

## Objectives

- Wrap up search
- Two-dimensional lists

Apr 2, 2012

Sprenkle - CSCI111

1

## Computer Science Understanding

- Do you understand what a class is and its purpose? What is a class made up of?
- Can you implement a class (of "reasonable" size), given what it is supposed to represent and what it is supposed to do?
- Given a class's API, can you solve problems with it?
  - When you write the UI for FaceSpace, you are using the API for the `SocialNetwork` class
- Do you understand the strengths and weaknesses of linear and binary search? When would you use one over the other?

Apr 2, 2012

Sprenkle - CSCI111

2

## Review

- How can we sort a list by some criteria that isn't the "natural" way to search?

Apr 2, 2012

Sprenkle - CSCI111

3

## List of Person objects

0	1	2	3	4
Person Id: "1" "George" "Hollywood"	Person Id: "2" "Jennifer" "Friends"	Person Id: "3" "Matt D" "Boston"	Person Id: "4" "Ben A." "Boston"	Person Id: "5" "Lucy L" "Southland"

0	1	2	3	4
Person Id: "4" "Ben A." "Boston"	Person Id: "3" "Matt D" "Boston"	Person Id: "2" "Jennifer" "Friends"	Person Id: "1" "George" "Hollywood"	Person Id: "5" "Lucy L" "Southland"

Sorted by network using:

```
personList.sort(key=Person.getNetwork)
```

Apr 2, 2012

Sprenkle - CSCI111

4

## Extensions to Search

In FaceSpace, we want to find people who are in a certain network.

Consider what happens when `searchList` is a list of *Persons* and `key` is a network name

- What if we wanted to find a *Person* whose network matched the key and return the *Person*?

Apr 2, 2012

Sprenkle - CSCI111

5

## List of Person objects

0	1	2	3	4
Person Id: "1" "George" "Hollywood"	Person Id: "2" "Jennifer" "Friends"	Person Id: "3" "Matt D" "Boston"	Person Id: "4" "Ben A." "Boston"	Person Id: "5" "Lucy L" "Southland"

Example: looking for a person in the "Boston" network...

Apr 2, 2012

Sprenkle - CSCI111

6

## Extensions to Solution

```
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
```

Consider what happens when `searchlist` is a list of *Persons*

- What if we wanted *all* the *Persons* with the network that matched the key?
- Assumes many different networks

Apr 2, 2012

Sprenkle - CSCI111

7

## 2D LISTS

Apr 2, 2012

Sprenkle - CSCI111

8

## Lists

- We've used lists that contain
  - Integers
  - Strings
  - Cards (Deck class)
  - Persons (your Person class)
- We discussed that lists can contain multiple types of objects within the same list
  - Wheel of Fortune: ["Bankrupt", 250, 350, ...]
- Lists can contain *any type* of object
  - Even **LISTS!**

Apr 2, 2012

Sprenkle - CSCI111

9

## Review of Regular (1D) Lists

- Create a list `onedlist = [ 7, -1, 23 ]`

Elements in the list

- How do we find the number of elements in the list?
- How can we find the value of the third element in the list?

Apr 2, 2012

Sprenkle - CSCI111

10

## Review of Regular (1D) Lists

- Create a list `onedlist = [ 7, -1, 23 ]`
- `len(onedlist)` is 3
- `onedlist[2]` is 23

Elements in the list

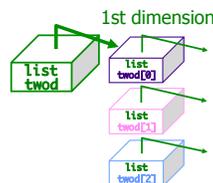
Apr 2, 2012

Sprenkle - CSCI111

11

## A List of Lists: 2-dimensional List

```
twod = [ twod[0], twod[1], twod[2] ]
twod = [ [1,2,3,4], [5,6], [7,8,9,10,11] ]
```



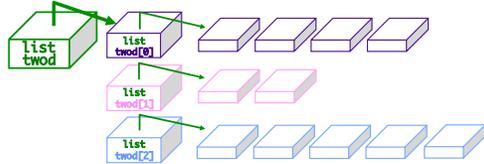
Apr 2, 2012

Sprenkle - CSCI111

12

## A List of Lists: 2-dimensional lists

```
twod = [ [1,2,3,4], [5,6], [7,8,9,10,11] ]
```



- “Rows” within 2-dimensional list do **not** need to be same length
- However, it’s often easier if they’re the same length!  
 > We’ll focus on “rectangular” 2-d lists

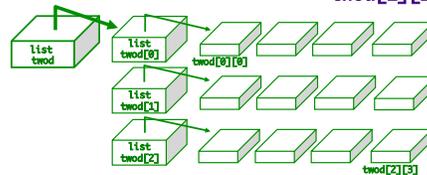
Apr 2, 2012

Sprenkle - CSCI111

13

## Handling Rectangular Lists

```
twod[1][2] = 42
```



- What does each component of `twod[1][2]` mean?
- How many rows does `twod` have, in general?
- How many columns does `twod` have, in general?

