Algorithms and Data Structures for Data Science
Introduction and Python Basics

CS 277

Brad Solomon

January 18, 2023

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Department of Computer Science

Learning Objectives

Get to know each other through brief introductions

Discuss class logistics and expectations

Review programming and Python fundamentals

Who am I?

Brad Solomon

Teaching Assistant Professor, Computer Science

2233 Siebel Center for Computer Science

Email: bradsol@illinois.edu

Office Hours:

Thursdays, 11:00 - 12:00 PM

(Details are on the website)

… can also make an appointment directly

Who are you?

Take a moment to introduce yourself to your neighbors!

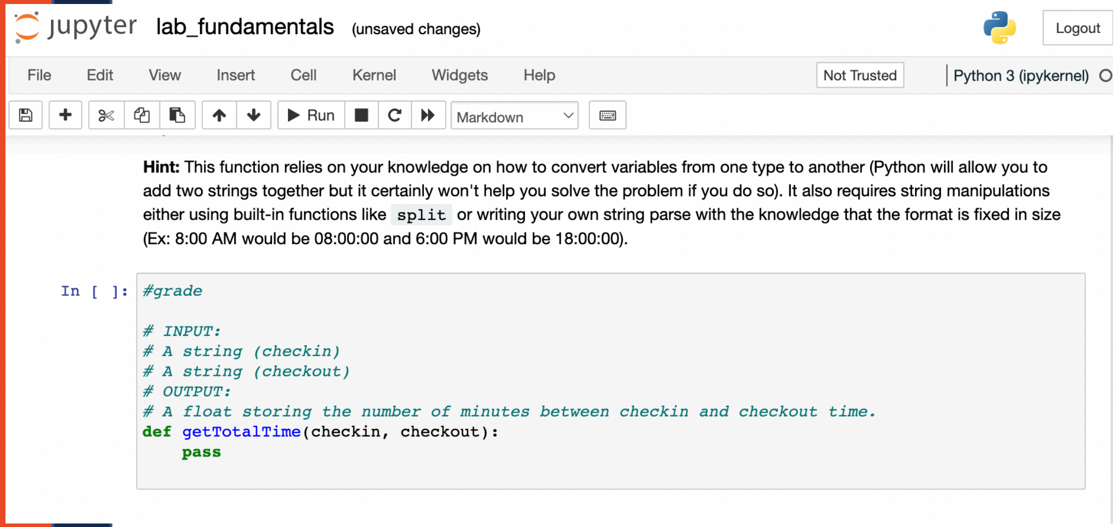
Feel free to introduce yourself on Piazza:

https://piazza.com/class/l6z8qmgyvblga/

Stop by my office hours at some point this semester!

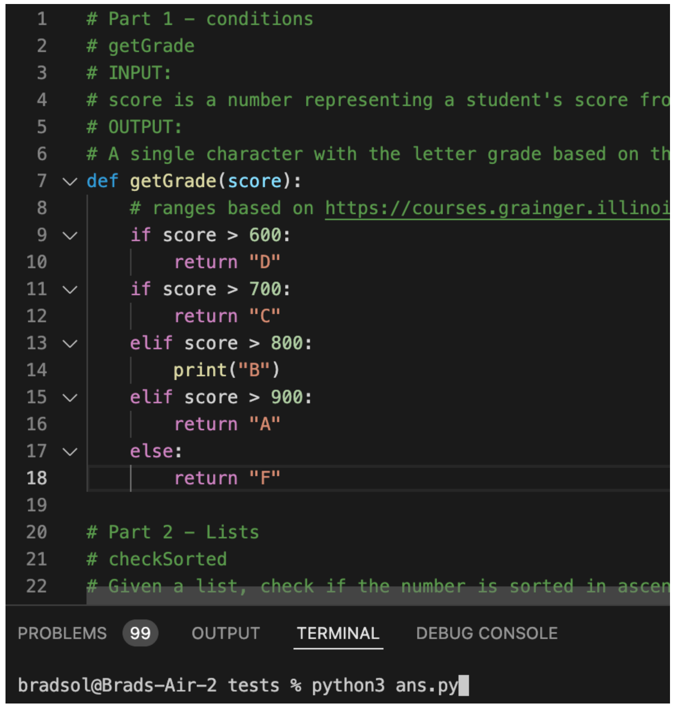
What will you get out of this class?

Navigate, organize, and run moderately complex Python projects



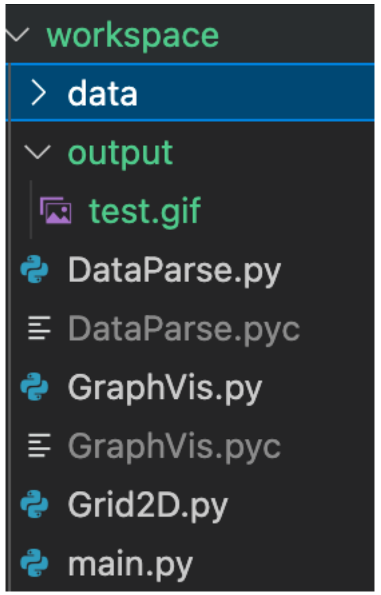
Hint: This function relies on your knowledge on how to convert variables from one type to another (Python will allow you to add two strings together but it certainly won't help you solve the problem if you do so). It also requires string manipulations either using built-in functions like `split` or writing your own string parse with the knowledge that the format is fixed in size (Ex: 8:00 AM would be 08:00:00 and 6:00 PM would be 18:00:00).

```
In [ ]: #grade

        # INPUT:
        # A string (checkin)
        # A string (checkout)
        # OUTPUT:
        # A float storing the number of minutes between checkin and checkout time.
        def getTotalTime(checkin, checkout):
            pass
```



