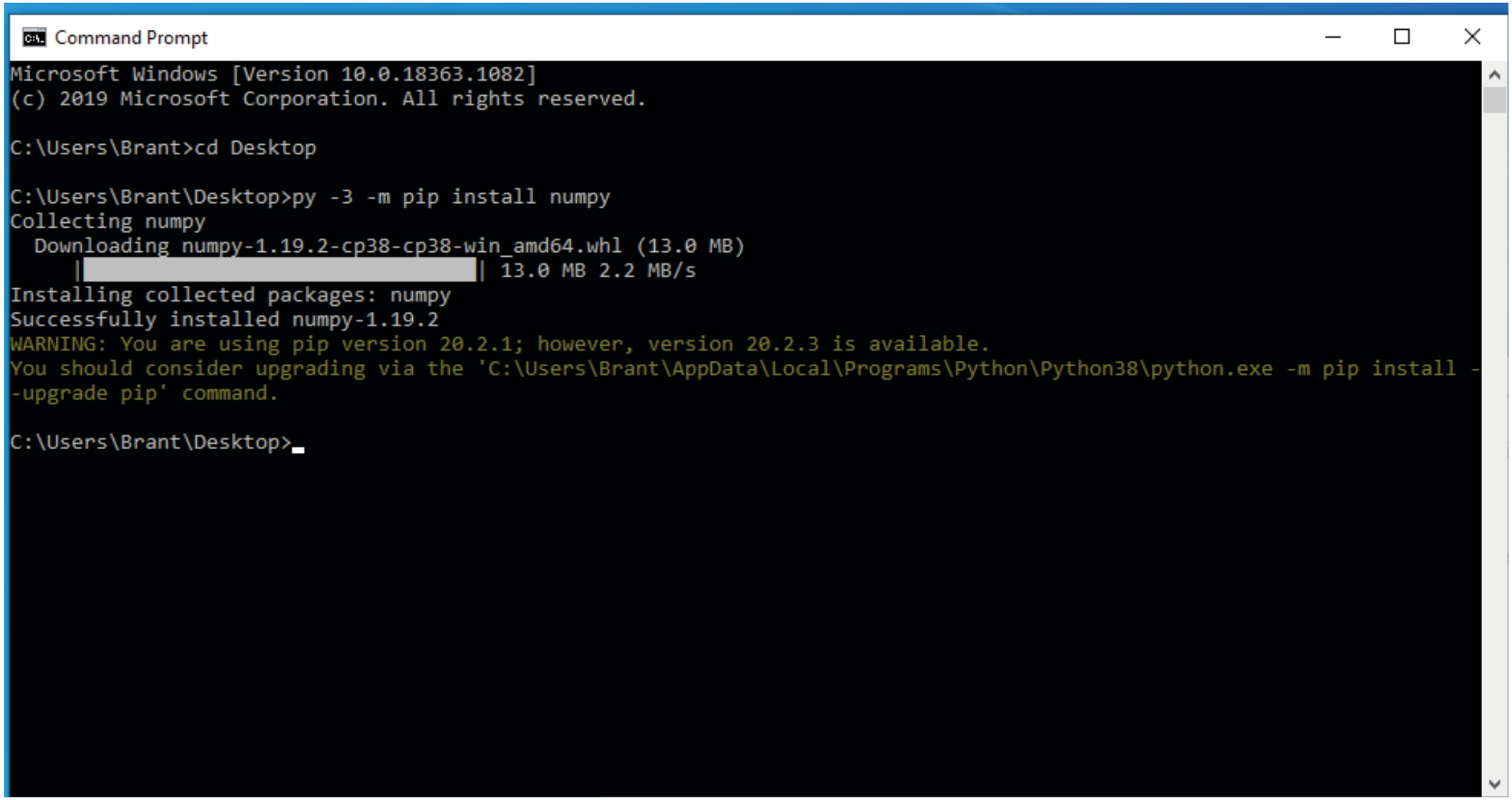
Mac OSX / Linux:

```
$ pip3 install numpy
```

Windows:

```
$ py -3   -m pip install numpy
```

Installation on Windows



Numpy

```python
import numpy as np

x = 1.0      #define a float
y = 2.0      #define another float

#trigonometry
print(np.sin(x))          #sin(x)
print(np.cos(x))          #cos(x)
print(np.tan(x))          #tan(x)
print(np.arcsin(x))       #arcsin(x)
print(np.arccos(x))       #arccos(x)
print(np.arctan(x))       #arctan(x)
print(np.arctan2(x,y))    #arctan(x/y)
print(np.rad2deg(x))      #convert rad to deg

#hyperbolic functions
print(np.sinh(x))         #sinh(x)
print(np.cosh(x))         #cosh(x)
print(np.tanh(x))         #tanh(x)
print(np.arcsinh(x))      #arcsinh(x)
print(np.arccosh(x))      #arccosh(x)
print(np.arctanh(x))      #arctanh(x)
```

Numpy

1. brant@eduroam-169-233-222-100.ucsc.edu: /Users/brant/Desktop (bash)

```
[16:47:13][brant@eduroam-169-233-222-100:~/Desktop]$ python demo_numpy.py
0.841470984808
0.540302305868
1.55740772465
1.57079632679
0.0
0.785398163397
0.463647609001
57.2957795131
1.17520119364
1.54308063482
0.761594155956
0.88137358702
0.0
demo_numpy.py:22: RuntimeWarning: divide by zero encountered in arctanh
  print(np.arctanh(x))  #arctanh(x)
inf
```

Numpy Continued

```
 1 import numpy as np
 2
 3 x = 1.0      #define a float
 4 y = 2.0      #define another float
 5
 6 #exponents and logarithms
 7 print(np.exp(x))          #e^x
 8 print(np.log(x))          #ln x
 9 print(np.log10(x))        #log_10 x
10 print(np.log2(x))         #log_2 x
11
12 #min/max/misc
13 print(np.fabs(x))         #absolute val as a float
14 print(np.fmin(x,y))       #min of x and y
15 print(np.fmax(x,y))       #max of x and y
16
17 #populate arrays
18 n = 100                              #define an int
19 z = np.arange(n,dtype=float)         #get an array [0.0,n-1.]
20 z *= 2.0*np.pi /float(n-1)           #z = [0,2*pi]
21 sin_z = np.sin(z)                    #get an array sin(z)
22
23 #interpolation
24 print(np.interp(0.75,z,sin_z))  #interpolate sin(0.75)
25 print(np.sin(0.75))
```

Numpy, Continued

```
1. brant@eduroam-169-233-222-100.ucsc.edu: /Users/brant/Desktop (bash)
[16:48:23][brant@eduroam-169-233-222-100:~/Desktop]$ python demo_numpy_continued
.py
2.71828182846
0.0
0.0
0.0
1.0
1.0
2.0
0.681436721777
0.681638760023
[16:48:26][brant@eduroam-169-233-222-100:~/Desktop]$
```

Save Your Work

Make a GitHub project “astr-119-session-4”, and commit the program files you made today.