Objectives

- Program Organization
- Introduction to Objects
- Working with text files

Program Organization

- Larger programs require functions to maintain readability
  - Use main() and other functions to break up your program into smaller, more manageable chunks
  - "**Abstract** away" the details
- As before, you can still write smaller scripts without any functions
  - Can try out functions using smaller scripts
- Need the main() function when using other functions to keep "driver" at top
  - Otherwise, functions need to be defined **before** use

Programming Paradigm: Imperative

- Most modern programming languages are imperative
- Have data (numbers and strings in variables)
- Perform operations on data using operations, such as + (addition and concatenation)
- Data and operations are separate

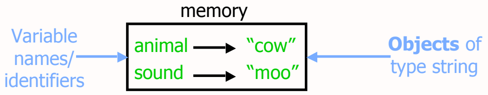
- Add to imperative: object-oriented programming

Object-Oriented Programming

- Program is a collection of **objects**
- Objects **combine** data and methods together
- Objects interact by invoking methods on other objects
  - Methods perform some operation on object

Object-Oriented Programming

- We've been using objects and methods
  - Just didn't call them objects
- For example: **string** is a data type (or **class**)
  - We created objects of type (*class*) string
    - animal = "cow"
    - sound = "moo"



memory

Variable names/ identifiers

animal → "cow"

sound → "moo"

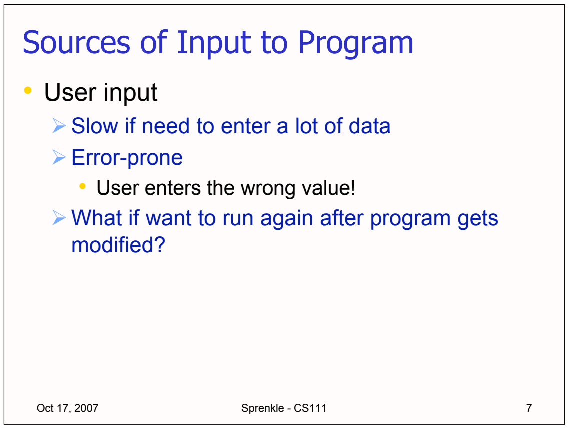
Objects of type string

Object-Oriented Programming

- The string **class** defined methods that you can use on objects of **type** string
  - Example methods: lower, replace, find, …

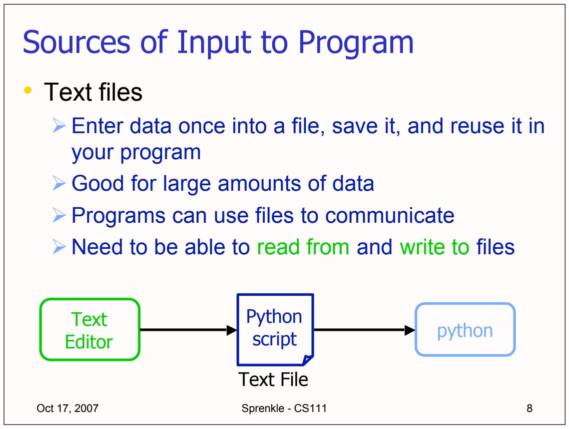
- Methods are similar to functions but *called/used* differently:
  - objectname.methodname([parameters])
  - Examples: animal.upper(), sound.center(10)

- Today: new **class** with its own methods



Sources of Input to Program

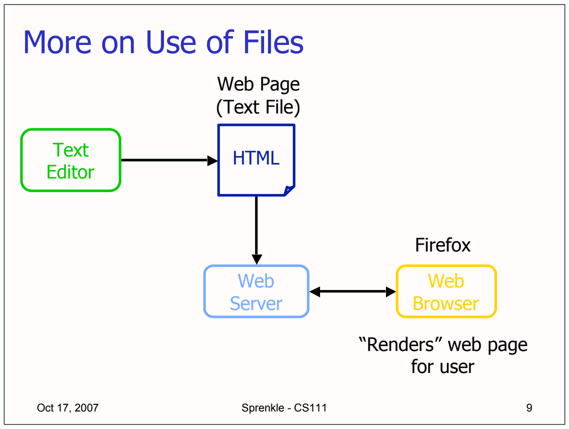
- User input
  - Slow if need to enter a lot of data
  - Error-prone
    - User enters the wrong value!
  - What if want to run again after program gets modified?