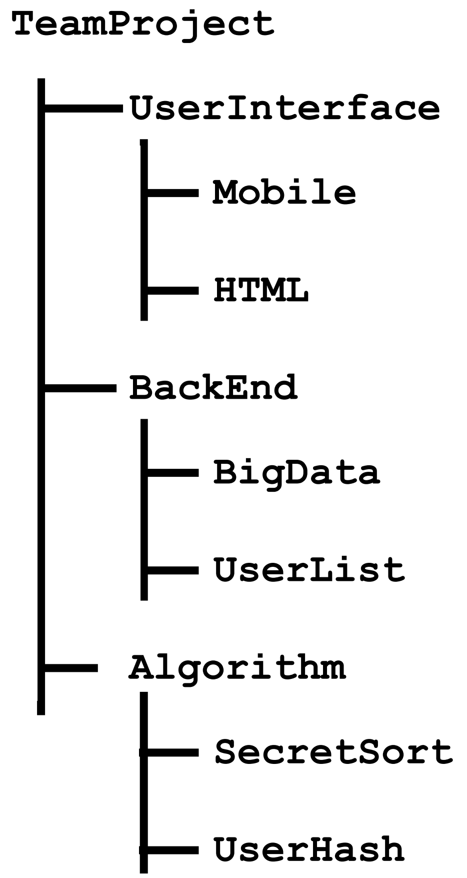
```
# Part 1 - conditions
# getGrade
# INPUT:
# score is a number representing a student's score fro
# OUTPUT:
# A single character with the letter grade based on th
def getGrade(score):
    # ranges based on https://courses.grainger.illinoi
    if score > 600:
        return "D"
    if score > 700:
        return "C"
    elif score > 800:
        print("B")
    elif score > 900:
        return "A"
    else:
        return "F"

# Part 2 - Lists
# checkSorted
# Given a list, check if the number is sorted in ascen
```

```
bradsol@Brads-Air-2 tests % python3 ans.py
```

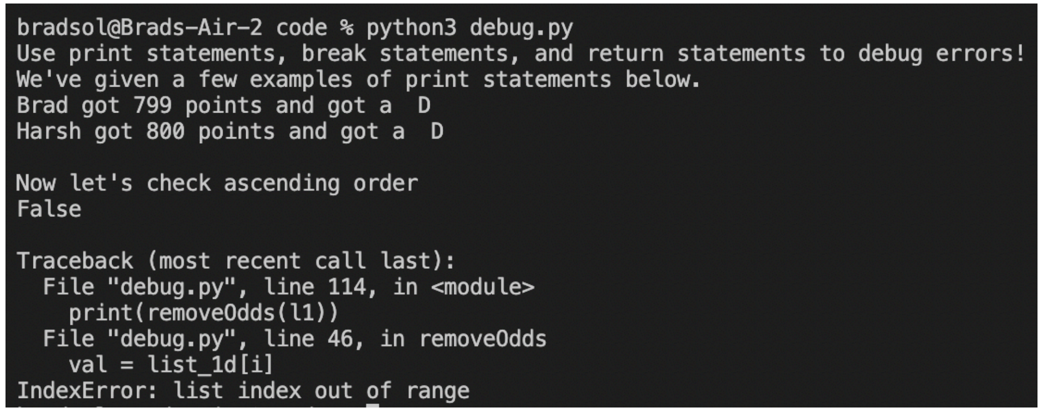


Why should you care?

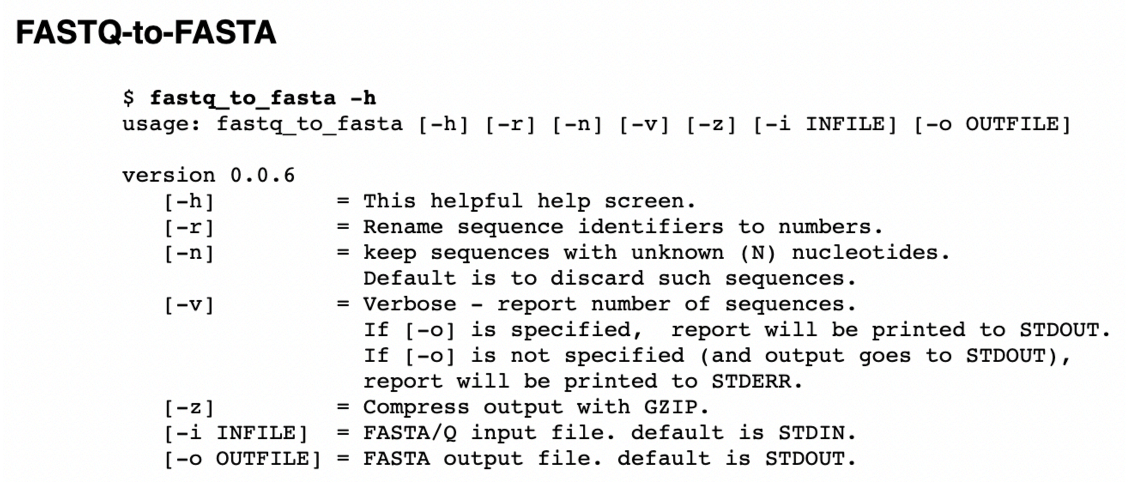
Navigate, organize, and run moderately complex Python projects



```
#include <vector>

#include "util/coloredout.h"

#include "cs225/point.h"

using std::vector;
using std::string;
using std::ostream;
using std::cout;
using std::endl;
```



```
bradsol@Brads-Air-2 code % python3 debug.py
Use print statements, break statements, and return statements to debug errors!
We've given a few examples of print statements below.
Brad got 799 points and got a  D
Harsh got 800 points and got a  D

Now let's check ascending order
False

Traceback (most recent call last):
  File "debug.py", line 114, in <module>
    print(removeOdds(l1))
  File "debug.py", line 46, in removeOdds
    val = list_1d[i]
IndexError: list index out of range
```



