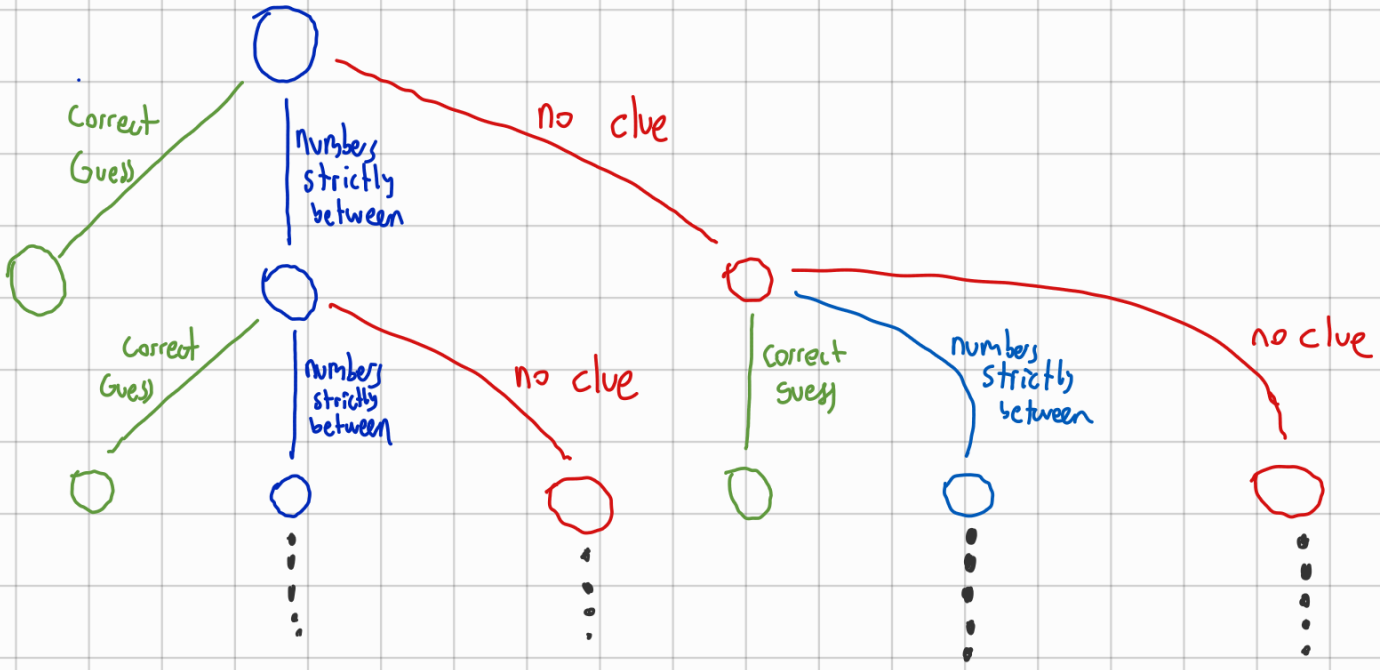
Exercise 5.2

After each guess : i) Bob guesses correctly

ii) Alice tells numbers strictly between

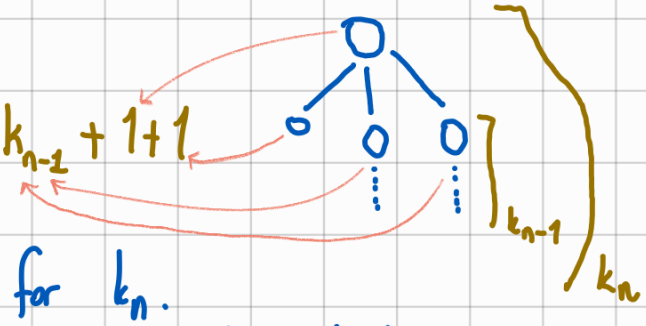
iii) Alice doesn't give any clue



k_n = maximum number of vertices in a tree of depth $n \in \mathbb{N}_0$.

$k_0 = 1 \qquad k_1 = 4 \qquad k_2 = 10$

$k_0 = 1, \quad \forall n \geq 1 \quad k_n = 2k_{n-1} + 1 + 1$



Find a non-recursive formula for k_n.
First guess it as $3 \cdot 2^n - 2$, then prove by induction.

Total numbers of pairs Alice can choose is $\binom{200}{2} = \frac{200 \cdot 199}{2} = 19900$

Total numbers of pairs Alice can choose is $\binom{200}{2} = \frac{200 \cdot 199}{2}$

The tree representing Bob's strategy must have at least 19900 leaves.

Note that the depth of the tree is the number of guesses Bob needs to make in the worst case.

$k_{12} = 12286$

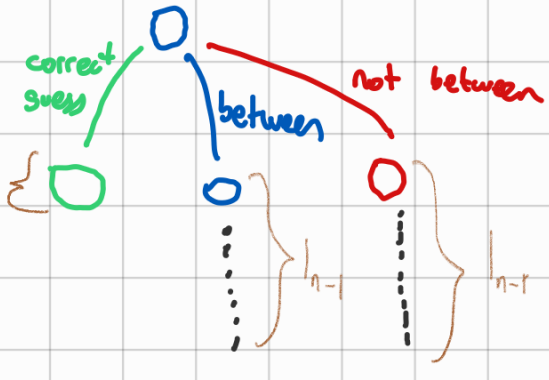
Bob cannot certainly win in at most 12 attempts.

We need 19900 leaves but the decision tree with depth 12 has < 12286 leaves.

b) If the decision tree with depth 14 has at least 19900 leaves, then Bob can certainly win in at most 14 attempts. Let's determine the number of leaves.

f_n : maximum number of leaves in a tree of depth n.

$$f_1 = 1, \quad \forall n > 1 \quad f_n = 2f_{n-1} + 1$$



Guess $f_n = 2^n - 1$, then prove by induction.

So, the tree with depth 14 has $2^{14} - 1 = 16383$ leaves. Hence Bob cannot certainly win in at most 14 attempts.

Exercise 5.3

a) After each iteration of the while loop, the depth of N gets decreased by 1. Since H is a complete binary tree, it has at most $O(\log n)$ depth. Hence N can be pushed $O(\log n)$-times up.

b)

i) Algorithm terminated because N_{stop} was the root node. Heap condition satisfied because N_{stop} is the root node

ii) Algorithm terminated because key(P) ≥ key(N_{stop}). So N_{stop} satisfies the heap condition. Besides all nodes with depth less than or equal to depth of N_{stop} have their keys unchanged. So these keys still satisfy heap condition.

c) Induction on t, number of iterations of while loop. Base case is $t = 0$.

Exercise 5.4

a) Linked List. Push & Pop requires $O(1)$ time.

b) Doubly Linked List Enqueue & Dequeue requires $O(1)$ time