My Setup : Vector <list <Node>>

Node { int value
list <Node>::iterator down

why?

Node
- int to store value
- iterator to make vertical connections

List
- Doubly Linked List
  - Easy to access and insert/delete elements

Vector
- Good Container because of random access

Tip #1 : make parameterized constructor for Node

Tip#2: There is no Default value for an iterator

Make "down" connections with -inf and inf nodes

$$\begin{tikzcd}
10 & -\infty \arrow[r] \arrow[d] & \infty \arrow[d] \\
\vdots & {} & {} \\
2 & -\infty \arrow[r] \arrow[d] & \infty \arrow[d] \\
1 & -\infty \arrow[r] \arrow[d] & \infty \arrow[d] \\
0 & -\infty \arrow[r] & \infty
\end{tikzcd}$$

1. push_back an empty list<Node> to vector

2a. push_back two nodes $-\infty$ and ∞ for bottom row

```
_vec[0].push_back(Node(-inf, vec[0].end()))
```

$\uparrow$ 1st row

$\uparrow$ $-\infty$

$\uparrow$ for 1st row it doesn't matter where you point iterator to

- Do same for ∞ on 1st row

2b. You can use a set size skip list so I use for loop to make other levels

* If you are going to use a set size, I recommend using ≥ 10 because there is

less than 0.1% chance it will get that high

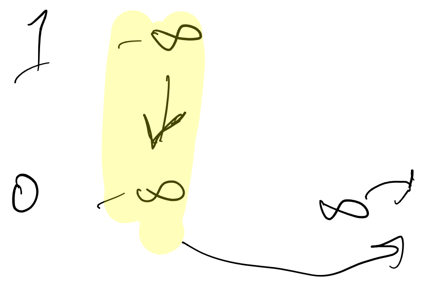
start at 1 not 0

for (i = 1   i < size ...)   Point to list before

- push_back empty list to vector
- push_back (Node(-inf, vec[i-1].begin())) to list

end

e.g. i=1

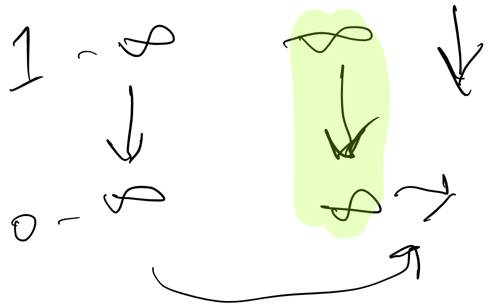


- push_back (Node(inf, --vec[i-1].end())) to list

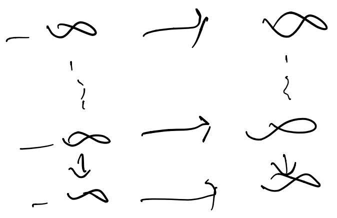
points to 1 before end (points to last element)

end

i=1



Now skip list should have correct connections



Insert - the way I did insert is top down
that's why I only use one iterator

1. use coin flip (random-number generator) to find heads/tails. keep local int to keep track of what level to start inserting ON

2. use 2 local list<Node> iterators
   - one for traversing the list
   - one to help make "vertical" connections

```
list<Node>:: iterator it = vec[level].begin()

list<Node>:: iterator temp = vec[level].end()
```

Set at end so you do not accidentally change or modify Nodes in list

Example: { 9 H T

$$\begin{tikzcd}
10 & -\infty \arrow[r] \arrow[d, dotted] & \infty \arrow[d, dotted] \\
1 & -\infty \arrow[r] \arrow[d] & \infty \arrow[d] \\
0 & -\infty \arrow[r] & \infty
\end{tikzcd}$$

No so increment it

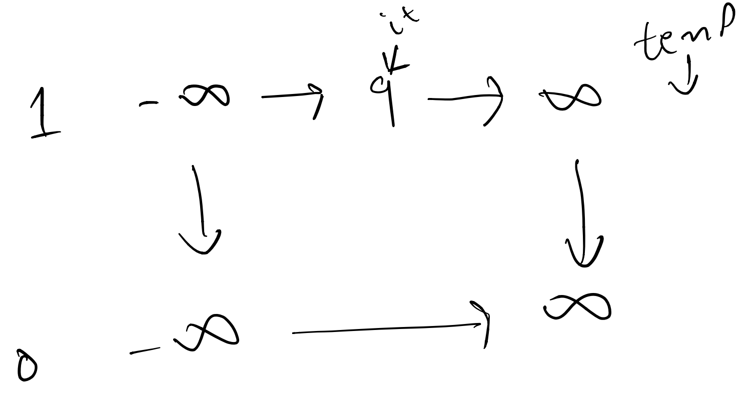
Insert 9 $9 < -\infty$?   temp

$$\begin{tikzcd}
 & \text{it} \arrow[d] & \text{temp} \arrow[d] \\
1 & -\infty \arrow[r] \arrow[d] & \infty \arrow[d] \\
0 & -\infty \arrow[r] & \infty
\end{tikzcd}$$