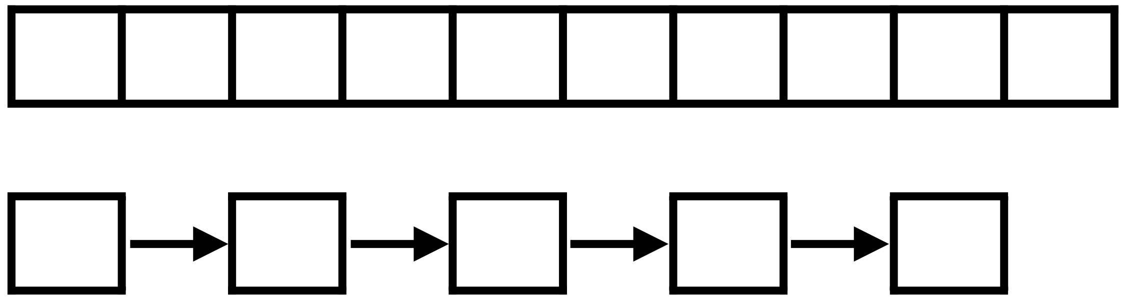
FASTQ-to-FASTA

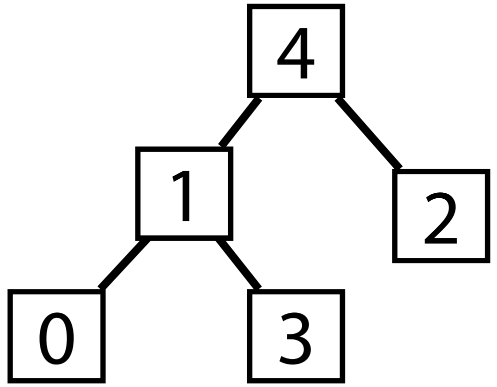
```
$ fastq_to_fasta -h
usage: fastq_to_fasta [-h] [-r] [-n] [-v] [-z] [-i INFILE] [-o OUTFILE]

version 0.0.6
   [-h]         = This helpful help screen.
   [-r]         = Rename sequence identifiers to numbers.
   [-n]         = keep sequences with unknown (N) nucleotides.
                  Default is to discard such sequences.
   [-v]         = Verbose - report number of sequences.
                  If [-o] is specified,  report will be printed to STDOUT.
                  If [-o] is not specified (and output goes to STDOUT),
                  report will be printed to STDERR.
   [-z]         = Compress output with GZIP.
   [-i INFILE]  = FASTA/Q input file. default is STDIN.
   [-o OUTFILE] = FASTA output file. default is STDOUT.
```

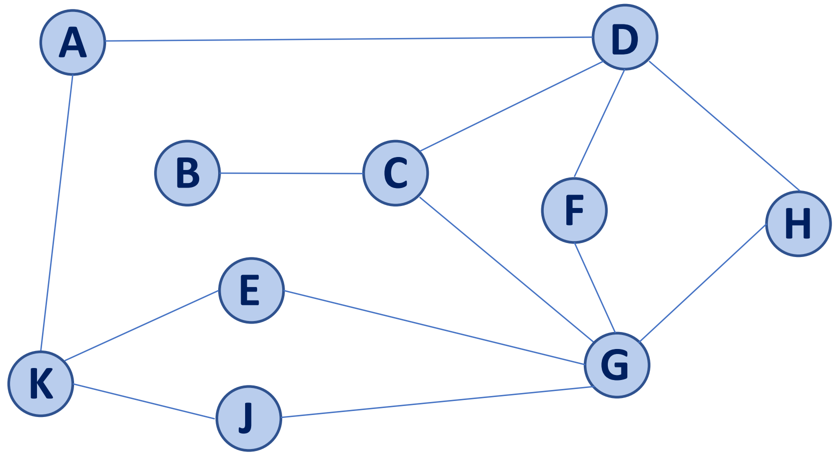
Taken from FastX-Toolkit (hannonlab.cshl.edu)

What will you get out of this class?

Understand foundational data structures and algorithms

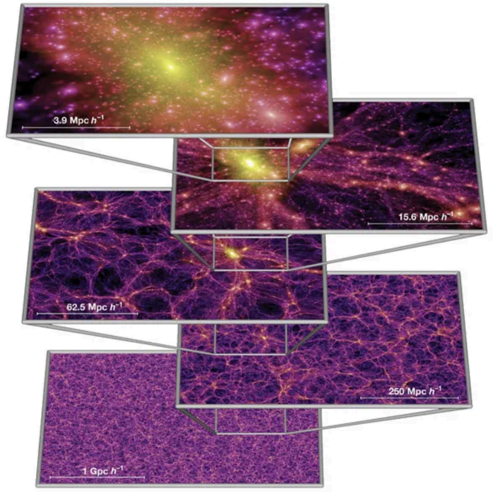




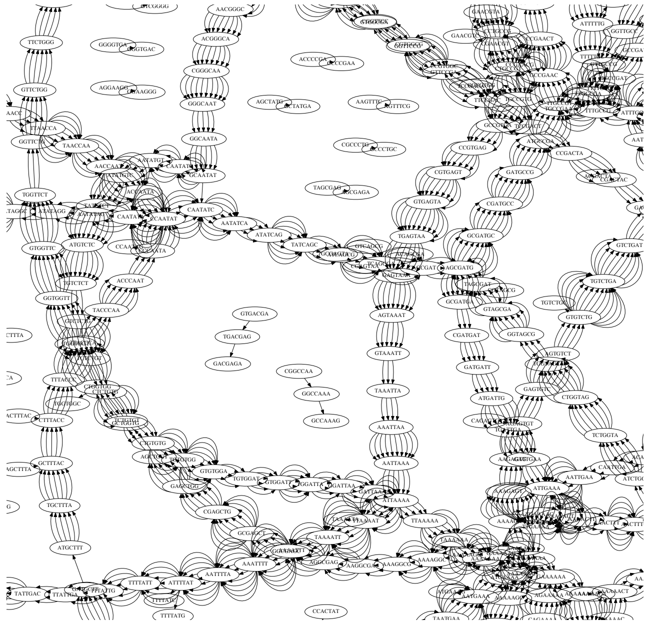


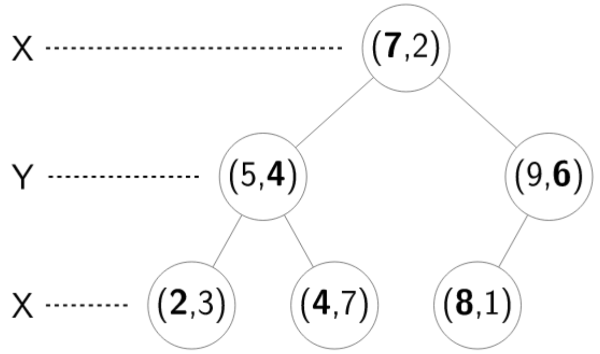
Why should you care?

