Outline for October 27, 2022

Reading: §7

Assignments: Homework 3, due November 10, 2022

1. Binary search example
2. Dictionary
   (a) Collection of key-value pairs
3. Creating dictionaries
   (a) Using \( d = \{ \} \)
   (b) Using \( d = \text{dict()} \)
4. Methods for dictionaries
   (a) \( k \text{ in } D: \text{True if dictionary } D \text{ has key } k; \text{ else False} \)
   (b) \( D.\text{keys}(): \text{list of keys in } D \)
   (c) \( D.\text{values}(): \text{list of values in } D \)
   (d) \( D.\text{items}(): \text{list of tuples (key, value) in } D \)
   (e) \( D.\text{get}(k, d): \text{if key } k \text{ in } D, \text{return associated value; else return } d \)
   (f) \( \text{del } D[k]: \text{delete tuple with key } k \text{ from } D \)
   (g) \( D.\text{clear}(): \text{delete all entries in } D \)
5. Example: memos
   (a) Remember how slowly the recursive Fibonacci number program \( rfib.py \) ran? Here is a faster recursive version that uses memos \([rfibmemo.py]\)
6. Sorting the dictionary
   (a) \( \text{sorted} \) sorts based on keys
7. Example: word frequency count
   (a) Unsorted \([wfc-1.py]\)
   (b) Sorted alphabetically \([wfc-2.py]\)
   (c) Sorted alphabetically, but dictionary order (note key=str.lower() in \( \text{sorted} \)) \([wfc-3.py]\)
   (d) Sorted by frequency (treat \( \lambda x: x[1] \) as an idiom to reference the value of the dictionary entry, not the key—to go from highest to lowest, replace \( x[1] \) with \(-x[1]\)) \([wfc-4.py]\)
   (e) Sorted by frequency first, then alphabetically—note use of function \( \text{alphafreq}(x) \); you can use any function here, and the parameter is the item \([wfc-5.py]\)