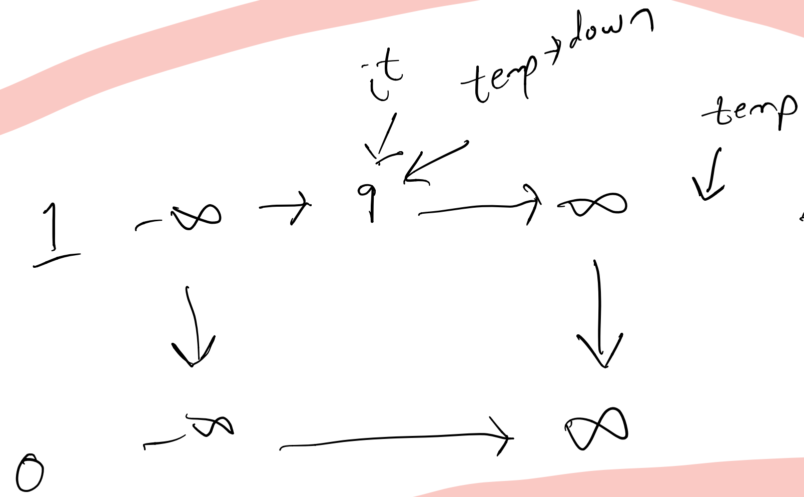
$9 < \infty$?

$$\begin{tikzcd}
 & & \text{it} \arrow[d] & \text{temp} \arrow[dl] \\
1 & -\infty \arrow[r] \arrow[d] & \infty \arrow[d] & \\
0 & -\infty \arrow[r] & \infty &
\end{tikzcd}$$

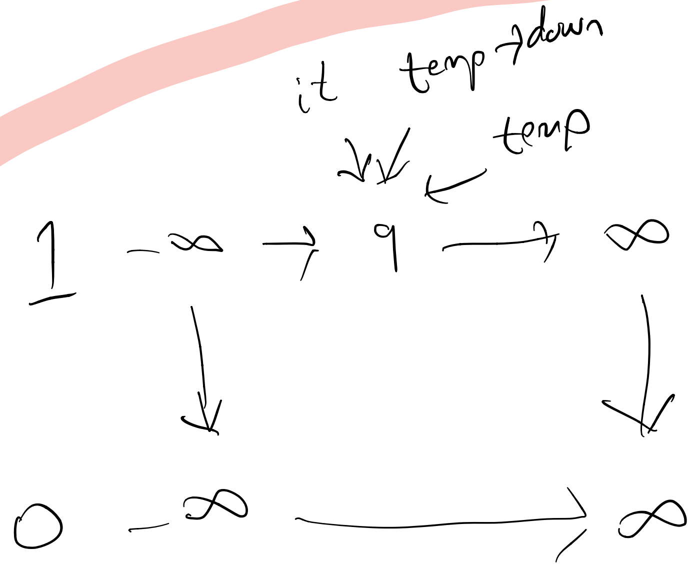
Yes so insert before ∞ and point it to value being inserted



* Don't Forget temp is list<Node>::iterator so you have access to the data members of the Node as well



* point temp's "down" iterator to it



* point temp to it

* move it to next level

(go left) – decrement it

(go down) – point it to "it→down"

$$\begin{tikzcd}
 & it \arrow[d] & temp \arrow[d] & & \\
1 & -\infty \arrow[r] \arrow[d] & 9 \arrow[r] & \infty \arrow[d] & \arrow[l, bend right] \\
0 & -\infty \arrow[rr] & & \infty &
\end{tikzcd}$$

$$\begin{tikzcd}
 & & temp \rightarrow down \arrow[d] & & \\
 & & temp \arrow[d] & & \\
1 & -\infty \arrow[r] \arrow[d] & 9 \arrow[r] & \infty \arrow[d] \\
0 & -\infty \arrow[rr] & & \infty \\
 & it \arrow[u] & & &
\end{tikzcd}$$

$9 < -\infty$?