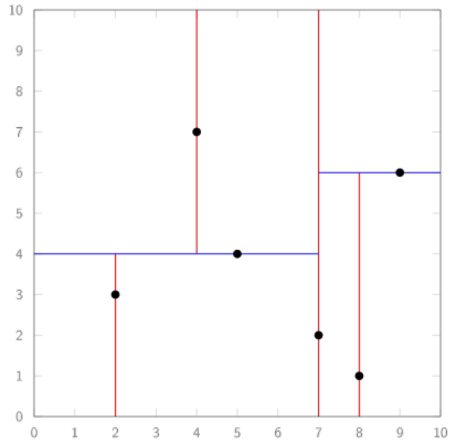
Understand foundational data structures and algorithms



```
Query: 161 atatcaccacgtcaaaggtgactccaactcca---ccactccattttgttcagataatgc 217
             |||||||||||||||||||||||||||||  |     | |   || ||||||||||||||
Sbjct: 481 atatcaccacgtcaaaggtgactccaact-tattgatagtgttttatgttcagataatgc 539
```







What will you get out of this class?

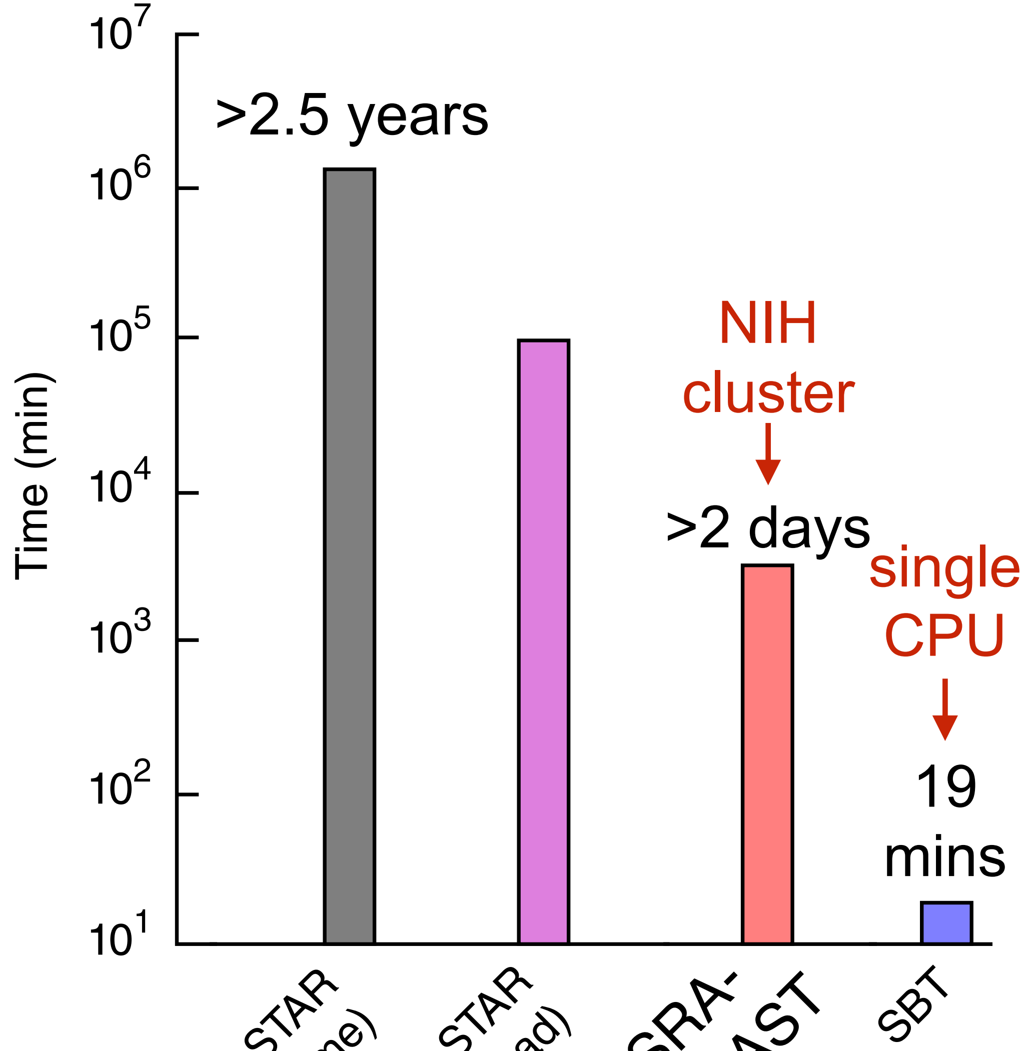
Justify appropriate algorithms for data science problems

Decompose problem into supporting data structures

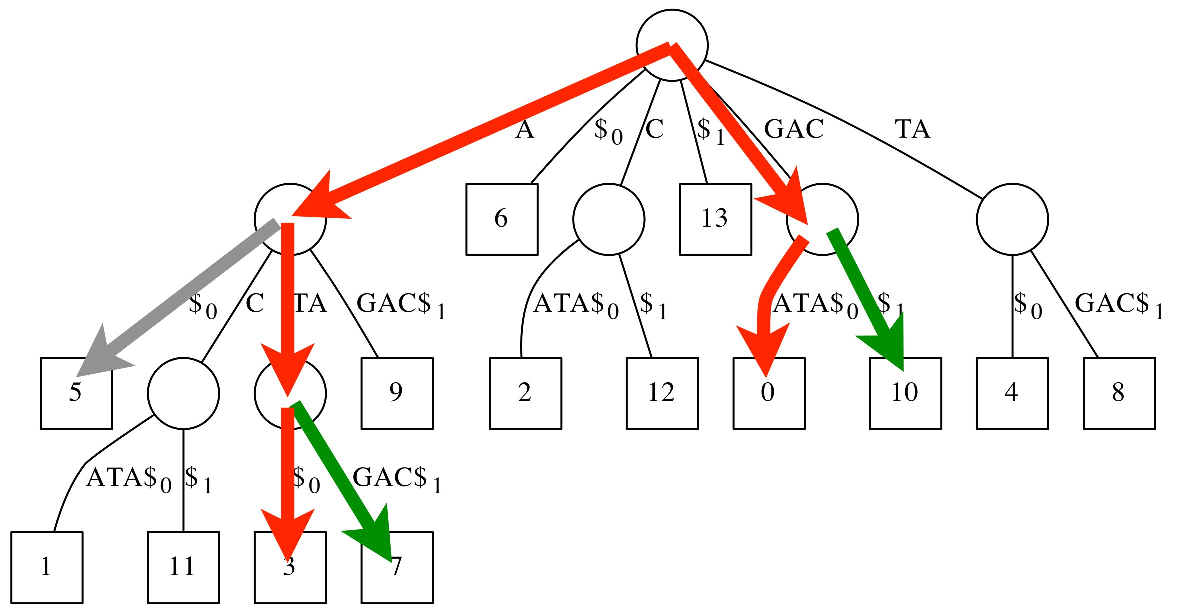
Analyze efficiency of implementation choices

Why should you care?

