Individual Assessment 2: Balanced Trees

This is an individual assessment, and, as the name suggests, must be completed individually. Specifically, you're not allowed to work with a partner, and you should not discuss these problems with other students in CS166. However, the course staff are happy to answer clarifying questions on Ed-Stem (if you do, please post the question privately) or in our office hours.

Due Thursday, April 21\textsuperscript{nd} at 3:15PM Pacific time.
Although we've spent a lot of time talking about balanced trees, they are not the only data structure we
can use to implement a sorted dictionary. Another popular option is the **skiplist**, a data structure consisting of a collection of nodes with several different linked lists threaded through them.

Before attempting this problem, you'll need to familiarize yourself with how a skiplist operates. We recommend a combination of reading over the Wikipedia entry on skiplists and the original paper “Skip Lists: A Probabilistic Alternative to Balanced Trees” by William Pugh (available on the course website). You don't need to dive too deep into the runtime analysis of skiplists, but you do need to understand how to search a skiplist and the normal (randomized) algorithm for performing insertions.

The original version of the skiplist introduced in Pugh's paper, as suggested by the title, is a randomized data structure and gives expected $O(\log n)$ performance on each of the underlying operations. In this problem, you'll use an isometry between multiway trees and skiplists to develop a deterministic skiplist that supports all major operations in worst-case time $O(\log n)$.

i. Briefly explain how to encode a multiway tree as a skiplist. Include illustrations as appropriate.

To design a deterministic skiplist supporting insertions, deletions, and lookups in time $O(\log n)$ each, we will enforce that the skiplist always is an isometry of a 2-3-4 tree.

ii. Using the structural rules for 2-3-4 trees and the isometry between multiway trees and skiplists you noted in part (i) of this problem, come up with a set of structural requirements that must hold for any skip list that happens to be the isometry of a 2-3-4 tree. To do so, go through each of the structural requirements required of a 2-3-4 tree and determine what effect they will have on the shape of a skiplist that's an isometry of a 2-3-4 tree.

Going forward, we'll call a skiplist that obeys the rules you came up with in part (ii) a **2-3-4 skiplist**.

iii. Give two explanations as to why a lookup on a 2-3-4 skiplist takes worst-case $O(\log n)$ time. First, do so purely by making reference to the rules you came up with in part (ii) and without referencing 2-3-4 trees. Second, do so by using the isometry from 2-3-4 trees to 2-3-4 skiplists.

iv. Show the effect of inserting the value 8 into the 2-3-4 skiplist shown below.

```
1   3   5   7   9   11  13  15  17  19
```

Congrats! You've just used an isometry to design your own data structure! If you had fun with this, you're welcome to continue to use this isometry to figure out how to delete from a 2-3-4 skiplist or how to implement split or join on 2-3-4 skiplists as well.

Fun fact: Here, we started with a deterministic balanced tree (2-3-4 trees) and derived a deterministic skiplist. You can also run this process in reverse by starting with a randomized skiplist and deriving a randomized balanced binary search tree (the **zip tree**). This could make for a great final project topic!