Data Structures and Algorithms in Python

Michael T. Goodrich
Department of Computer Science
University of California, Irvine

Roberto Tamassia
Department of Computer Science
Brown University

Michael H. Goldwasser
Department of Mathematics and Computer Science
Saint Louis University

Study Guide: Hints to Exercises

WILEY

Chapter 10 Maps, Hash Tables, and Skip Lists

Hints

Reinforcement

R-10.1) This is more challenging because __delitem__ does not return the deleted value.

R-10.2) You must rely on the iter method to access the contents of the map.

R-10.3) You should not directly call __iter__, but you can mimic its style.

R-10.4) The first insertion is $O(1)$, the second is $O(2)$, ...

R-10.5) Review the API for PositionalList from Section 7.4.

R-10.6) Think about which of the schemes use the array supporting the hash table exclusively and which of the schemes use additional storage external to the hash table.

R-10.7) Use of Python's __hash__ method is discussed on page 415.

R-10.8) There is a lot of symmetry and repetition in this string, so avoid a hash code that would not deal with this.

R-10.9) Try to mimic the figure in the book.

R-10.10) Try to mimic the figure in the book.

R-10.11) The failure occurs because no empty slot is found. For the drawing, try to mimic the figure in the book.

R-10.12) Try to mimic the figure in the book.

R-10.13) Think of the worst-case time for inserting every entry in the same cell in the hash table.

R-10.14) Mimic the way the figure is drawn.

R-10.15) Allow the user to express the maximum load factor as an optional parameter to the constructor.

R-10.16) It is okay to insert a new entry on "top" of the deactivated entry object.

R-10.17) You will need to keep track of the number of probes in order to apply quadratic probing.

R-10.18) Think of where the entry with minimum key is stored.

R-10.19) The crucial methods are __getitem__ and __setitem__.

R-10.20) Since the map will still contain n entries at the end, you can assume that each __delitem__ operation takes the same asymptotic time.

R-10.21) What happens in the case where the middle table value equals k?

R-10.22) Assume that a skip list is used to implement the sorted map.

R-10.23) Mimic the style of the figures in the book.

R-10.24) You must link out the removed entry's tower from all the lists it belongs to.

R-10.25) You will need to rely on the iter behavior to find an arbitrary element of the set.

R-10.26) For each element of the smaller set, check to see if it is contained in the larger set.

R-10.27) Something from this chapter should be helpful!

Creativity

C-10.28) The existing __setitem__ implementation can serve as a reasonable model for this new method.

C-10.29) You might provide an implementation directly within the ProbeHashMap, but you'll need to borrow some techniques from the HashMapBase class.

C-10.30) You might provide an implementation directly within the ChainHashMap, but you'll need to borrow some techniques from the HashMapBase class.

C-10.31) 1

C-10.32) Prepare a concise table of your experimental results.

C-10.33) Fortunately, a Python list can be heterogeneous!

C-10.34) Oops. We meant to say to reimplement the HashMapBase class. Feel free to subclass the nested item class.

C-10.35) You need to do some shifting of entries to close up the "gap" just made, but you should only do this for entries that need to move.

C-10.36) For part a, note that the symmetry will halve the range of possible values. For part b, note that such automatic collisions will not occur.

C-10.37) Perhaps you might define a nonpublic method that generates probe indices.

C-10.38) Consider the way find_range was implemented for SortedTableMap.

C-10.39) Try out some examples.

C-10.40) Do a "double" binary search.

C-10.41) Dovetail two binary searches.

C-10.42) Think first about how you can determine the number of 1's in any row in $O(\log n)$ time.

C-10.43) Think about first sorting the pairs by cost.

C-10.44) 1

C-10.45) Consider augmenting each node v in a higher level with the number of missing entries in the gap from v to the next node over.

C-10.47) Make sure to start with a new empty set (or make a copy of one of the two initial sets).

C-10.48) Think of how you could transform D into L.

C-10.49) Maintain a secondary PositionalList instance that represents the FIFO order

Projects

P-10.50) In a Unix/Linux system, a good place to start is /usr/dict.

P-10.51) It is okay to generate these phone numbers more-or-less at random.

P-10.52) You might consider embedding a next pointer within each item composite.

P-10.53) Sentinels can be used in place of the theoretical $-\infty$ and $+\infty$.

P-10.54) Try to make your screen images mimic the skip list figures in the book.

P-10.55) For each word t that results from a minor change to s, you can test if t is in W in $O(1)$ time.