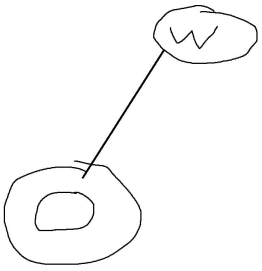
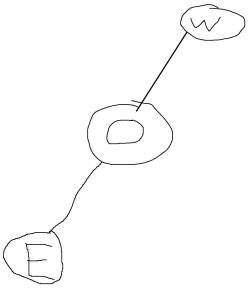
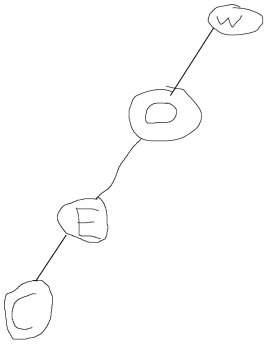
Först har vi W, och eftersom trädet är tomt så sätter vi den som roten(root). Sedan tar vi nästa input O och kollar ifall den är större eller mindre än W, och eftersom O är mindre(alfabetisk ordning) så går den åt vänster om W



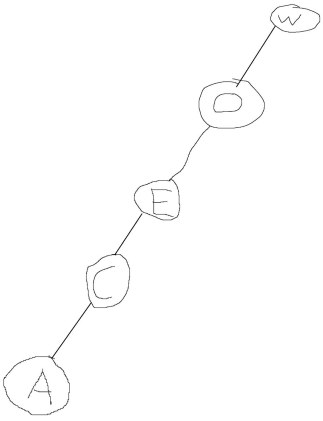
Då får vi att O grenar ner här. Sedan tar vi nästa, E och börjar med att jämföra överst med W, E är mindre så då går vi vänster och jämför med O.



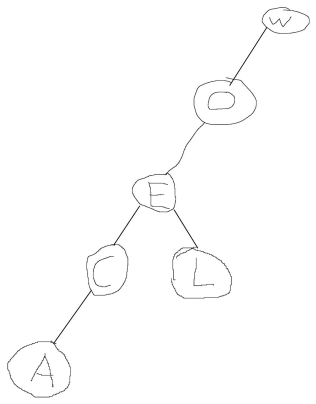
E är mindre och det finns inget vänster nere om O så vi placerar E där.



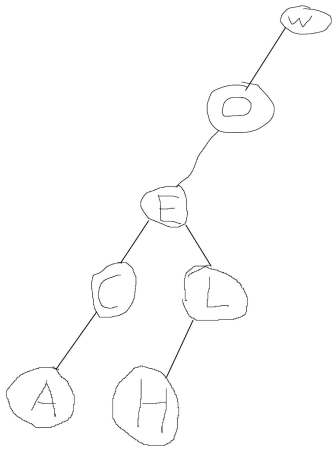
Sedan gör vi detsamma med C och hamnar nedanför vänster om E.



Sedan gör vi detsamma med A och hamnar nedanför vänster om C.



Sedan jämför vi L med W, går vänster. Jämför L med O, går vänster. Jämför L med E, och eftersom L är större så går vi höger men eftersom höger om E är tomt placerar vi L där.



Sedan gör vi detsamma med H, så den går vänster vänster höger till L och eftersom H är mindre än L så placerar vi den till vänster nere.

Prefix: W O E C A L H
Infix: A C E H L O W
Postfix: A C H L E O W

2:
Det jämförs med System.nanotime funktion för att ta tid. Även om de sorteras olika så måste man gå igenom hela ST/BST för att hitta det mest förekommande ordet eftersom att allt är sorterat efter nyckel och inte value. Så teoretiskt sätt bör det inte vara någon större skillnad.
Graf:https://docs.google.com/spreadsheets/d/1m3qVGifkF5xmCZFUp0K7L6tDOU1Y3ewiaoxJueOcUXM/edit?usp=sharing