Justify appropriate algorithms for data science problems



Find all overlap: GACATA vs ATAGAC



CS 225 vs CS 277

If you've taken CS 128, you should probably be taking CS 225

Course Webpage

https://courses.grainger.illinois.edu/cs277/sp2023/

All course information and links can be found here!

Course Schedule and Lecture Material

Assignment links and descriptions

Piazza links and Office Hours

Syllabus

In-Lecture Course Expectations

Attendance is encouraged but not mandatory

Ask questions!

Participate in class exercises / labs

Out-of-lecture Course Expectations

Weekly assessments

Lab assignments are published **Friday** and due **Monday @ 11:59 PM**

Mini-projects deadlines are published with each project (~3 weeks)

Watch recorded lectures (if you missed in-person)

All lectures are published on Mediaspace:

https://mediaspace.illinois.edu/channel/CS+277/225216063

Grading

Category	Contribution	Notes
Mini-Projects	300	75 points each
Labs	300	25 points each
Exams	300	100 points each
Final	100	+ 1 retake exam

Points	Grade
900	A-
800	B-
700	C-
600	D-
	F

Mental Health

This class should be low-stress, medium work-load.

UIUC offers a variety of confidential services:

Counseling Center: 217-333-3704

610 East John Street Champaign, IL 61820

McKinley Health Center: 217-333-2700

1109 South Lincoln Avenue, Urbana, Illinois 61801

Diversity, Equity, and Inclusion

"If you witness or experience racism, discrimination, micro-aggressions, or other offensive behavior, you are encouraged to bring this to the attention of…"

Course CAs

Faculty

Campus Belonging Office (Link)

The Office of Student Conflict Resolution (Link)

CS CARES (Link)

Class structure is under development!

First cohort of students with the required pre-requisites

Frequent assessment will allow adjustments as needed

Reviewing the fundamentals

This is not an intro class but the pre-requisites are not programming classes

We will spend the next week and a half (roughly) on review

Hopefully at least some of it will be new or interesting!

Fundamentals of Programming

A program is a set of sequential instructions

```
a="3"
b=3
c=3.0
d=True

print(a + b)

print("3 + 3”)

print(b + c)

print(c + d)

print(d)

print(d - d)


```

What information is necessary to define a variable?

Data Types

Python has many built-in data types:

https://www.w3schools.com/python/python_datatypes.asp

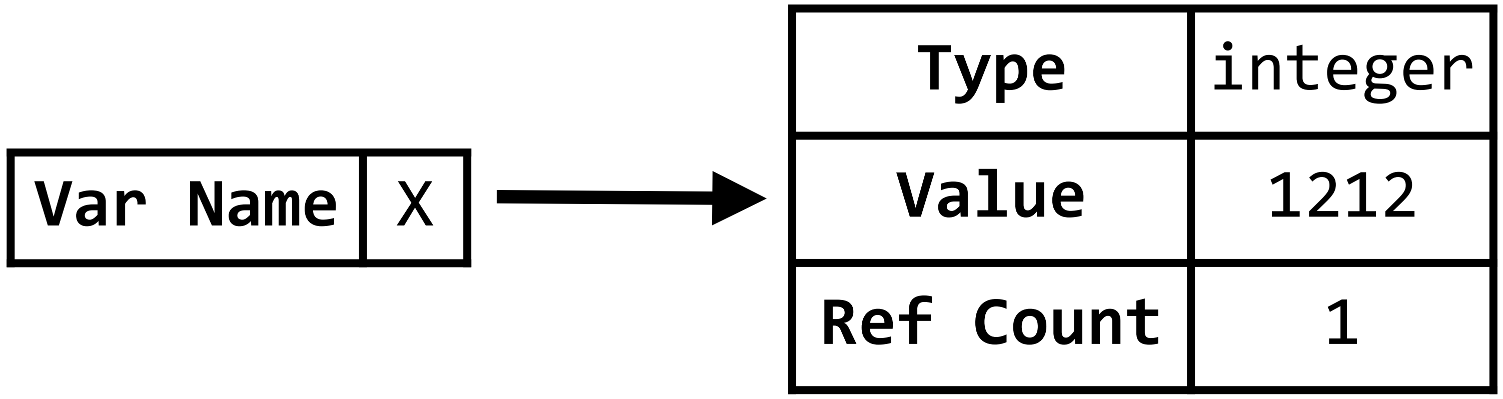
If you want to define (or convert) a variable's data type:

```
a="3"
b=3
c=3.0
d=True

print(a + b)
```