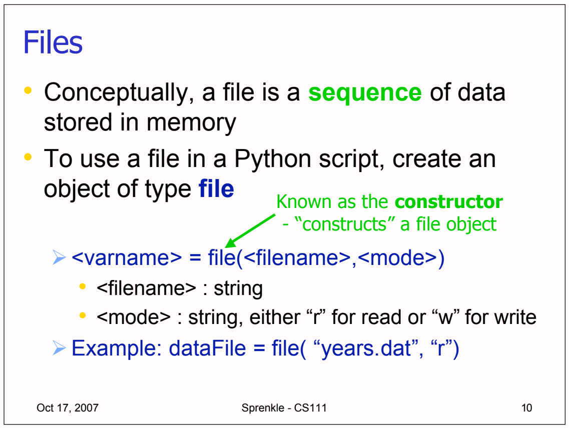
Sources of Input to Program

- Text files
  - Enter data once into a file, save it, and reuse it in your program
  - Good for large amounts of data
  - Programs can use files to communicate
  - Need to be able to read from and write to files

Text Editor
Python script
python
Text File



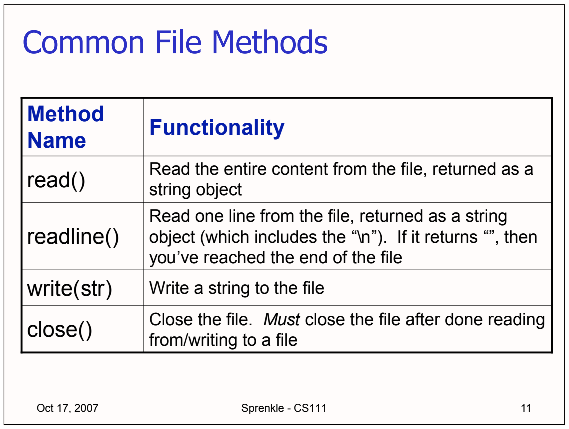
More on Use of Files



Files

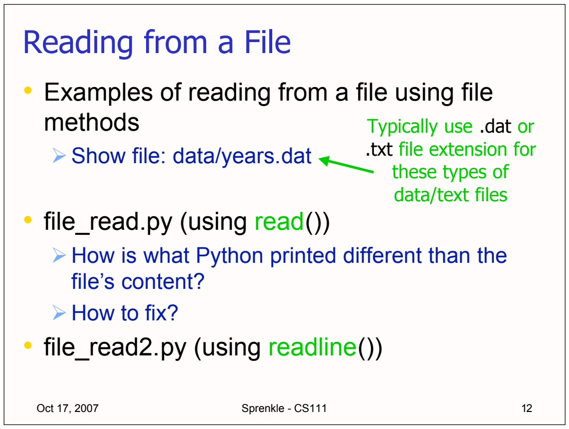
- Conceptually, a file is a **sequence** of data stored in memory
- To use a file in a Python script, create an object of type **file**
  - <varname> = file(<filename>,<mode>)
    - <filename> : string
    - <mode> : string, either "r" for read or "w" for write
  - Example: dataFile = file("years.dat", "r")

Known as the **constructor** - "constructs" a file object



Common File Methods

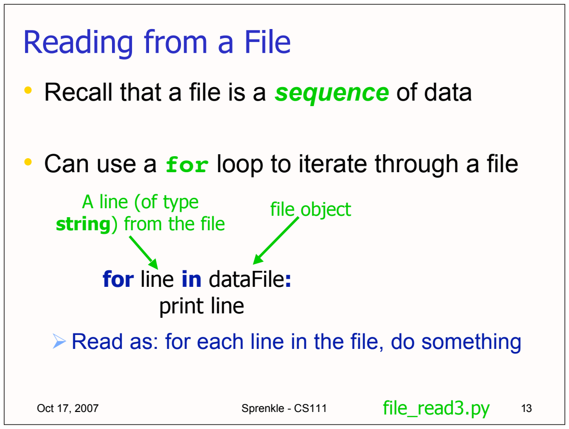
Method Name	**Functionality**
read()	Read the entire content from the file, returned as a string object
readline()	Read one line from the file, returned as a string object (which includes the "\n"). If it returns "", then you've reached the end of the file
write(str)	Write a string to the file
close()	Close the file. *Must* close the file after done reading from/writing to a file



Reading from a File

- Examples of reading from a file using file methods
  - Show file: data/years.dat
- file_read.py (using read())
  - How is what Python printed different than the file's content?
  - How to fix?
- file_read2.py (using readline())

