Steven Kravitsky
CS260 - Homework 3

Code without memoization

```
import sys

def fib(x):
	if x < 2:
		return 1
	else:
		return fib(x-1) + fib(x-2)

def main():
	x = int(sys.argv[1])
	y = fib(x)
	print y

if __name__ == '__main__':
	main()
```

Code with memorization

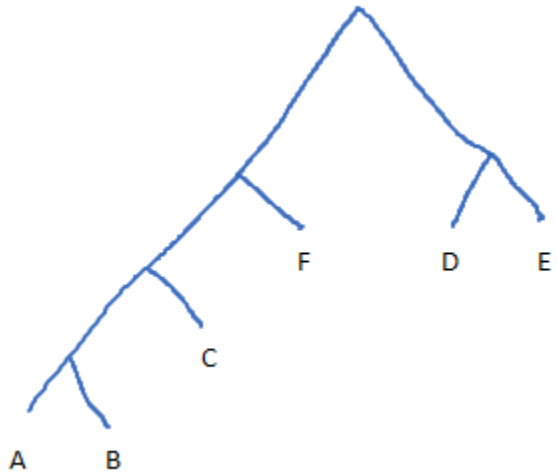
```
import sys

memo = {0:1, 1:1}
def fib_memo(x):
	if x not in memo:
		memo[x] = fib_memo(x-1) + fib_memo(x-2)
		return memo[x]
	return memo[x]

def main():
	x = int(sys.argv[1])
	y = fib_memo(x)
	print y
	memo = {0:1, 1:1}

if __name__ == '__main__':
	main()
```

1) The code for the Fibonacci calculation is O(2^n) since each calculation requires recursion with twice. The code with memorization will be O(1) since it is accessing calculated results, but it will still need to run the Fibonacci calculation if the number had not been previously computed.
2) The running time would stay relatively the same for smaller array sizes, as the array size increases then there could be a point where the array is larger than the available memory.
3) Tree question
    a. D, M, N, J, K, L
    b. A
    c. A
    d. F, G, H
    e. B, A
    f. I, M, N
    g. E is the right sibling of D, E does not have a right sibling
    h. Left: C, A, B, D, E, I, M, N, F, J Right: K, H, L
    i. 1
    j. 1
4) 6: ABEI, ACGJ, ACGK, ACHL, BEIM, BEIN
5)

	Preorder(n) < Preorder(m)	Inorder(n) < Inorder(m)	Postorder(n) < Postorder(m)
N is to the left of M	**Yes**	**Yes**	**Yes**
N is to the right of M			
N is a proper ancestor of M	**Yes**	**Yes**	
N is a proper descendant of M			**Yes**

6) Huffman Code

Letter						Code	
A	**.07**					0000	
B	**.09**	**.16**				0001	
C	.12	**.12**	.28	**.28**		001	
D	.22	.22	**.22**			01	
E	.23	.23	**.23**	.45	**.45**	10	
F	.27	.27	.27	**.27**	**.55**	11	



(4 *.07) + (4 * .09) + (3 * .12) + (2 * .22) + (2 * .23) + (2 * .27) = 2.44 Avg. Code Length

7) The depth of a node on a Huffman tree is directly correlated to the probability of that node compared to other nodes on the tree. On a Huffman tree if you are a sibling then your probability compared to your other sibling depends on your position. If you're on the left then you have a lower probability and vice versa. The nodes above you must have a higher probability than either you or your sibling or else they would have been chosen for either of your spots. This proves that if a node has a smaller depth than you then it must have a higher probability than you