Python Variables

Everything in Python is an **object**

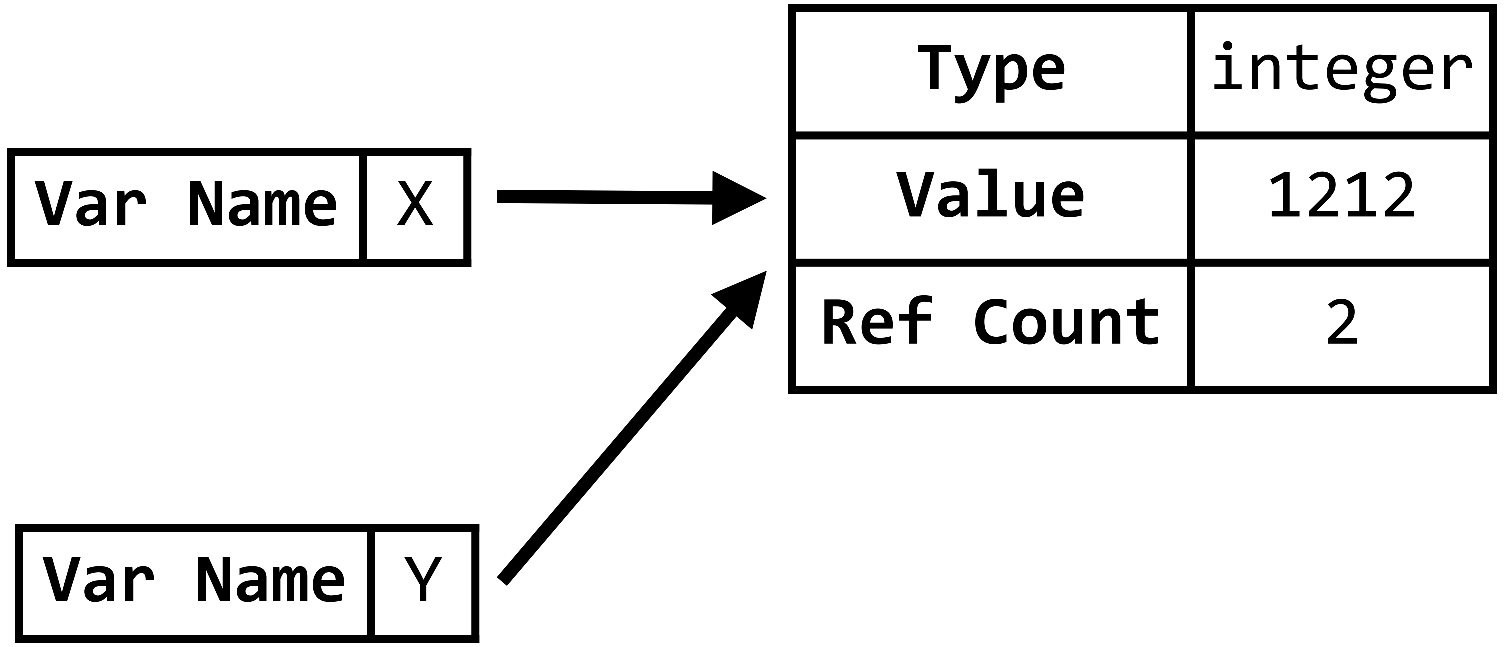
```
X = 1212
```



Python Variables

Everything in Python is an **object**

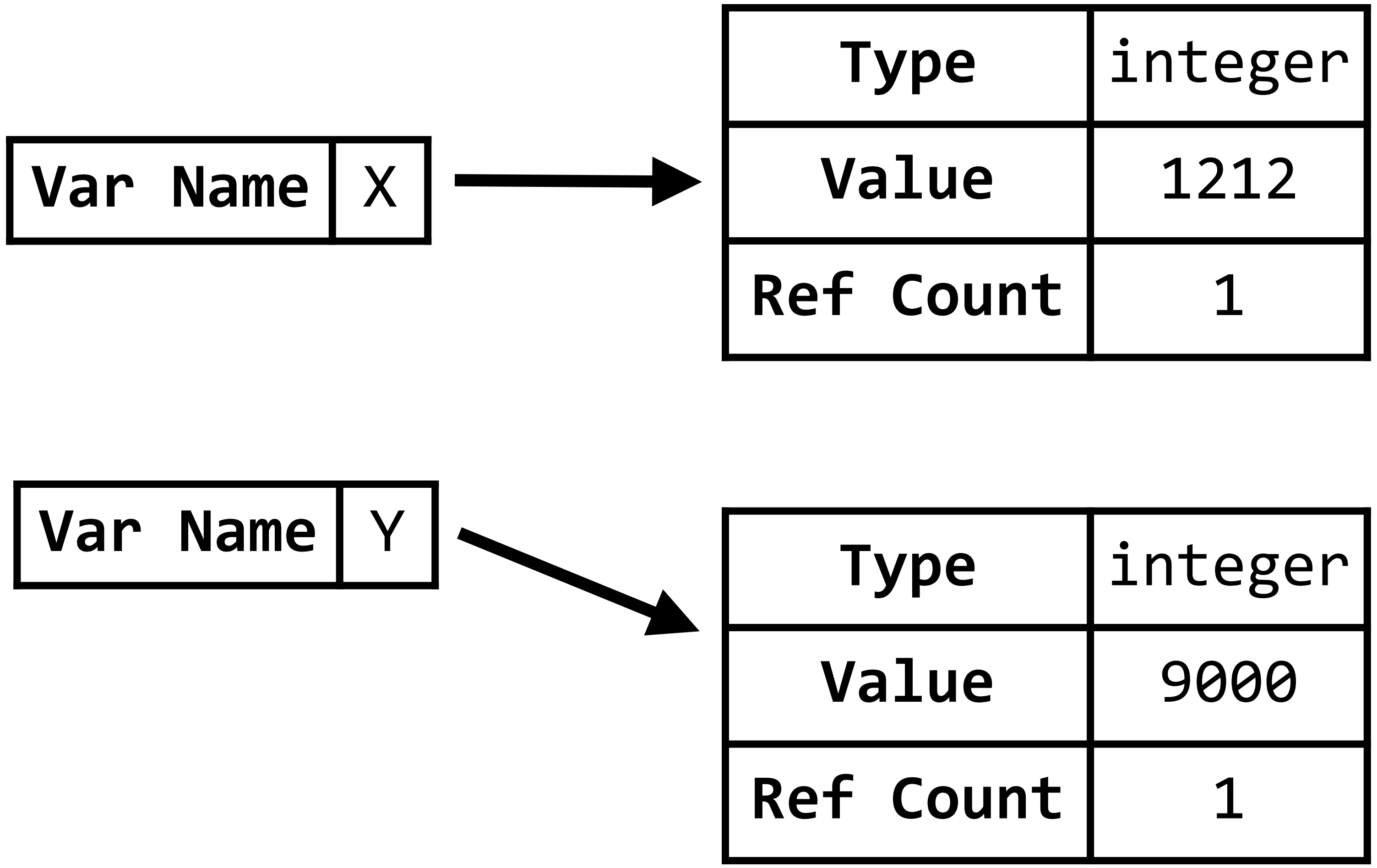
```
X = 1212

Y = X
```



Python Variables