Typically use .dat or .txt file extension for these types of data/text files



Reading from a File

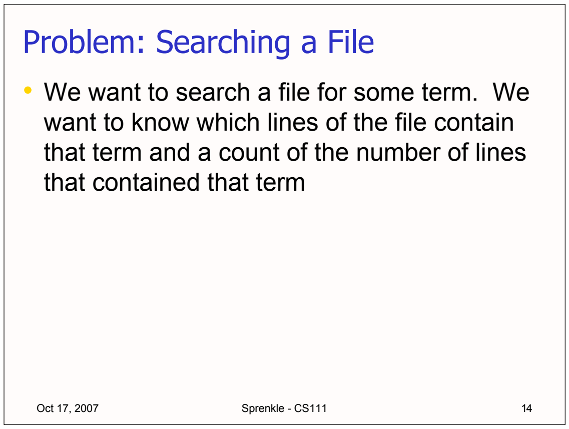
- Recall that a file is a ***sequence*** of data
- Can use a **`for`** loop to iterate through a file

A line (of type **string**) from the file
file object

```
for line in dataFile:
    print line
```

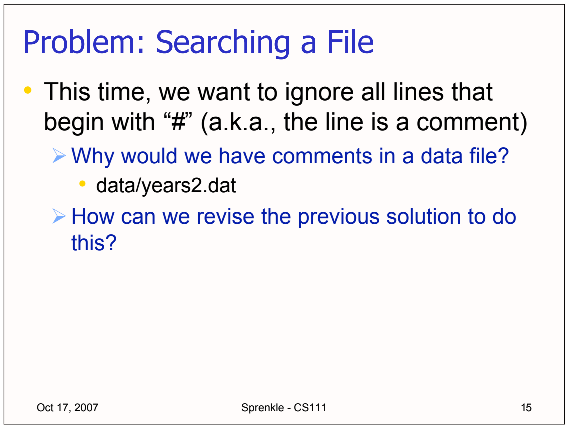
- Read as: for each line in the file, do something

file_read3.py



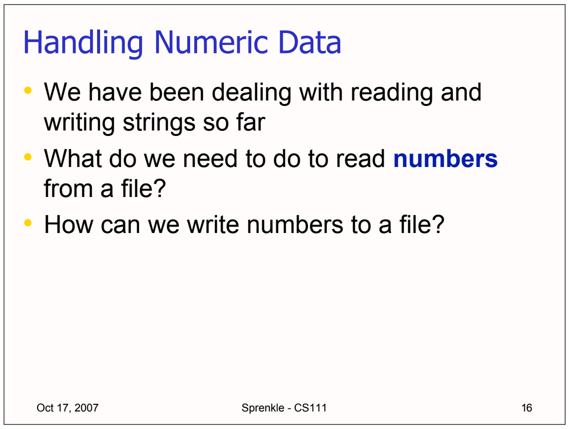
Problem: Searching a File

- We want to search a file for some term. We want to know which lines of the file contain that term and a count of the number of lines that contained that term



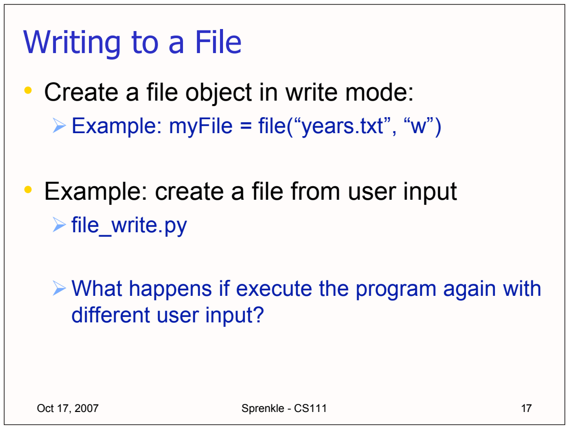
Problem: Searching a File

- This time, we want to ignore all lines that begin with “#” (a.k.a., the line is a comment)
  - Why would we have comments in a data file?
    - data/years2.dat
  - How can we revise the previous solution to do this?



Handling Numeric Data

- We have been dealing with reading and writing strings so far
- What do we need to do to read **numbers** from a file?
- How can we write numbers to a file?



Writing to a File

- Create a file object in write mode:
  - Example: myFile = file(“years.txt”, “w”)
- Example: create a file from user input
  - file_write.py
  - What happens if execute the program again with different user input?