

Objectives

- Exam next Friday
- More on lists
- Introduction to Dictionaries

Mar 5, 2008

Sprengle - CS111

1

Exam Next Friday

- Cumulative - Up through next Monday's class
- Focus:
 - Strings, functions, modules, files, lists, dictionaries
- Slightly more reading, writing focus
- I will be at a conference next Wed - Sat
 - Ask questions early
 - Wed class: finish up your lab, study for exam

Mar 5, 2008

Sprengle - CS111

2

Lists: A Sequence of Data Elements

element

daysInWeek

"Sun"	"Mon"	"Tue"	"Wed"	"Thu"	"Fri"	"Sat"
0	1	2	3	4	5	6

Position in the list

len(daysInWeek) is 7

- `<listname>[<int_expr>]`
 - Similar to accessing characters in a string
 - `daysInWeek[-1]` is "Sat"
 - `daysInWeek[0]` is "Sun"

Mar 5, 2008

Sprengle - CS111

3

Fibonacci Sequence

- Create a list of the 1st 15 Fibonacci numbers

➢ $F_0 = F_1 = 1; F_n = F_{n-1} + F_{n-2}$

Grows list as we go

```
fibs = []           # create an empty list
fibs.append(1)      # append the first two Fib numbers
fibs.append(1)
for x in xrange(2,15): # compute the next 13 nums
    newfib = fibs[x-1]+fibs[x-2]
    fibs.append(newfib)

print fibs          # print out the list
```

Mar 5, 2008

Sprengle - CS111

fibs.py

4

Fibonacci Sequence

- Create a list of the 1st 15 Fibonacci numbers

➢ $F_0 = F_1 = 1; F_n = F_{n-1} + F_{n-2}$

Create list;
Update values as app.

Similar to xrange,
Call similarly

```
fibs = range(15)    # creates a list of size 15,
                    # containing nums 0 to 14
fibs[0] = 1
fibs[1] = 1
for x in xrange(2,15):
    newfib = fibs[x-1]+fibs[x-2]
    fibs[x] = newfib

for num in fibs:    # print each num on sep line
    print num
```

Mar 5, 2008

Sprengle - CS111

fibs2.py

5

range VS xrange

- **range**: creates a list
 - Use when you want a list
 - for x in range(10000):**
 - Loop goes through each element in the list
 - list has 10,000 integers from 0 to 9,999
- **xrange**: creates an iterator
 - More efficient to use in for loops when you want a counter (not a list)
 - for x in xrange(10000):**
 - Generates 10,000 numbers, one by one

Mar 5, 2008

Sprengle - CS111

6

Lists vs. Arrays

- Briefly, lists are similar to arrays in other languages
 - More similar to *Vectors* in C++ and *ArrayLists* in Java
- Typically, arrays have **static** lengths
 - Can't insert and remove elements from arrays so that the length of the array changes
 - Need to make the array as big as you'll think you'll need

Mar 5, 2008

Sprenkle - CS111

7

Lists vs. Strings

- Strings are **immutable**
 - Can't be mutated?
 - Er, can't be modified/changed
- Lists can be changed
 - Mutable!
 - Changes how we call/use methods

```
groceryList=["milk", "eggs", "bread", "Doritos", "OJ", "sugar"]
```

```
groceryList[0] = "skim milk"  
groceryList[3] = "popcorn"
```

```
groceryList is now ["skim milk", "eggs", "bread",  
"popcorn", "OJ", "sugar"]
```

Mar 5, 2008

Sprenkle - CS111

8

Practice

- list = [1,2,3]
- string = "123"
- list[1]
- string[1]
- string.upper()
- list.reverse()
- string
- list
- string = string.upper()
- list = list.reverse()
- string
- list

Mar 5, 2008

Sprenkle - CS111

9

Practice: Wheel of Fortune

- Allow user to choose between several categories of puzzles
 - Each category is represented by a different file, e.g., oscars.txt, grammy_noms.txt, famous_pairs.txt
- How to model/implement this in Python?
 - How to represent data?
- Where in code?

Mar 5, 2008

Sprenkle - CS111

10

Practice: Wheel of Fortune

- Modify to keep track of previous guesses
 - If user made that guess before, print message
- What are the data types of the data we're modeling?

Mar 5, 2008

Sprenkle - CS111

11

Practice: Wheel of Fortune

- Model the wheel
 - Money
 - Bankruptcy, lose a turn, free spin
- Simulate spinning the wheel

Mar 5, 2008

Sprenkle - CS111

12