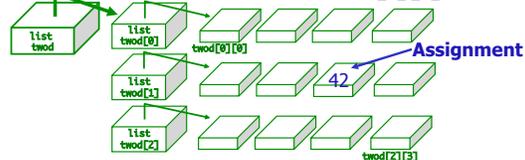
Apr 2, 2012

Sprenkle - CSCI111

14

## Handling Rectangular Lists

Row pos  
Col pos  
2-d list var  
`twod[1][2] = 42`



- What does each component of `twod[1][2]` mean?
- How many rows does `twod` have, in general?  
 > `rows = len(twod)`
- How many columns does `twod` have, in general?  
 > `cols = len(twod[0])`

Apr 2, 2012

Sprenkle - CSCI111

15

## Practice

Starting with the 2d list `twod` shown here, what are the values in `twod` after running this code?

**twod Before**

row 0	1	2	3	4
row 1	5	6	7	8
row 2	9	10	11	12
	col 0	col 1	col 2	col 3

```
def mystery(twod):
    """ 'run' this on twod, at right """
    for row in range( len(twod) ):
        for col in range( len(twod[0]) ):
            if row == col:
                twod[row][col] = 42
            else:
                twod[row][col] += 1
```

**twod After**


Apr 2, 2012

Sprenkle - CSCI111

mystery.py

16

## Practice

Starting with the 2d list `twod` shown here, what are the values in `twod` after running this code?

**twod Before**

row 0	1	2	3	4
row 1	5	6	7	8
row 2	9	10	11	12
	col 0	col 1	col 2	col 3

```
def mystery(twod):
    """ 'run' this on twod, at right """
    for row in range( len(twod) ):
        for col in range( len(twod[0]) ):
            if row == col:
                twod[row][col] = 42
            else:
                twod[row][col] += 1
```

**twod After**

42	3	4	5
6	42	8	9
10	11	42	13

Apr 2, 2012

Sprenkle - CSCI111

mystery.py

17

## Creating a 2d List

- ```
twod = [ ]
```
- Need to create a row of the list  
`row = [1, 2, 3, 4]`
  - Then append that row to the list  
`twod.append( row )`  
`print(twod)`
    - `[[1, 2, 3, 4]]`
  - Repeat  
`row = [1, 2, 3, 4]`  
`twod.append( row )`  
`print(twod)`
    - `[[1, 2, 3, 4], [1, 2, 3, 4]]`

Apr 2, 2012

Sprenkle - CSCI111

18

## Generalize Creating a 2D List

- Create a function that returns a 2D list with width **cols** and height **rows**
  - Initialize each element in list to 0

Apr 2, 2012

Sprenkle - CSC1111

19

## Generalize Creating a 2D List

- Create a function that returns a 2D list with width **cols** and height **rows**
  - Initialize each element in list to 0

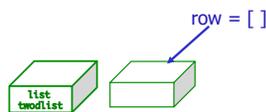
```
def create2DList(rows, cols):  
    twodlist = [ ]  
    # for each row  
    for row in range( rows ):  
        row = [ ]  
        # for each column, in each row  
        for col in range( cols ):  
            row.append(0)  
        twodlist.append(row)  
    return twodlist
```

