Part One

In this project, I implement the skip Lists. It has head and tail pillar and insert, remove, findmostrecent and findrange functions. The last two functions can both find the Event[] list with all the nodes that satisfy the requirement.

In particular, in the findRange function, I use x and y to track the node. When y is between first year and last year, x is before the first year. At 1th level, it record the next node of x and other all nodes which are < and = the last year as Event[] result and return them.

In the findMostRencent function, I not only track the x, y but also check the node after y. When y.next.year=y.year, it move y to the next. When y.next.year > the recent year, it begin to record all the next nodes of x before the node's year > the recent year as Event[] result and return them.

Part Two

(a) In this part, I implement a new type of head and tail with original height 1. When the height of inserted pillar is higher than 1, it will build new head and tail pillars with double height and copy all the node in original head and tail to the new head and tail, then continue the operation of insertion.

(b) In this part, I improve my code by giving up the usage of previous pointer. It uses only one pointer next. In the findMostRecent function. I use x to store the node before the node which have the most recent year, at level 1, return the Event[] result stored all the nodes after x and with the most recent year.