Steven Friedman
Lab 3 (skip)

PART 1

I created a Node class and a Pillar class. A Node is one element in a Pillar's array. It has a next pointer, and it also keeps track of what Pillar it is a part of, which is necessary for skipping from one Pillar to another, as opposed to just between Nodes. It used to have a previous pointer, but this is commented out for part 2. A Pillar constructs an array of the size of a given height composed of Nodes that have that Pillar as their Pillar and initially null next pointers. Pillar also keeps track of its event. The array itself is called tower, as naming it pillar would be really confusing.

Constructor
The constructor creates head and tail Pillars with year values +/- 100,000 and initial height 1. They are then linked together. For explanation of extensible head/tail, see PART 2.

Insert
insert creates a new Pillar of height t (randomHeight()) for the input event and inserts it in the appropriate place. At each level it jumps from Pillar to Pillar until the next Pillar's year isn't greater than the input event's and if the level it's checking is less than the height t, it will insert the event's Node at that level into that level's linked list.

Remove
remove removes all Pillars that have a given input year. It iterates through each level of the skip list, and keeps track of if the year occurs in that level, and if so, what the first and last occurrences of that year are. The "first" Pillar is actually the Pillar before the first occurrence of the year, so that its Node's next pointer can be directed at the last occurrence's next pointer.

Find Most Recent
findMostRecent finds the last event or series of events that occurred in a year <= an input year. I ran into a lot of trouble, and my implementation ending up getting very complicated. The x Pillar will end up being the year that we want to return Pillars for. The xprev Pillar keeps track of the Pillar before x, so that we can find all of the Pillars from that year, not just the ones x and after. The first Pillar keeps track of the Pillar that is the first occurrence of a certain year as well. This is necessary because the conditional (year > x.next's year) is true when there are no events of the input year, and xprev could become the same year as x when this happens. In the end if this happens, xprev resets to first's Pillar. The events of Pillars with the same year as x are then added to an array and returned. I had to make a special case to return null if only the head was present.

Find Range
findRange finds all events that occurred between (inclusive) the first and last input years. The implementation is very similar to findMostRecent, except it required no first Pillar,

and instead of stopping when it hit x's year, starts right before the start Pillar's year and finishes when the year is no longer <= last.

PART 2

Extensible head/tail pointers

I created a new method doubleEnds to clean up the insert method. Every time a random height t is generated that is greater than the height of head and tail, this method is called. This also required creation of a new instance variable t that kept track of this new height. The height of head and tail is doubled until > t. New head and tail Pillars are created with the same events as the old ones and linked, and the next Pointers from head are transferred over. While all the Nodes in the skip list that point to the old tail don't point to the new one, it's functionally the same. They point to a Pillar with the same event and with a null next pointer in its Nodes.

Singly linked skip list

This wasn't too bad, as I didn't used prev pointers in too many places, just to insert and remove from the list. I tried getting rid of the tail pointer, but this was really challenging, and I couldn't find a way to get rid of null pointer exceptions and still maintain the functionality of the skip list.