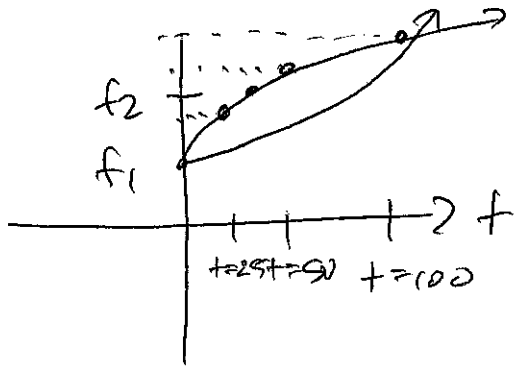
Binary Search

$$\frac{f_2 - f_1}{f_1 f_2} = at + b(1 - e^{-ct})$$




```
while (high - low > epsilon) {
    mid = (low + high)/2;
    if (f(mid) < target)
        low = mid;
    else
        high = mid;
}
```

```
for (int i=0; i<100; i++) {
    mid = (low + high)/2;
    if (f(mid) < target)
        low = mid;
    else
        high = mid;
}
```

guarantees it'll end
better dealing w/ floating pt error

Careful Approach

Upto 8 planes

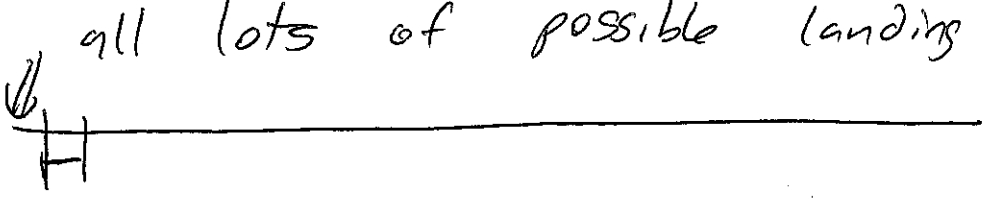
P_1 [~~20~~, 35]
P_2 [5, 30]
P_3 [25, 50]
P_4 [20, 40]
P_5 [40, 60]

Maximize the minimum interval btw any pair of landings.

P_2 5
P_1 15 → 10
P_4 25 → 10
P_3 ~~30~~ 35 → 25
P_5 60

→ 10

① NOT SURE WHAT ORDER TO LAND
② HARD TO "MAXIMIZE" gap since there all lots of possible landing times.



→ EASY- TRY ALL PERMUTATIONS!

P_2, P_1, P_4, P_3, P_5

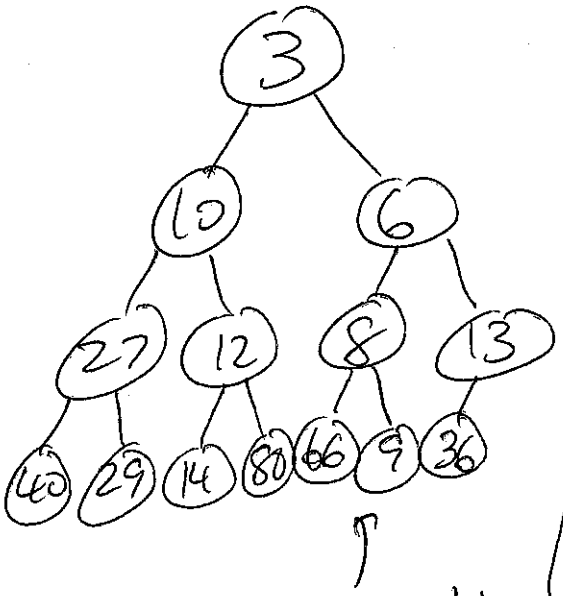
→ unclear when to definitely land second plane!

→ Can I land the planes in this order with a minimum gap of 13? ~~20~~.

5 18 33 46 59
P_2 5, P_1 ~~18~~, P_4 31, P_3 44, P_5 57

14, P_2 5, P_1 19, P_4 33, P_3 46, P_5 60

Binary Heap



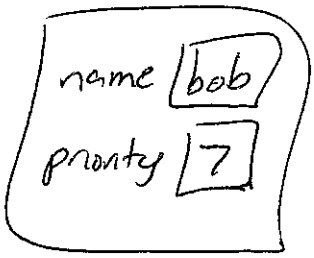
Tree Representation

Array representation

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	3	10	6	27	12	8	13	40	29	14	80	66	9	36

node i's left child is 2i
node i's right child is 2i+1
node i's parent is i/2.

nodes



Heap property - all values in a node's subtree are bigger than it.

Structural property - must be complete binary tree (fill in each "row/level", left to right)