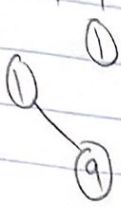
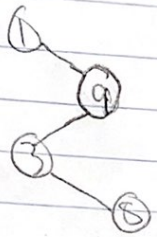
1. 1, 9, 3, 8, 2, *, 3



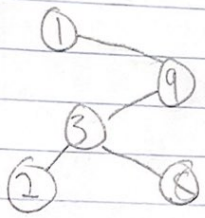
insert 1.

insert 9, goes to the right

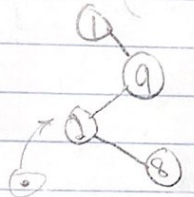
insert 3, goes to left of 9



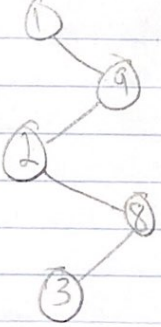
insert 8, goes to right of 3.



insert 2, goes to left of 3.



delete 3, 2 moves up in rank to take 3's spot



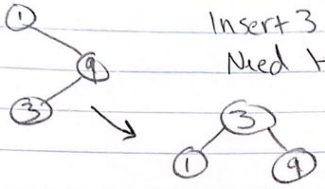
insert 3, 3 is now to the left of 8.

1, 9, 3, 8, 2, 8, 3

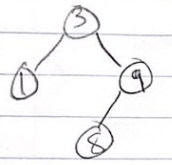
2.

Insert 1.

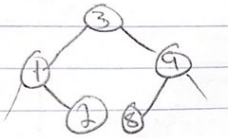
Insert 9 to the right.



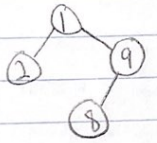
Insert 3, to the left of 9.
Need to rotate to stay balanced.



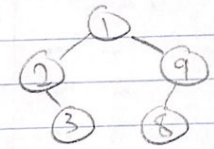
Insert 8 to the left of 9.



Insert 2 to the right of 1.



delete 3, still balanced I think.
(2) ranks up.



insert 3, balanced again I think.

3.
```
q = lambda l: q([x for x in l[1:] if x <= l[0]] + \
    [l[0]] + q([x for x in l if x > l[0]]) \
    if l else []
```

4.
```
def perm(word, step=0):
    if step == len(word):
        print("".join(word))
    for i in range(step, len(word)):
        wordcopy = [x for x in word]
        wordcopy[i], wordcopy[step] = wordcopy[step], wordcopy[i]
        perm(wordcopy, (step+1))
```