Objectives

- Wrapping up 2D Lists
- Comparing Programming Languages

1

Review: Extensions to S

```
def search(searchlist, key):
    low=0
    high = len(searchlist)-1
    while low <= high :
        mid = (low+high)//2
        if searchlist[mid] == key:
            return mid
        elif key > searchlist[mid]:
            # look in upper half
            low = mid+1
        else:
            # look in lower half
            high = mid-1
    return -1
```

Goal: find a Person with a certain name
Consider what happens when **searchlist** is a list of *Persons*, ***key*** is a ***str*** representing the *name*

Good capstone problem:
Brings together
- Algorithms
- Classes/Objects
- Lists
- Methods
- While loops
- Strings

0	1	2	3	4
Person Id: "4" "Ben"	Person Id:"3" "Brie"	Person Id: "1" "Gal"	Person Id:"2" "Henry"	Person Id:"5" "Samuel"

2

Review: Summary of Search

- Add a search method
  - Takes as parameter the name to search for
    - Need to lowercase that name for more intuitive results
  - Original binary search function took a list as a parameter; our method does not
    - Where should we get our list to search?
  - The list to search must be sorted in alphabetical order
- Check the *name* of the Person that is at the midpoint, lowercased
  - If they match, return that Person
  - Otherwise, …
- Represent (in method) and handle (in UI) when no person has that name

Incorrect: Creating a 2D List

- The following code **won't** work. Why?
- Example output from using this function to create a 2D list is on the right

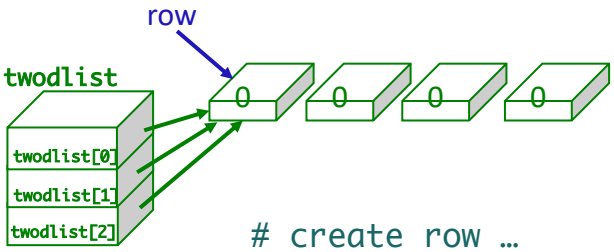
```
def noCreate2DList(rows, cols):
    twodlist = [ ]
    row = [ ]

    for col in range( cols ):
        row.append(0)

    for r in range( rows ):
        twodlist.append(row)
    return twodlist
```

```
Incorrect Matrix Creation:
----------------------------
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]

Assigning matrix[1][2] = 3

Result:
[[0, 0, 3, 0], [0, 0, 3, 0], [0, 0, 3, 0]]
```

twod_exercises.py

All Rows of 2D List Point at Same Block of Memory

- Each "row" points to the **same** list in memory

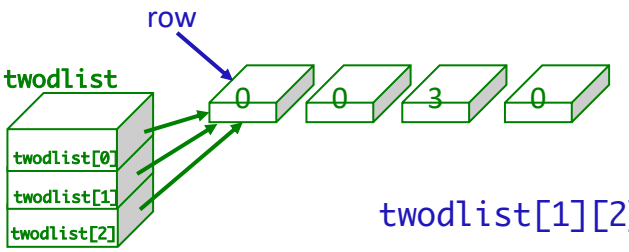



```
# create row …
twodlist.append(row)
twodlist.append(row)
twodlist.append(row)
```


5

All Rows of 2D List Point at Same Block of Memory

- Each "row" points to the **same** list in memory




```
twodlist[1][2] = 3
```


6

Applying What You Know To Other Languages

- At the beginning of the semester, some of you wondered
  - "Why the Python programming language?"
  - "Will I be able to read/write programs in other programming languages?"
- We'll answer the first question by showing that you can do the second

7

Applying What You Know To Other Languages

- **Syntax**: symbols used
- **Semantics**: what the symbols *mean*

8

What is the Python3 Program Doing?

What is the Python3 Program Doing?

- Getting a line of input from "standard in" (from the user)
- Splitting the input into integers
- Calculating the result of a formula
- Deciding if a student is admitted, based on the result of the formula
- Displaying the result

Admissions Problem

- Binary University decides to admit students based on a formula that weighs various factors
  - Scores of 70 or better are admitted
- Input: single line, 4 integers, in order below

Category	Range	Weight Factor (Multiplier)
AP Courses	0-10	10
Intangibles	1-10	8
High School GPA	0 - 10	0.25
SAT score	600-2400	.01

11

Example Input/Expected Output

Input	Expected Output
0 1 0 300	DENY
6 10 99 2390	ADMIT
0 7 82 1500	ADMIT
2 5 0 990	DENY
2 5 0 1000	ADMIT
2 5 0 1010	ADMIT

12

What is the Python Program Doing?

- Getting a line of input from "standard in" (from the user)
- Splitting the input into integers
- Calculating the result of a formula
- Deciding if a student is admitted, based on the result of the formula
- Displaying the result

Identify these pieces in the other programs

Comparing Programming Languages

- How is the syntax/semantics of these languages different from Python?
- What is easier or harder to do in these other programming languages than in Python?

Comparing Programming Languages

- Benefits of Python:
  - Simpler syntax (e.g., fewer {} and ())
  - Can cover some content with less overhead
- Drawbacks
  - Data types aren't explicit (static)
    - Can be harder for you to remember and keep straight
  - Not compiled explicitly beforehand
    - Keep executing to find all the syntax bugs
    - Doesn't check: "you're passing a file instead of a string"
  - Allows you to do some things that won't work in other programming languages

Bash

- Scripting language
  - Can call Unix commands
- Example program:
  - `createPrintableLab`

Tiobe Index

based on the number of skilled engineers world-wide, courses and third party vendors

Apr 2021	Apr 2020	Change	Programming Language		Ratings	Change
1	2	↑	C	CSCI210, 320	14.32%	-2.40%
2	1	↓	Java	CSCI209, 335	11.23%	-5.49%
3	3		Python	CSCI111, 112	11.03%	+1.72%
4	4		C++		7.14%	+0.36%
5	5		C#		4.91%	+0.16%
6	6		Visual Basic		4.55%	-0.18%
7	7		JavaScript	CSCI335	2.44%	+0.06%
8	14	⇈	Assembly language		2.32%	+1.16%
9	8	↓	PHP		1.84%	-0.54%
10	9	↓	SQL	CSCI335, 317	1.83%	-0.34%

http://www.tiobe.com/tiobe_index

Course Evaluations

- On Canvas, due Monday at 11:59 p.m.
- Incentive
  - If 60% of students complete evaluation, 1% Extra Credit on *lab* grades
  - For each additional 10% of students who complete evaluation, additional 1% EC on lab grades
  - Total possible EC: 5%

Final Exam Take Home Questions

- 2 essay questions about the Broader Issues
- **Due before end of exam period -** Noon Friday
- Each essay should be about ¾ of a page, single-spaced
- Goal: answer the question clearly, precisely, and convincingly
  - Not too wordy
  - Evidence/examples to support your argument
  - Correct spelling, grammar, punctuation

19

Final Exam

- Final will be in Canvas due end of exam period - Friday at noon
- Prep document on schedule
  - Similar format to previous exams but in Canvas
  - More on Friday

20

Looking Ahead

- Friday:
  - Lab 11 due
  - BI write up due
  - Review computer science
    - Where we've been and where you can go
  - Bring your exam questions
    - Practice
- All lab work and extra credit articles must be submitted by **MONDAY 11:59 p.m.**