Homework 3

You may collaborate with *one or two* other students on this homework if you wish, or work alone. Collaboration must be true collaboration however, which means that the work put into each problem should be roughly equal and all parties should come away understanding the solution. Here are some suggested ways of collaborating.

- Pair programming. Two or three of you look at the same screen and only one of you operate the keyboard. The one at the keyboard is the "driver" and the other is the "navigator". The driver explains everything they are doing as they do it and the navigator asks questions and makes suggestions. If the navigator knows how to solve the problem and the driver does not, the navigator should not dictate solutions to the driver but instead should tutor the driver to help them understand. The driver and navigator should switch roles every 10-15 minutes. Problems solved this way can then be individually submitted.
- Code review. The members of the collaborative each try to solve the problem independently. They then take turns analyzing each other's code, asking questions trying to understand each other's algorithms and suggesting improvements. After the code reviews, each of you can then fix your code using what you learned from the reviews. Do not copy code. If the result of code review is that your code needs changes to be more like your partner's, do not copy it. Instead recreate your own variant without looking at your partner's.

The goal is to learn enough from one another so that you each can do similar work independently in an exam situation.

If you want a collaborator but don't know people in the class, you can ask on Discord and/or use the group-finding post on Piazza.

Homework Exercises

Completion of these tasks by 11:59pm February 27 is worth approximately 1-2% of your course grade. No late homework will be accepted.

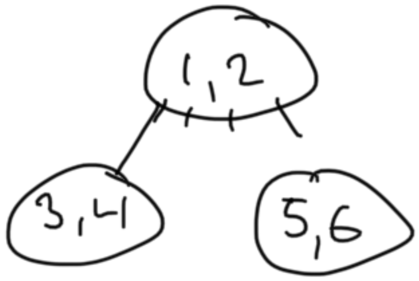
1. (graded) Consider the persistent list class as described in class and implemented in this code. Add to the provided list135 class a len() method which returns the number of elements in the list. Do not alter any of the other code in the class. For submission, a Mimir assignment will be provided shortly before the deadline, but in the meantime test your code as you see fit.

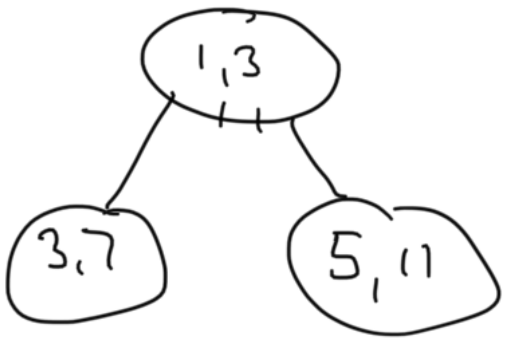
2. (graded) Consider the Hash Array Mapped Trie (HAMT) as described in class and implemented in this code. Add to the provided HAMT class a len() method which returns the number of keys in the trie. Do not alter any of the other code in the class. For submission, a Mimir assignment will be provided shortly before the deadline, but in the meantime test your code as you see fit.

3. (graded) Consider the Hash Array Mapped Trie (HAMT) as described in class and implemented in this code. Add to the provided HAMT class a get(k) method which returns the value k is mapped-to or None if k is not a key in the HAMT. Do not alter any of the other code in the class. For submission, a Mimir assignment will be provided shortly before the deadline, but in the meantime test your code as you see fit.

4. (graded) Consider the Hash Array Mapped Trie (HAMT) as described in class and implemented in this code. Add to the provided HAMT class a map(f) method which applies f to every node's key and value and returns a duplicate trie with each node's value updated with the result of f applied to that node's key and value. The function f passed into map will take a key and value as parameters and return a new value. Do not alter any of the other code in the class. A Mimir assignment will be provided shortly before the deadline, but in the meantime test your code as you see fit.

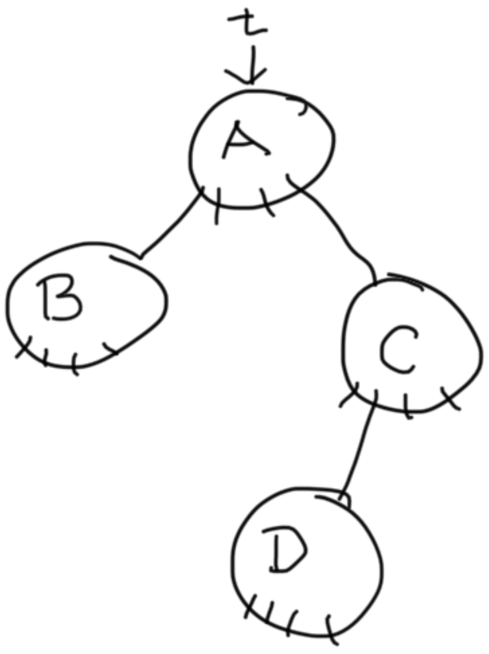
As an example, if `lambda k, v: k+v` were the function f and map(f) were called on the left trie below, the right trie should be the result. Note that the keys have all stayed the same, but the values have been changed. Also, remember that this is a persistent data structure, so the left trie should remain unchanged.





For submission, a Mimir assignment will be provided shortly before the deadline, but in the meantime test your code as you see fit.

5. (ungraded) The following figure is an HAMT with four keys (their associated values are not shown). What trailing bits can you deduce for the hashes of A, B, C, and D?



6. (ungraded) Draw the resulting HAMT when the values have been added with the keys A, B, C, D, E, F (in that order). Your drawing should look similar to the one in Problem 5 (ie, ignoring the values and beginning with root A). Use the following hashes.

```
h(A) = 10010000
h(B) = 10110001
h(C) = 11110011
h(D) = 10000111
h(E) = 10011100
h(F) = 01001101
```

7. (ungraded) Begining with the trie in Problem 5, let's say `s = t.set(E, "e")` is invoked. Let `hash(E) = 00110011`. Draw the resulting tries s and t, including all nodes and links between them. (Don't include any values in your drawing.)

8. (ungraded) Draw all nodes in the resulting structure (including their data element) and indicate which nodes are pointed at by each variable.

```
t = list135()
a = t.cons("A")
b = a.cons("B")
c = a.cons("C")
d = c.cons("D")
e = d.rest()
f = e.cons("F")
```

Ungraded solutions:

After working through the ungraded problems, compare your solution to those provided. If they differ, determine how and why they do. If you need help understanding the solutions, just ask.

http://krovetz.net/135/module_hamt/sol.html