6.00 Quiz 1, 2011

Name _____

1. 12 /15
2. 3 /10
3. 5 /10
4. 12 /18
5. 7 /8
6. 4 /13
7. 13 /15
8. 6 /9
9. 1 /1
10. 1 /1

Total 64/100

This quiz is open book and open notes, but do not use a computer.

Please **write your name on the top of each page**. Answer all questions in the boxes provided.

1) Are each of the following True or False (15 points)

F 1.1. In Python the **values** of a dict must be immutable.

T 1.2. There exist problems that **cannot** be solved in Python **without** using either iteration or recursion.

F 1.3. Floating point arithmetic behaves exactly like normal arithmetic on real numbers.

F 1.4. On all inputs, a bisection search will run faster than a linear search.

T 1.5. Let `L` be a list, each element of which is a list of ints. In Python, the assignment statement `L[0][0] = 3` mutates the list `L`.

FALSE, MUTATES L[0] not L

Name

2) What does the following code print? (10 points)

```
T = (0.1, 0.1)
x = 0.0
for i in range(len(T)):
    for j in T:
        x += i + j
        print x
print i
```

0 1
x += 0.2 x += 1.1
0.2 x += 1.1
+= 2.2
+ .2
2.4

<0.1
0.2
2.4 < 1.3
1

Name

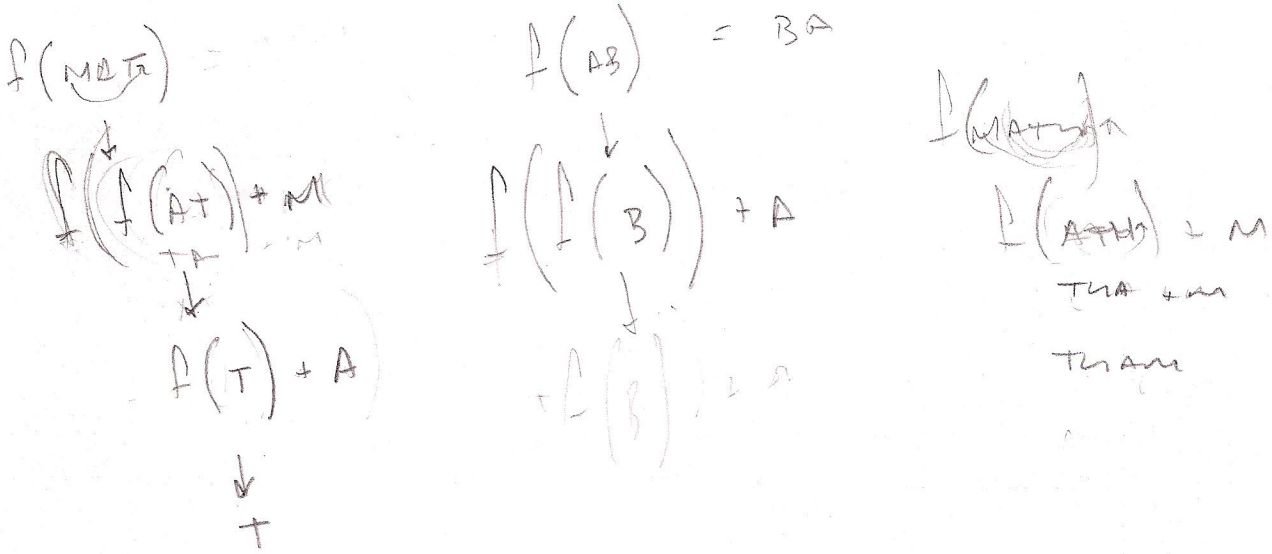
3) What does the following code print? (10 points)

```
def f(s):
    if len(s) <= 1:
        return s
    return f(f(s[1:])) + s[0]   #Note double recursion

print f('mat')
print f('math')
```

