CS 315, Day 37 – Problem Day

1. What is an optimal Huffman code for the following set of frequencies, based on the first 8 non-zero (sometimes 0 is considered the first) Fibonacci numbers? Can you generalize your answer to find the optimal code when the frequencies are the first n Fibonacci numbers?

a:1 b:1 c:2 d:3 e:5 f:8 g:13 h:21

2. You have a deck of cards that consists of one ace, two deuces, three 3's, and on up to nine 9's.

How many cards are there?

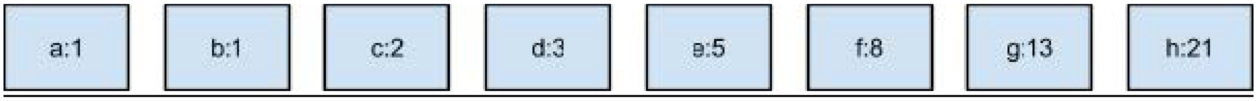
What is their sum?

Design a strategy that minimizes the expected number of questions asked in the following game: Someone draws a card from the shuffled deck, which you have to identify by asking questions answerable with yes or no.

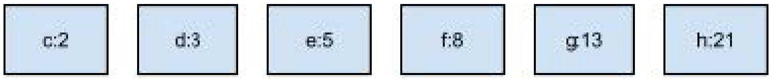
3. Determine a ***Longest Common Subsequence*** (LCS) of $< 1, 0, 0, 1 >$ and $< 0, 1, 0, 1 >$.

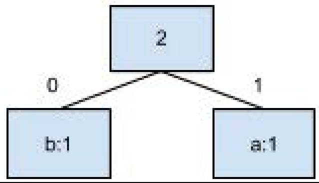
4. Construct the (1) string matching automaton and (2) the state-transition table for the pattern $P = aabab$ over the alphabet $\{a, b\}$.