Everything in Python is an **object**

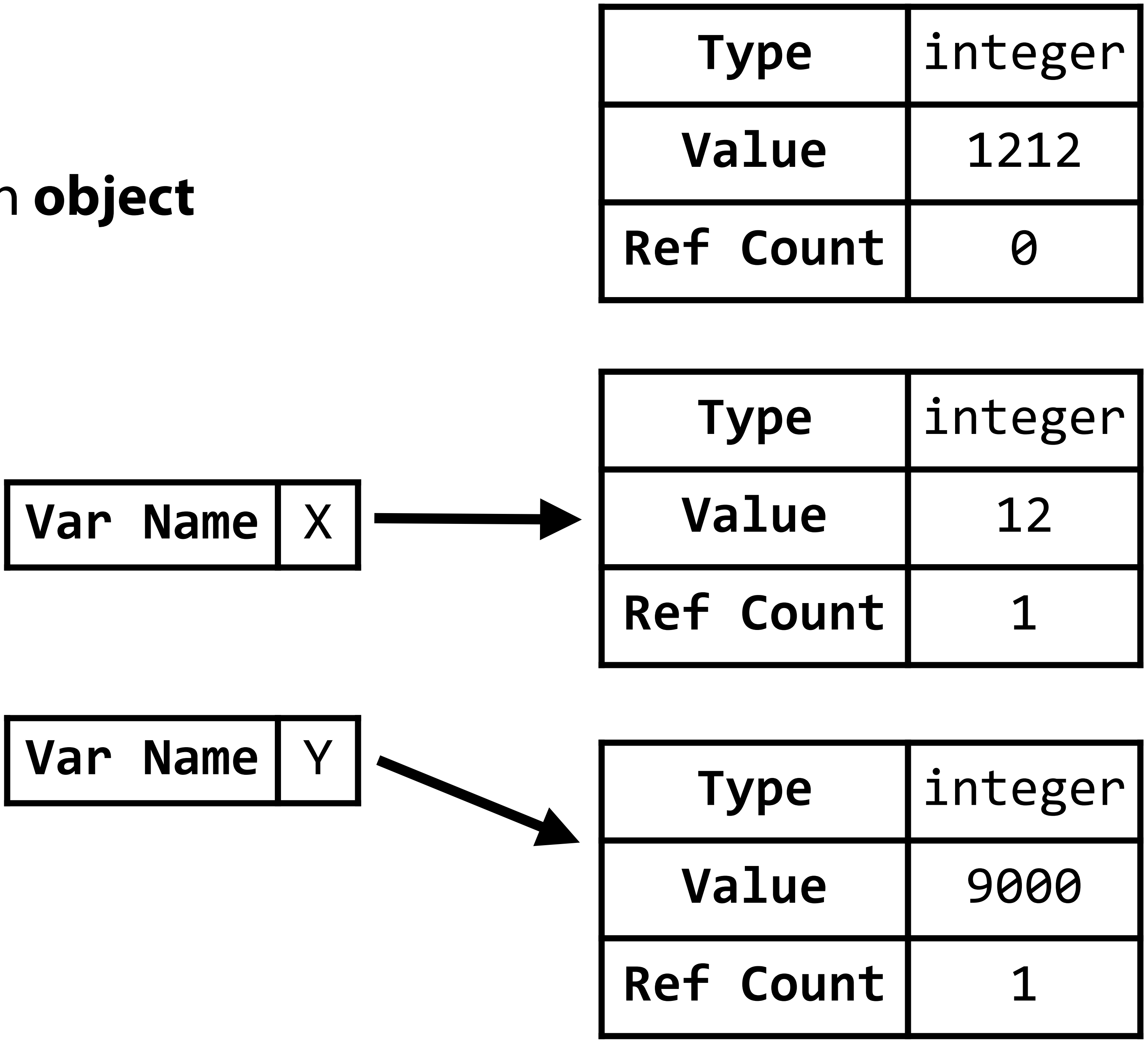
```
X = 1212

Y = X

Y = 9000
```



Python Variables

Everything in Python is an **object**

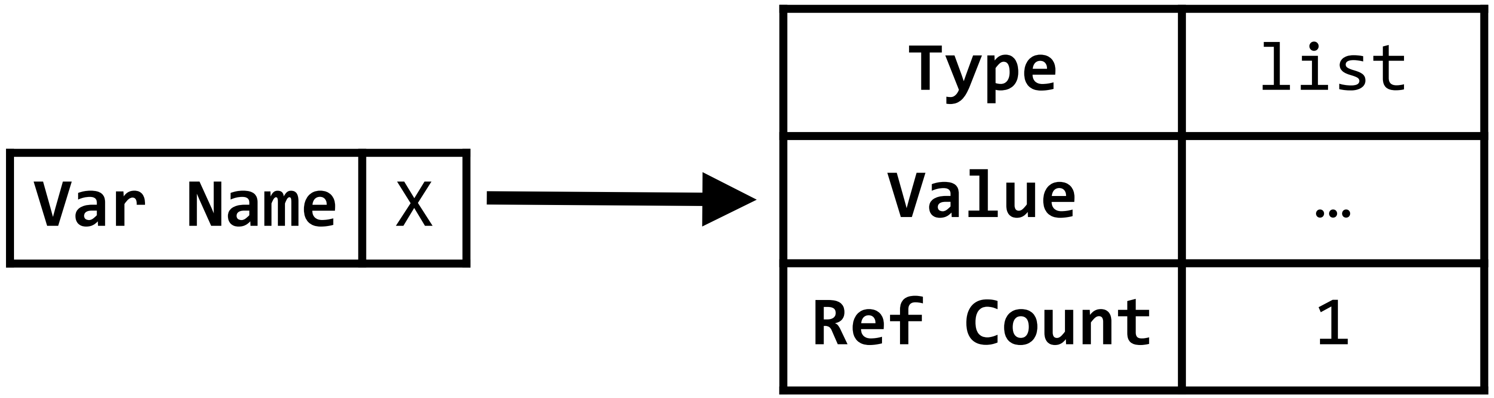
```
X = 1212

Y = X

Y = 9000

X = 12
```



