You are allowed 1 piece of paper (both sides) for this quiz. You may not use any computing or communication device. **You have 90 minutes.**

Please **write your name on the top of each page**, and your user name and the hour of the recitation you attend on the first page. Answer all questions in the boxes provided.
1) [11 points] Are each of the following True or False?

   a) The Python expression `'3' * 5` is syntactically correct. [T]
   b) A Python dictionary can have another Python dictionary as a key. [F]
   c) The first element of a Python list named `L` is `L[0]`. [T]
   d) Every function must have a `return` statement. [F]
   e) Suppose `flag` is a Boolean variable. These two conditionals are equivalent:
      (a) `if flag==True:`          (b) `if flag:`
               [T]
   f) The bisection method always divides the interval approximately in half. [T]
   g) A variable defined inside a function can be accessed outside the function. [T/F]
   h) When implementing a function, checking that the assumptions in the specification hold is rarely a good idea. [F]
   i) A recursive function in Python without a `return` statement will never terminate. [F]
   j) A Python list is mutable. [T]
   k) The last element of a Python list named `L` is `L[len(L)]`. [F]
2) [15 points] Consider the following code.

```python
def mystery_function(s):
    def helper(s):
        result = ''
        for i in range(len(s)):
            result = result + s[len(s)-i-1]
        return result
    mystery_var1 = len(s)
    mystery_var2 = helper(s)
    if mystery_var1 == 0:
        return "Nothing there!"
    elif s == mystery_var2:
        return "Congratulations!"
    else:
        return "Nothing special :("

Suppose we call the function in the following ways. What will be printed? If nothing is printed, write down “blank”. If there is an error or exception, write “error” and continue executing the statements after.

a) mystery_function('')
   blank

b) print mystery_function('A')
    print mystery_var1
    Congratulations! 
    error

c) print mystery_function('Congratulations!')
   Nothing special :(

d) print mystery_function('Desserts, I stressed!')
   Nothing special :(

e) s = 'yo bananab oy'
   print mystery_function(s)
   Congratulations!
3) [14 points] Consider the following transcript of Python commands. For each question, assume that the code from previous questions has already been run. Write what is printed in the boxes provided. If there is an error or exception, write ‘error’ and continue executing the statements after. If nothing is printed, write ‘blank’.

```python
fruits = ['apple', 'banana']
desserts = ['pie', 'split']
L = len(fruits)
combo = fruits + desserts
print combo
['apple', 'banana', 'pie', 'split']

for f in range(L):
    combo[f] = fruits[f] + desserts[len(desserts)-1-f]
print combo
['applesplit', 'bananapie', 'pie', 'split']

meal = [fruits]
print meal
[(['apple', 'banana'])]

veggies = fruits
veggies[1] = 'tomato'
print veggies
['apple', 'tomato']

veggies = fruits[:]
veggies[0] = 'melon'
print veggies
['apple', 'tomato']

veggies = ['broccoli', 'spinach']
meal.append(veggies)
print meal
[(['apple', 'tomato'], ['broccoli', 'spinach'])]

meal[0][1][0] = 'p'
print meal
error
[(['apple', 'tomato'], ['broccoli', 'spinach'])]
```
4) [16 points] What does the following code print? If there is an error or exception, write ‘error’ and continue executing the statements after.

```python
def build_dict(words):
    word_dict = {}
    for w in words:
        key_list = []
        for c in w:
            key_list.append(c)
        key_list.sort()
        key = ''
        for e in key_list:
            key = key + e
        if key not in word_dict:
            word_dict[key] = [w]
        elif w not in word_dict[key]:
            word_dict[key].append(w)
    return word_dict

words = ['dog', 'good', 'god', 'nan', 'ann', 'god']
new_dict = build_dict(words)
keys = new_dict.keys()
keys.sort()
for k in keys:
    print new_dict[k]
L = new_dict['ann']
L.insert(1, 'dog')
print new_dict['ann']
print new_dict[0]
```

['nan', 'ann']
['dog', 'god']
['good']
['nan', 'dog', 'ann']
error
5) [15 points] Write a Python function that satisfies the specification below. For example, if \( n = 3 \), the function prints out:

```
***
***
***
```

```python
def star_grid(n):
    '''
    n: an int
    This function prints "not an integer" if n is not an integer
    Otherwise, it prints an n by n grid of the * character
    '''
    if type(n) != int:
        print "not an integer"
    else:
        s = ''
        for i in range(n):
            s += "*"
        for i in range(n):
            print s
```

--------- OR ---------

```python
if type(n) != int:
    print "not an integer"
else:
    for i in range(n):
        print n * '*'
```
6) [15 points] Write a recursive function in Python that implements the formula:

\[
\begin{align*}
    f(n) &= 2 & \text{if } n = 1 \\
    f(n) &= (f(n-1))^2 + 2 & \text{otherwise}
\end{align*}
\]

Assume that \( n \geq 1 \). The first few terms of this function are: 2, 6, 38, 1446…

```python
def f(n):
    if n == 1:
        return 2
    else:
        return f(n-1)**2 + 2
```
7) [12 points]
   a) What is wrong with the following snippet of Python code?

   ```python
   i = raw_input("What course is this? ")
   if i == 6.0001:
       print "Correct!"
   ```

   Not casting `raw_input` to float.
   Comparing `raw_input` (a string) to a float.
   Forgot to put “6.0001”.

   b) Does this code calculate \((\frac{1}{n})^p\) correctly? Assume \(p\) is a positive integer. Explain why or why not.

   ```python
   def inverse_power(n, p):
       while p > 0:
           n = 1.0/n
           p -= 1
       return n
   ```

   No. The variable that is supposed to calculate that quantity just flip flops between \(n\) and \(1/n\). We have to assign \(1/n\) to a different variable, not \(n\).
c) Consider the following two snippets of Python code. Does each snippet print the string ‘M_I_T’? Explain why or why not for each snippet.

# SNIPPET 1
s = "M I T"
for c in s:
    if c == " ":
        c = "_"
print s

# SNIPPET 2
s = "M I T"
s[1] = '_'
s[3] = '_'
print s
8) [2 points]

a) Are problem sets (circle one):
   
   Too time consuming
   
   Too short
   
   Just right

b) Is the lecture pace (circle one):
   
   Too fast
   
   Too slow
   
   Just right