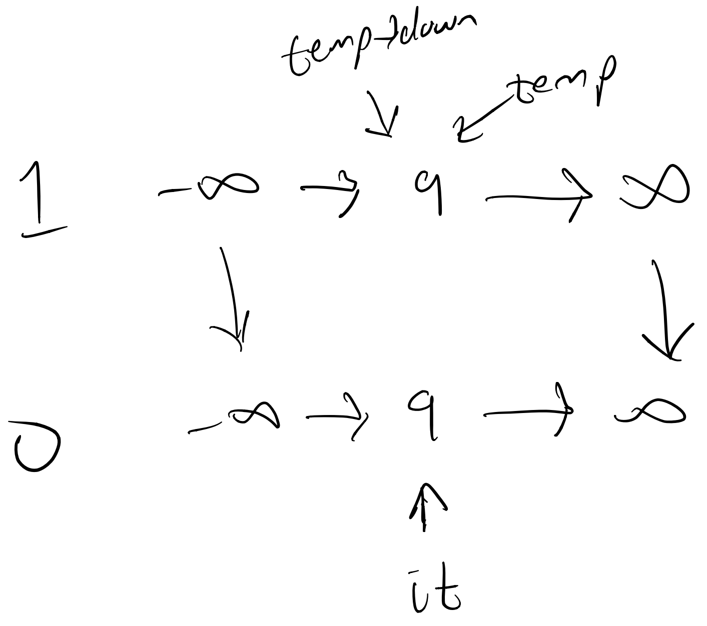
Nope

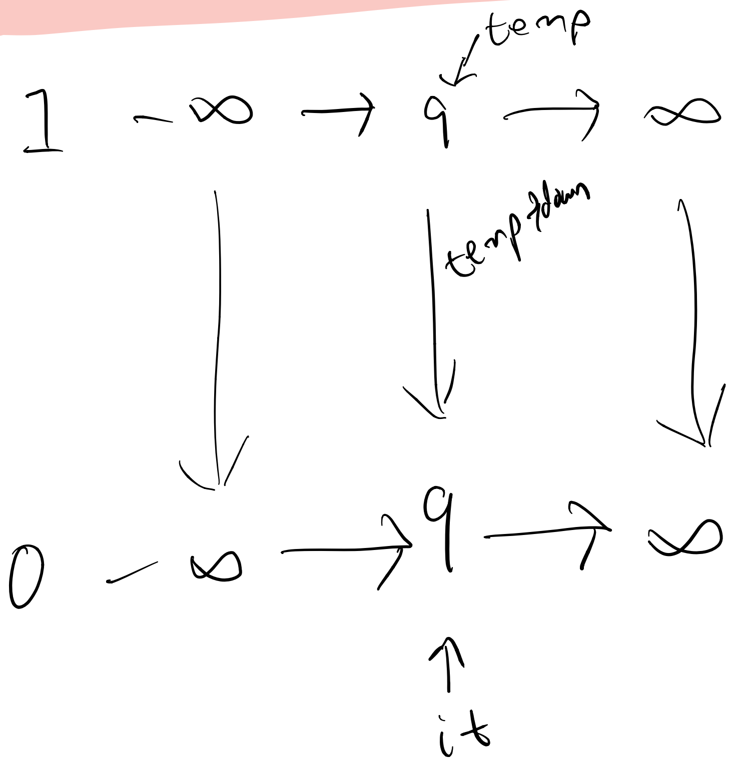
$*$ Repeat Steps

$$\begin{tikzcd}
 & & temp \rightarrow down \arrow[d] & & \\
 & & temp \arrow[d] & & \\
1 & -\infty \arrow[r] \arrow[d] & 9 \arrow[r] & \infty \arrow[d] \\
0 & -\infty \arrow[rr] & & \infty \\
 & & & it \arrow[u]
\end{tikzcd}$$

$9 < \infty$?

yes

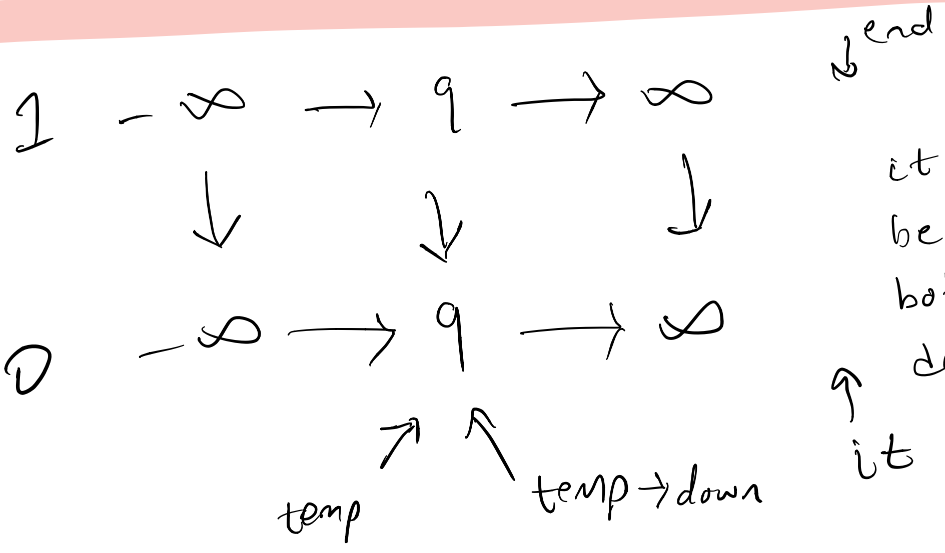




* Set temp→down to it to make vertical connection

* temp is 9[1] so temp→down is 9[1] pointing to 9[0]

End product



it at end because bottom row down points there