atm

tham



Name _____

4) Implement the body of the function specified in the box. (18 points)

```
def findAll(wordList, lStr):
    """assumes: wordList is a list of words in lowercase.
                lStr is a str of lowercase letters.
                No letter occurs in lStr more than once
       returns: a list of all the words in wordList that contain
                each of the letters in lStr exactly once and no
                letters not in lStr."""
```

```
out = []
for word in wordlist:
    if len(word) == len(lStr) and all_letters_used(word, lStr):
        out.append(word)
return out
```

sorted(lStr) == sorted(word)

```
def all_letters_used(word, letters)
    for char in letters:
        if char not in word:
            return False
    return True
```

Name _____

5) The following code does not meet its specification. Correct it. (8 points)

```
def addVectors(v1, v2):
    """assumes v1 and v2 are lists of ints.
       Returns a list containing the pointwise sum of
       the elements in v1 and v2.  For example,
       addVectors([4,5], [1,2,3]) returns [5,7,3],and
       addVectors([], []) returns []. Does not modify inputs."""
    if len(v1) > len(v2):
        result = v1[:]
        other = v2
    else:
        result = v2[:]
        other = v1
    for i in range(len(other)):
        result[i] += other[i]
    return result
```

```
def addVectors(v1, v2):
    v1_len = len(v1)
    v2_len = len(v2)
    if v1_len == 0:
        return v2[:]
    elif v2_len == 0:
        return v1[:]
    else:
        if v1_len > v2_len:
            result = v1[:]
            other = v2
        else:
            result = v2[:]
            other = v1
        for i in range(len(other)):
            result[i] += other[i]
        return result
```

NOT NECESSARY

Name _____

6) Consider the following code:

```
def f(s, d):
    for k in d.keys():
        d[k] = 0
    for c in s:
        if c in d:
            d[c] += 1
        else: d[c] = 0
    return d

def addUp(d):
    result = 0
    for k in d:
        result += d[k]
    return result

d1 = {}
d2 = d1
d1 = f('abbc', d1)
print addUp(d1)
d2 = f('bbcaa', d2)
print addUp(d2)
print f('', {})
print result
```

sets all values of keys in d to 0

if item in c is a key in d, increment by 1
otherwise add the key and set to 0

sums values in d

d1 and d2 both alias empty dict

{'a': 0, 'b': 1, 'c': 0} = d1 + d2

{'a': 2, 'b': 3, 'c': 1}

6.1) What does it print? (9 points)

1
6
{}
None

6.2) Does it terminate normally? Why or why not? (4 points)

Yes, Both functions loop on finite inputs
len(s) in f()
len(d.keys()) in addUp() } Decrementing functions

Name

7) Consider the following code:

```
def logBase2(n):
    """assumes that n is a positive int
       returns a float that approximates the log base 2 of n"""
    import math
    return math.log(n, 2)

def f(n):
    """assumes n is an int"""
    if n < 1:
        return
    curDigit = int(logBase2(n))
    ans = 'n = '
    while curDigit >= 0:
        if n%(2**curDigit) < n:
            ans = ans + '1'
            n = n - 2**curDigit
        else:
            ans = ans + '0'
        curDigit -= 1
    return ans

for i in range(3):
    print f(i)
```

7.1) What does it print? (10 points)

Name
ans = 1
ans = 10
should be 'n'

7.2) Under the assumption that `logBase2` is `O(n)`, what is the order (use big Oh notation) of `f`? (5 points)

O(n)

Name _____

8) Next to each item in the left column write the letter labeling the item in the right column that best matches the item in the left column. No item in the right column should be used more than once. (9 points)

[E]	Big O notation	a) induction
[D]	Newton's method	b) upper bound
[A]	recursion	c) lower bound
		d) approximation
		e) expected running time
		f) exponential

9. Do you think that the lectures are too slow paced, too fast paced, about right? (1 point)

10. Do you think that the problem sets are too easy, too hard, about right? (1 point)