Apr 2, 2012

Sprenkle - CSC1111

20

## How Does This Work?

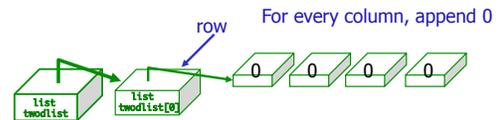


Apr 2, 2012

Sprenkle - CSC1111

21

## How Does This Work?



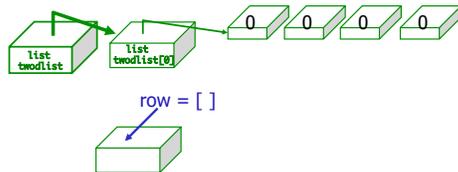
Append row to twodlist

Apr 2, 2012

Sprenkle - CSC1111

22

## How Does This Work?

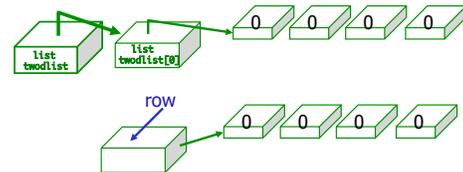


Apr 2, 2012

Sprenkle - CSC1111

23

## How Does This Work?

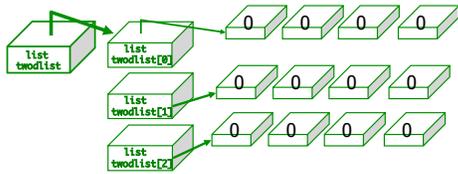


Apr 2, 2012

Sprenkle - CSC1111

24

## How Does This Work?



Apr 2, 2012

Sprenkle - CSC1111

25

## Incorrect: Creating a 2D List

- The following code **won't** work. Why?
- Explain output from example program

```
def noCreate2DList(rows, cols):  
    twodlist = [ ]  
    row = [ ]  
    # create a row with appropriate columns  
    for col in range( cols ):  
        row.append(0)  
    # append the row rows times  
    for row in range( rows ):  
        twodlist.append(row)  
    return twodlist
```

twod\_exercises.py

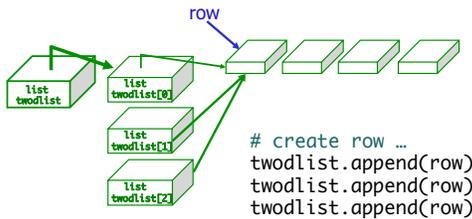
Apr 2, 2012

Sprenkle - CSC1111

26

## All Rows Pointing at Same Block of Memory

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



Apr 2, 2012

Sprenkle - CSC1111

27

## Problem: Create a Tic-Tac-Toe board

- Returns a 2-d list that represents a tic-tac-toe board
  - What elements should be in the 2D list?

Apr 2, 2012

Sprenkle - CSC1111

28

## Problem: Tic-Tac-Toe

- How do we represent player's moves?
  - How do we update the board to say "Player X goes into the bottom right corner."

Apr 2, 2012

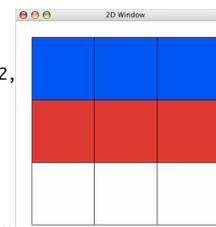
Sprenkle - CSC1111

29

## Graphical Representation of 2D Lists

- Module: `csp1ot`
- Allows you to visualize your 2D list
  - Numbers are represented by different colors

```
import csp1ot  
...  
# create 2D list...  
twodlist=[ [0,0,0], [1,1,1], [2,2,  
# display list graphically  
csp1ot.show(twodlist)
```



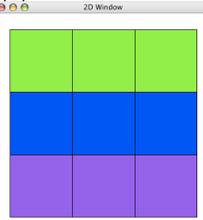
Apr 2, 2012

Sprenkle - CSC1111

## Graphical Representation of 2D Lists

- Can assign colors to numbers

```
import csploit
...
# create 2D list...
twodlist=[ [0,0,0], [1,1,1], [2,2,2] ]
# create optional dictionary of num
numToColor={0:"purple", 1:"blue", 2
csploit.show(twodlist, numToColor)
```

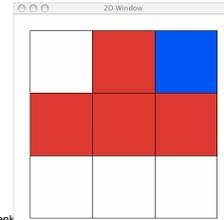


Apr 2, 2012 Sprenkle - CSCI1111

## Graphical Representation of 2D Lists

matrix = [[0,0,0], [1,1,1], [0,1,2]]

What values map to which colors?

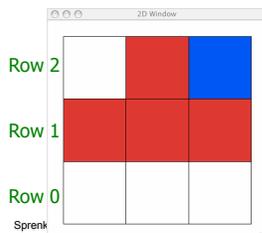


## Graphical Representation of 2D Lists

- Note that representation of rows is backwards from how we've been visualizing

matrix = [[0,0,0], [1,1,1], [0,1,2]]

What values map to which colors?



## Game Board for Connect Four

- 6 rows, 7 columns board
- Players alternate dropping red/black checker into slot/column
- Player wins when have four checkers in a row vertically, horizontally, or diagonally

How do we represent the board as a 2D list, using a graphical representation?

