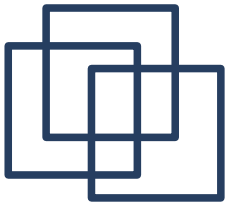
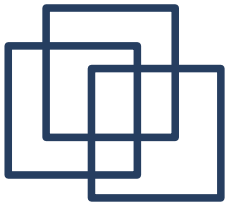
CS 1

Working with Data in Python



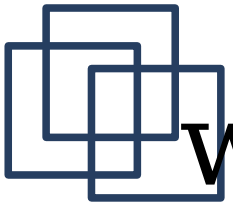
String Formatting

- Can format the items we put into strings
  - So that they look nice.
  - Format:
  - “string text {:<width>}”.format(<value>)
  - Stuff in the {} is replaced by the formatted value
  - Eg
    - str = “a string with width 4 number {:4}”
    - str.format(1)
    - Yields:
    - a string with width 4 number 1



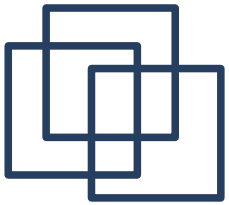
String Formatting 2

- Format floating point numbers
  - “ {:7.5}”.format(3.3333333)
  - Puts a width 7 number with 5 digits of precision
  - “ {:7.5f}”.format(3.3333333)
  - Puts a width 7 number with 5 digits after the decimal point



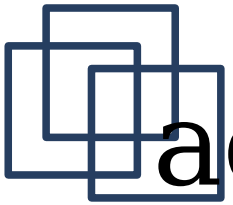
working with lots of data

- sometimes working with one piece of data at a time is fine
  - current temperature
  - name
- often want to work with lots of related data.
  - names of all students in class.
  - all grades for student x this semester
- easy way to do this in python: list



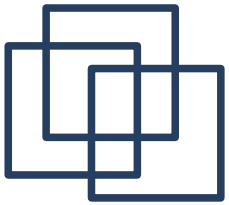
Lists

- python representation of well
  - ordered list of stuff
  - list can hold values of any data type
    - though often good to have all data in list of same type.
  - literal list
    - mylist = [“comp101”, “comp442”, “comp460”, “comp470”]
    - note list set off in []
    - items in list separated by commas.



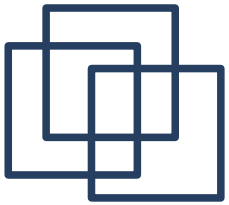
accessing items in the list

- access list items using the [] operation
- starts counting from 0.
  - mylist = [“comp101”, “comp442”, “comp460”, “comp470”]
  - print (mylist[0])
  - prints out what?
- update same way
  - mylist[3] = “comp570”



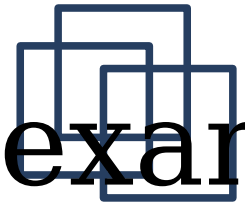
appending to lists

- can overwrite an item in a list
  - myList[0] = “new stuff”
- might also want to add to the end of the list.
  - lists also objects
  - append method for list objects.
  - myList.append(“even newer Stuff”)
  - puts that string as new last item in list



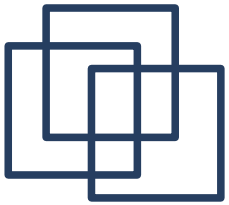
slicing lists

- can slice lists same as strings.
  - print (mylist[:2])
    - prints from beginning of list to item in slot 2 (third item in list.)
  - how will we get the 2nd and 3rd items only?



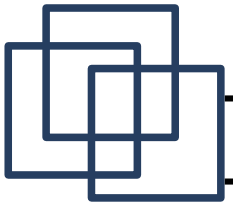
examine each list element once

- sometimes you'll want to do something with each element in the list
  - can't change the list itself, but can do something for each element.
  - called 'looping through the list'
  - do the same thing again and again for each list item.



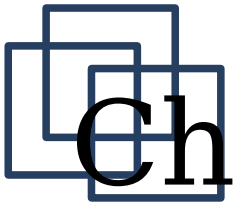
Review Questions

- Given the python code
  - question1a = “go to the store”
  - question1b = “sufferin' succotash!”
  - Use the variable question1b and what you've learned to print out Sylvester's catchphrase “Thufferin' Thuccotash!”
  - Use question1a to print out the 'little kid' version “go to the tore”
  - How would we use slicing to get the string uffer from question1b?



looping through the list

- syntax
  - for <var_name> in <list>:
    - statement
  - var_name will have a different value each time statement is called
  - example
  - for item in mylist:
    - print(“ teach %s this semester” % item)
  - note : at end of first line
  - note that statement is indented.
  - more examples?