1. ***initially***

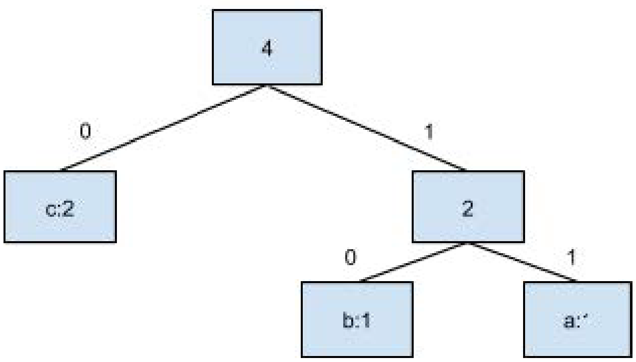


coalesce the two smallest nodes

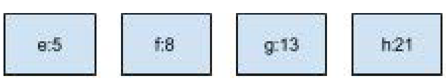


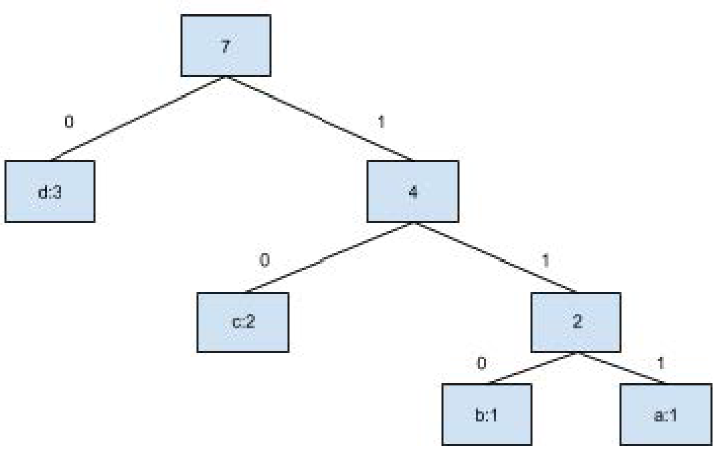


coalesce the next two smallest nodes

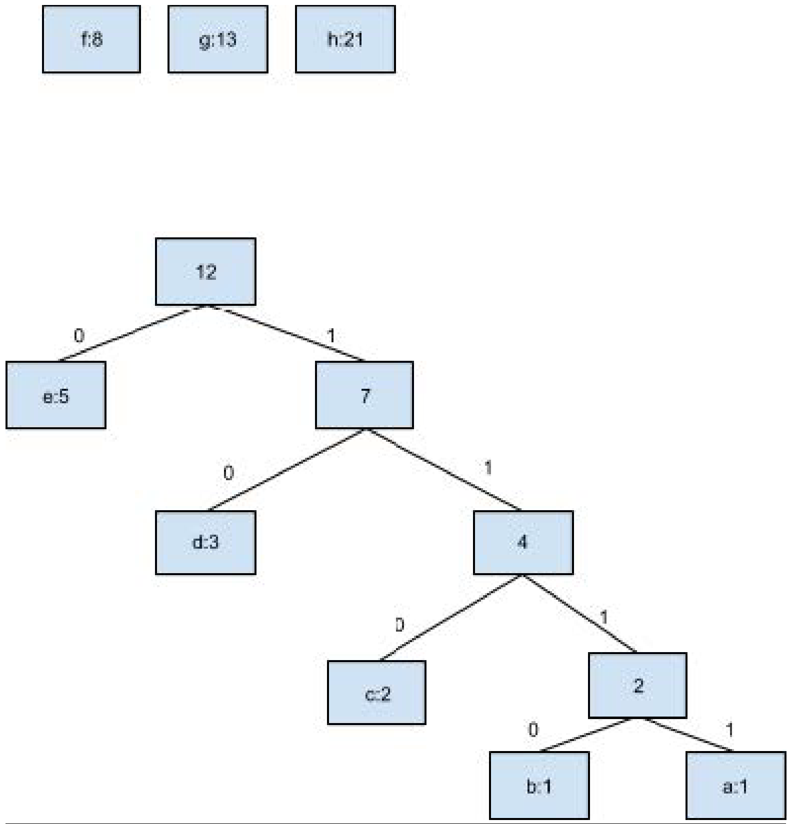


coalesce the next two smallest nodes

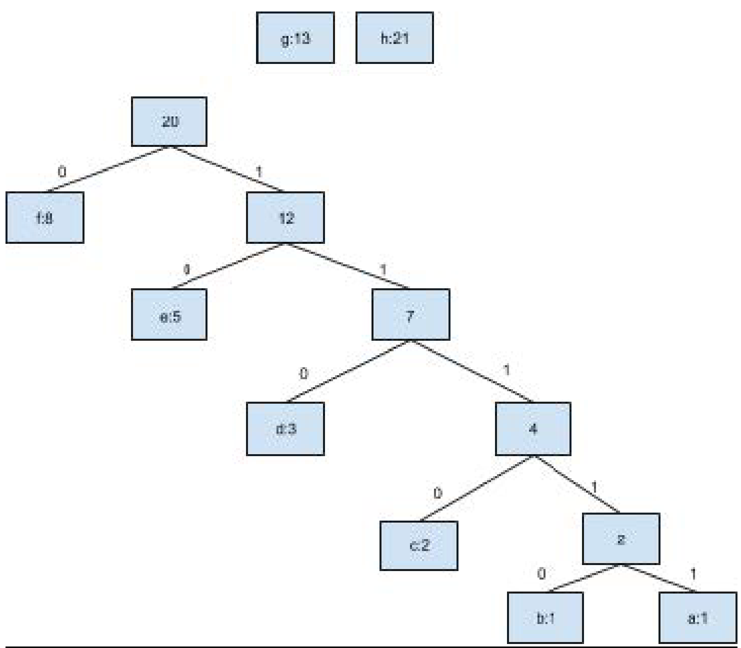




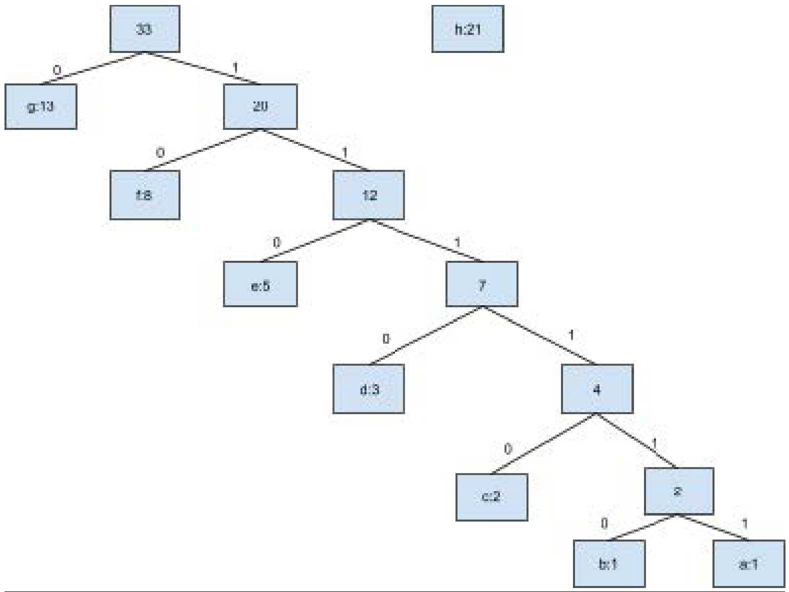
coalesce the next two smallest nodes



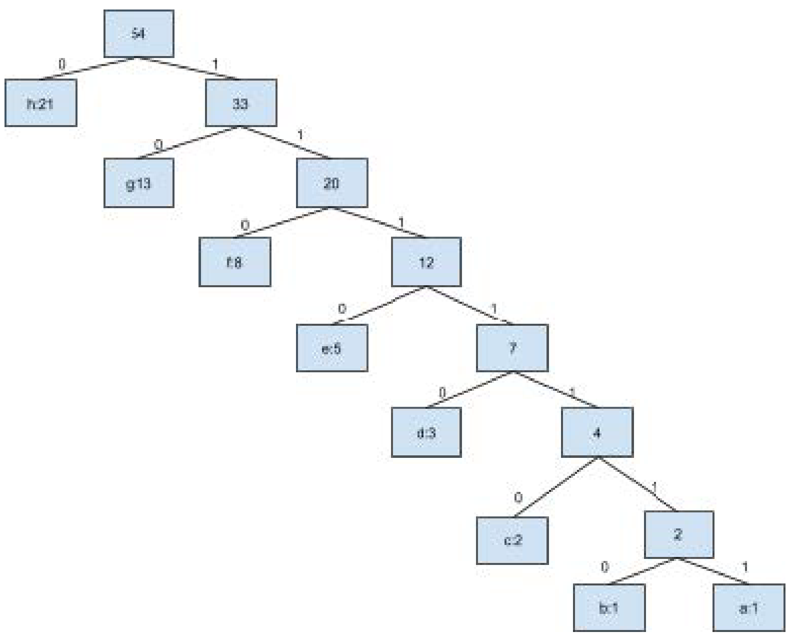
coalesce the next two smallest nodes



coalesce the next two smallest nodes