Apr 2, 2012

Sprenkle - CSCI1111

34

## Game Board for Connect Four

- How to represent board in 2D list, using graphical representation?

| Number | Meaning  | Color  |
|--------|----------|--------|
| 0      | Free     | Yellow |
| 1      | Player 1 | Red    |
| 2      | Player 2 | Black  |

Apr 2, 2012

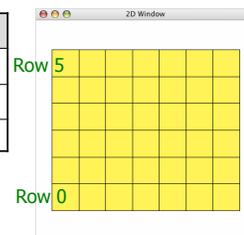
Sprenkle - CSCI1111

35

## Game Board for Connect Four

- How to represent board in 2D list, using graphical representation?

| Number | Meaning  | Color  |
|--------|----------|--------|
| 0      | Free     | Yellow |
| 1      | Player 1 | Red    |
| 2      | Player 2 | Black  |



## Connect Four (C4): Making moves

- User clicks on a column
  - “Checker” is filled in at that column

```
# gets the column of where user clicked  
col = csplot.sqinput()
```

Apr 2, 2012

Sprenkle - CSC1111

37

## Problem: C4 - Valid move?

- Need to enforce valid moves
  - In physical game, run out of spaces for checkers if not a valid move
- How can we determine if a move is valid?
  - How do we know when a move is *not* valid?

Apr 2, 2012

Sprenkle - CSC1111

38

## Problem: C4 - Valid move?

- Solution: check the “top” spot
  - If the spot is FREE, then it’s a valid move

Apr 2, 2012

Sprenkle - CSC1111

39

## ConnectFour Class

- Data
  - Board
- Methods
  - Constructor
  - Display the board
  - Play the game
    - Repeat:
      - Get input/move from user
      - Check if valid move
      - Make move
      - Display board
      - Check if win
      - Change player

Apr 2, 2012

Sprenkle - CSC1111

40

## Problem: C4 - Making a Move

- The player clicks on a column, meaning that’s where the player wants to put a checker
- How do we update the board?

Apr 2, 2012

Sprenkle - CSC1111

41

## Plan for This Week

- Tomorrow: Lab 11
  - SocialNetwork - binary search
  - 2D list practice
- Wednesday:
  - Security vulnerabilities
  - Course evaluations: completed by Sunday at midnight
- Friday
  - Programs in other programming languages
  - Broader Issue – One Laptop Per Child

If 70% of class responds,  
1% of possible lab points added (-12 pts)  
For each additional 10%,  
additional 1% off.  
Max ~58 pts

Apr 2, 2012

Sprenkle - CSC1111

42

### Problem: Print a Tic-Tac-Toe Board

- Print the board in a “nice” way, such as

```
x | |  
- - -  
| o |  
- - -  
| | |
```

Apr 2, 2012

Sprenkle - CSCI1111

tictactoe.py

43

### Problem: Determine Win in Tic-Tac-Toe

- Determine when a player wins in tic-tac-toe

Apr 2, 2012

Sprenkle - CSCI1111

44

### Problem: Determine Win in Tic-Tac-Toe

- Determine when a player wins in tic-tac-toe
  - What additional information would make this problem a little easier to manage?
- Note that we are not handling tie games well

Apr 2, 2012

Sprenkle - CSCI1111

tictactoe2.py

45

### Course Grades

- Final Exam: Comprehensive
  - Defining & using classes
  - Searches: Linear, Binary
  - Two-dimensional lists
  - ...
  - See FinalPrep document on line
  - Take-home question about broader issues
- Formula for final grade is on course Web page

Apr 2, 2012

Sprenkle - CSCI1111

46

### Problem: Inspired by W&L Trident

- Draw a letter using a 2D List



Apr 2, 2012

Sprenkle - CSCI1111

47

### Problem: Create a square matrix

- Function takes an argument that is its width
- Returns a matrix of spaces with asterisks (\*) on the diagonal

Apr 2, 2012

Sprenkle - CSCI1111

48

## Problem: ASCII Art

- ASCII Art: print plain-text characters to form pretty pictures
- Create a function that has an input parameter the width of a square 2D list
- Fill the 2D list with asterisks across the diagonal, spaces elsewhere

Apr 2, 2012

Sprenkle - CSCI111

49

## Problem: ASCII Art Frame

- Create a function that has an input parameter the width/height of the 2D list; fill the border with X's, rest with spaces

```
XXXX
X  X
X  X
X  X
XXXX
```

Apr 2, 2012

Sprenkle - CSCI111

50

## Grid with Frame

- Modify code to use a 0 instead of an 'x' for the border

Apr 2, 2012

Sprenkle - CSCI111

51

## Final Exam Details

- Discuss content later
  - [Focus on topics since last exam](#)
- All CS exams are taken in Parmly 405 (our lab)
- At your specified time, someone brings the tests to Parmly 405
- You have 3 hours to take the exam
- Can change exam time by using sheet outside of department office (Parmly 407)

Apr 2, 2012

Sprenkle - CSCI111

52

## Course Evaluations

- Available Wednesday, on Sakai
- General questions about the course
- Specific questions about broader issues

Apr 2, 2012

Sprenkle - CSCI111

53