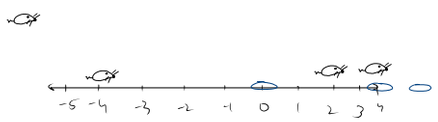
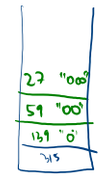
Rev 1- Morris Pre

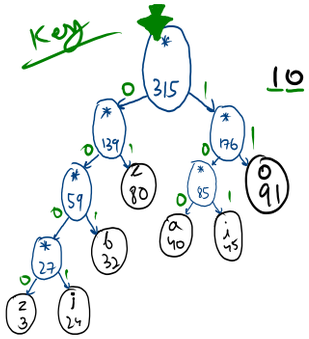



```
public void dfs(HuffmanNode node,ArrayList<String> ans,Str
    if(node.left==null && node.right == null){
        ans.add(psf);
        return;
    }
    dfs(node.left,ans,psf+"0");
    dfs(node.right,ans,psf+"1");
}
```



0000
0001

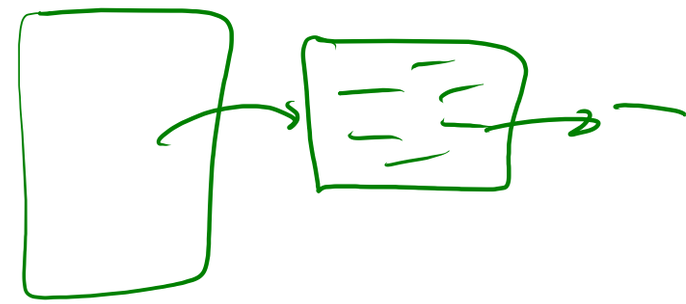
Key

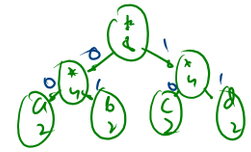


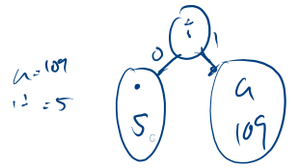
100 0101 101 11 0001 11001

a c k i o j o b

256 256 256 256 256







1) Min PQ
2) Insert all nodes in (char, freq, left, right)
3) Start popping until only 1 node is left
   a= pq.remove
   b= pq.remove
   mn= new node('*', a.freq + b.freq)
   mn.left = a
   mn.right = b
   pq.add(mn)
4) root = pq.remove();

0000, 0001, 001, 01, 100, 101, 11

1 2 3 4 5 6 7

1 2 3 7

4 5 6

pq= [4, 2, 3, 5, 8, 7]
cost= [16, 3, 1, 2, 5, 3]

start=2
cost= 7+8=5+13=19

pq= sort

pq= 7 3 5 6 10 10
cost= 6 3 4 9 3 1
start= 4
pq= 2 5+10=15

for(i → n-1){
  start += pq[i]
  start -= cost[i]
  if(start <0){
    start -=1
    start=0
  } else {
    if(start -=1) start = 0;
  }
}

Java will always try to sort
in Ascending order

Arrays.sort(arr) → O(n log n)
PriorityQueue<Integer> pq → O(n log n)

(a, b) → x
- +ve a>b
- 0 a==b
- -ve a<b

int → 5

a=5
b=10
(5, 10)
a=b-5
b=a+5
b a

a=b
b=a

a=7
b=3

a=b-7=-3
b=a+b=10

5

0000, 0001, 001, 01, 100, 101, 11