The final tree is:



Can you generalize your answer to find the optimal code when the frequencies are the first n Fibonacci numbers?

h 0
g 10
f 110
e 1110
d 11110
c 111110
b 1111110
a 1111111

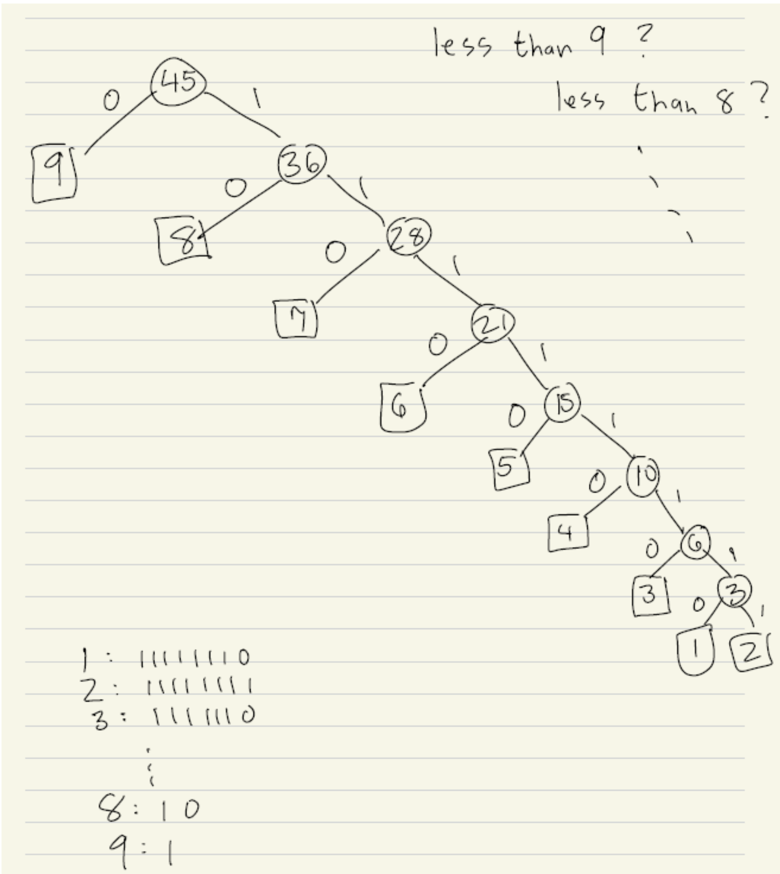
In general, if the letters are x_1 through x_n, the ***i-th*** Fibonacci number is ***F(i)***, and x_i appears with frequency ***F(i)***, then the code for letter x_i will consist of ***n - i*** 1's followed by one 0, except for the case of x_1, the least frequent letter, whose code will be ***n - 1*** 1's.