Choosing A random Element

- There is a random module/package standard as part of python
  - Have to import it
  - import random
  - Use the choice method
  - random.choice(<pass your list here>)
    - This will return a randomly chosen element of the list.

Data

- When you have a lot of data
  - In python we know you can put it in lists
  - One of several *sequence* data types/classes
    - Linear collection
    - Each data item is preceded by exactly one and succeeded by exactly one data item (except the first and last)
    - What is the other sequence data

Sequences

- Sequences all support a common set of operations
  - Concatenation: + <sequence>+<sequence>
  - Repetition: * <sequence>*<int>
  - Indexing: [ ]
  - Length len(<sequence>)
  - Slicing [:]
  - Iteration for <new variable> in <sequence>
  - Membership check <value> in <sequence>
  - See page 343 in book for more

Python Standard Dictionary

- Python also has a second very useful collection – the dictionary
- In a list you look up an item by its position in the list
  - Use number to index in and find data
  - Key-value pair
- In a dictionary you use an arbitrary value to index
  - Often a string

Dictionary literal

- To create a dictionary via a literal
  - gradeDict= { “Jed”:50, “Ann”:70, “Ed”:20, “Kat”:75, “Ted”:80}
  - Update
    - gradeDict[“Ed”] = 72
  - Insert
    - gradeDict[“NewStudent”] = 65

Differences from Sequences

- Dictionary doesn't support
  - Slice
  - Concatenation
  - Repetition
  - length

Similarities to Sequences

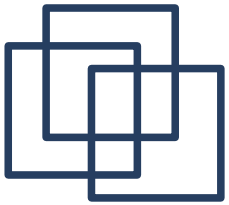
- Dictionary does support
  - Index
  - Membership
  - iteration

A special Dictionary

- Python Standard Dictionary
  - Is really powerful
  - Does a lot
  - But has drawback: by definition keys are unordered
  - What if we want to have keep track of the keys in the order we entered them?
    - OrderedDict object

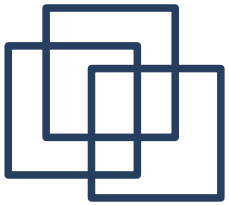
OrderedDict

- In the collections module
  - import collections
- Works just like regular dictionary except it remembers the order you added elements
- Useful traits
  - When iterating over OrderedDict, will always give same order.
  - Calling keys on a dictionary will give you a list
  - Keylist = dict.keys()
  - Keylist[-1] gives you the last one.



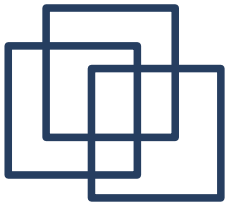
Reading a File

- Two types of files on computer
  - Text
  - binary
- Python makes reading from text files really easy (open a file)
  - Open file
  - <variable> = open(<filename>, mode)
  - Eg
    - Infile = open(“file1.txt”, 'r')



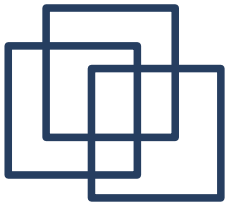
Reading file

- Read file
  - Once you have a file object – easiest
  - Way
  - File.readlines()
  - Returns a list of strings
  - Each string is a line in the file.
  -



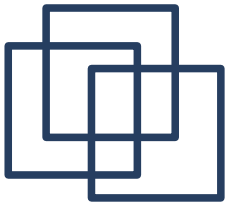
Writing a file

- Outfile = open(“myoutput.txt”, 'w')
- Print(“stuff to go in file”, file=outfile)
-



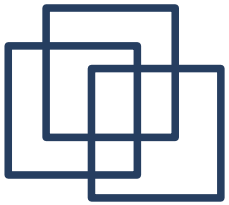
putting it all together

- lets put it all together and write a python program.
  - write a program that will print out a word in reverse.
  - how will we do this?
  - What about read in a file and print every second line to screen
  - Use sillyrec.txt



Review Questions

- Given the python code
  - question1a = “go to the store”
  - question1b = “sufferin' succotash!”
  - Use the variable question1b and what you've learned to print out Sylvester's catchphrase “Thufferin' Thuccotash!”
  - Use question1a to print out the 'little kid' version “go to the tore”
  - How would we use slicing to get the string uffer from question1b?



Lets play with some

- If there is time, you all will walk me through playing with some of this.
- Example – lets do word counts.