Fidele Donadje

CS260-002

1.

Fib_normal

```
def fib_normal(n):
   if n==0:
      return 0;
   elif n==1:
      return 1;
   return (fib_normal(n-1)+fib_normal(n-2))
```

Fib_memo

```
def fib_memo(n):
  if not n in array_fib:
    array_fib[n] = fib(n-1) + fib(n-2)
  return array_fib[n]
```

The complexity for fib_normal will be O(2^n)because fib_normal(n) = fib_normal (n-1) + fib_normal (n-2)

fib_normal(n-1) = fib_normal (n-2) + fib_normal (n-3)

fib_normal(n-2) = fib_normal (n-3) + fib_normal (n-4)

We will get fib_normal = 2*2*2…*2 =2^n

The complexity for fib_memo will be O(n) because calculating Fibonacci numbers now will be just looking up values in a dictionary. The worst case scenario when looking up will be O(n)

2. I could just append to the dictionary. The running time will be the same since appending is a constant time operation

Fidele Dinadje
CS260

3.1

a- D, M, N, F, J, K, L

b- A

c- A

d- F, G, H

e- B

f- J

g- F, G, H

h- Left: D, E, F   Right: H

i- The depth of C is 1

j- The height of C is 2

3.2

1- A-B E-I

2- B E I M

3- B E I N

4- A-C-H-L

5- A-C-G-J

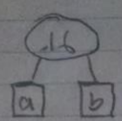
6- A-C-G-K

Pre: A B D E I M N C F G J K H L
In: D B M I N E A F C J G K L H
Post: D M N I E B F J K G L H C A

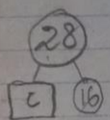
	preorder(n) < preorder(m)	inorder(n) < inorder(m)	postorder(n) < postorder(m)
n is to the left of m	✓	✓	✓
n is to the right of m	X	X	X
n is a proper ancestor of m	✓	✓	X
n is a proper descendant of m	X	X	✓

left 0 , right 1

a	b	c	d	e	f
.07	.09	.12	.22	.23	.27



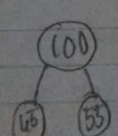
c	16	d	e	f
.12		.22	.23	.27

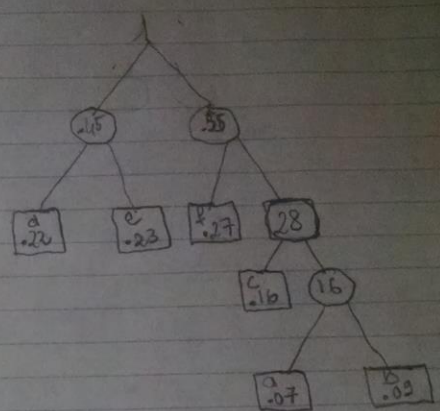


d	e	f	28
.22	.23	.27	

f	28	45
.27		

45 55



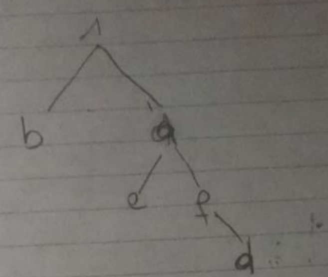


a	1110	4
b	1111	4
c	110	3
d	00	2
e	01	2
f	10	2

$$Avg = \frac{17}{6} = 2.83 \simeq 3$$

3.21

Prove that : $P(b) \geq P(a)$



If a has a greater depth than b then ~~the~~ $P(b) \geq P(a)$ because the greater the depth, the lower the probability is

Therefore $P(b) \geq P(a)$