Python Variables

Some objects are **mutable** — we can change their values after creation

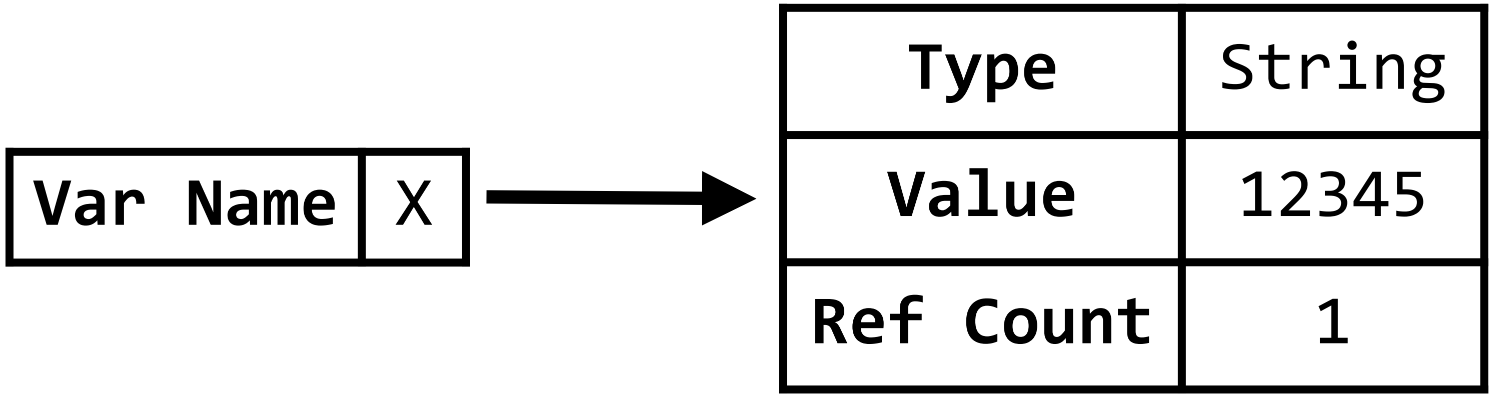
```
X = [1,2,3,4,5]

print(id(x))
X[2]=0
print(id(x))

print(x)
```



Python Variables

Some objects are **immutable** — you have to make a new object

```
X = "12345"

print(id(x))

X[2]=0


print(id(x))

print(x)
```



Type	String
Value	12345
Ref Count	1

Why do we care?

Most of the time we don't! *(A lot of this is just trivia)*

But if you are working with **Big Data** or care about efficiency, this sometimes matters

```
def type1(strList):
    out = ''
    for s in strList:
        out += s
    return out

def type2(strList):
    return ''.join(strList)
```

Logic Expressions in Python

Conditional statements control what blocks of code get run

```
num = 20

if num in [0,1,2,3,4]:
    print("Top 5!”)

elif num > 10:
    print("num too large!”)

elif num > 15:
    print("will this ever get called?”)

else:
    print(num)
```

Loops in Python

There are two kinds of loops in Python

```
for i in range(3):
    print(i)





count = 0
while(count <= 2):
    print(count)
    count+=1
```

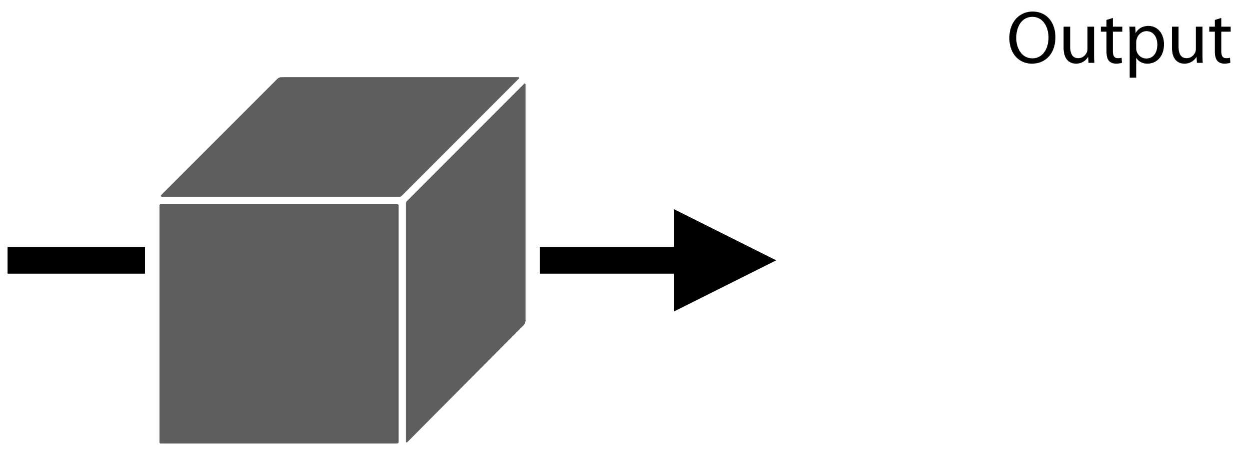
Python Looping Keywords

There are a number of useful keywords for writing loops

```
count = 0

while(True):
    if count % 2 == 0:
        count+=1
    else:
        pass

    if count > 10:
        break
    else:
        count+=1
        continue
        count+=1
    print('count: {}'.format(count))

print('count: {}'.format(count))

```

Functions

Functions are the building blocks of programming



```python
def type1(strList):
    out = ''
    for s in strList:
        out += s
    return out

def type2(strList):
    return ''.join(strList)
```

```python
def mystery(inValue):
    return inValue + inValue
```

Functions

Always document the intended input and output.

```
# INPUT:
# A string (checkin)
# A string (checkout)
# OUTPUT:
# A float storing the number of minutes between checkin and checkout time.
def getTotalTime(checkin, checkout):










def mystery(inValue):
    return inValue + inValue
```

Functions

Immutable variables created in a function have local scope

Mutable variables created in a function can be modified

```
def scopeTest(inNum, inString, inList):
    inNum = 3

    inString+="And After!"



    inList.pop(-1)
    inList.append(5)

x = 2
y = "Before! "
z = [1,2,3,4]

scopeTest(x,y,z)

print(x)
print(y)
print(z)
```

Functions

Functions are **objects** (like everything in Python)

```
# INPUT:
# Three integers (a, b, c)
# An optional function (f)
# OUTPUT:
# If f exists, return output of f(a, b, c). Else return defaultF(a,b,c)
def wrapperFunction(a, b, c, f=None):
    if f == None:
        return defaultF(a,b,c)
    else:
        return f(a,b,c)


if __name__ == '__main__':
    wrapperFunction(5,3,2, add)


    wrapperFunction(1,1,1, multiply)

```

Putting it all together

♦ ♥ ♠ ♣

Let’s program the output of a slot machine!

$500 = Four matching symbols

$100 = Four of a color

$50 = Three matching symbols in a row

$10 = One of each symbol

First Lab Friday!

Bring your laptop (first part will be going over installation instructions)