2. How many cards are there? $1 + 2 + \cdots + 9 = \frac{9(10)}{2} = 45$

What is their sum? $1^2 + 2^2 + \cdots 9^2 = \frac{9(9+1)(2\cdot 9+1)}{6}$=285

To decide which question to ask, you'd build a Huffman tree containing the numbers. Then you'd ask: "Is the chosen less than 9?" If it is, move down to the left subtree and repeat. Otherwise, move to the right subtree and repeat.

1:1 2:2 3:3 4:4 5:5 6:6 7:7 8:8 9:9

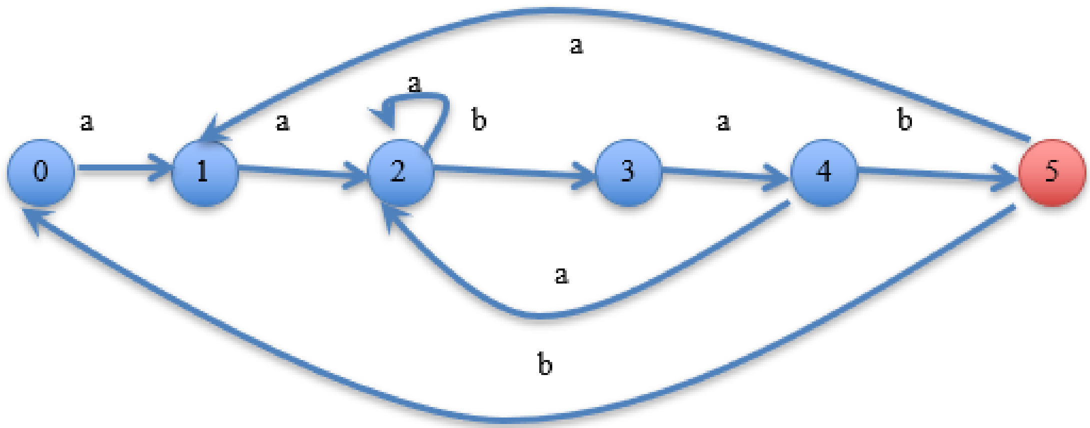


3. Determine a ***Longest Common Subsequence*** (LCS) of $< 1, 0, 0, 1 >$ and $< 0, 1, 0, 1 >$.

	j	0	1	2	3	4
i		y_j	0	1	0	1
0	x_i	0	0	0	0	0
1	1	0	↑ 0	↖ 1	← 1	↖ 1
2	0	0	↖ 1	↑ 1	↖ 2	← 2
3	0	0	↖ 1	↑ 1	↖ 2	↑ 2
4	1	0	↑ 1	↖ 2	↑ 2	↖ 3

The LCS is 101, which can be found by starting lower right and using the character corresponding to any cell with a diagonal arrow.

4. Construct the (1) string matching automaton and (2) the state-transition table for the pattern $P = aabab$ over the alphabet $\{a, b\}$.



$\sigma(P_ja)$ = "length of the longest prefix of P which is a suffix of P_ja"

δ(state , input char) →
ex: $\delta(1,a) = \sigma(P_1a) = \sigma(aa)$ = "length of the longest prefix of P which is a suffix of 'aa' " = 2

	Input		
State	***a***	***b***	***P***
0	$\delta(0,a) = \sigma(P_0a) = \sigma(a) = \boldsymbol{1}$	$\delta(0,b) = \sigma(P_0b) = \sigma(b) = \boldsymbol{0}$	***a***
1	$\delta(1,a) = \sigma(P_1a) = \sigma(aa) = \boldsymbol{2}$	$\delta(1,b) = \sigma(P_1b) = \sigma(ab) = \boldsymbol{0}$	***a***
2	$\delta(2,a) = \sigma(P_2a) = \sigma(aaa) = \boldsymbol{2}$	$\delta(2,b) = \sigma(P_2b) = \sigma(aab) = \boldsymbol{3}$	***b***
3	$\delta(3,a) = \sigma(P_3a) = \sigma(aaba) = \boldsymbol{4}$	$\delta(3,b) = \sigma(P_3b) = \sigma(aabb) = \boldsymbol{0}$	***a***
4	$\delta(4,a) = \sigma(P_4a) = \sigma(aabaa) = \boldsymbol{2}$	$\delta(4,b) = \sigma(P_4b) = \sigma(aabab) = \boldsymbol{5}$	***b***
5	$\delta(5,a) = \sigma(P_5a) = \sigma(aababa) = \boldsymbol{1}$	$\delta(5,b) = \sigma(P_5b) = \sigma(aababb) = \boldsymbol{0}$	

Notes: ***w [x → w is a prefix of x*** ***w] x → w is a suffix of x***