

CS 150 Spring 2014
PRACTICE Final

Name: _____

This test is closed notes, closed book. No aids are permitted. The test has 9 pages including this page. There are 8 questions. You have 2 hours. You may use the back of any page if you need more space; indicate on the front of that page if you do so.

Good luck!

Question 1	/15 pts
Question 2	/10 pts
Question 3	/5 pts
Question 4	/15 pts
Question 5	/15 pts
Question 6	/10 pts
Question 7	/15 pts
Question 8	/15 pts
Total	/100 pts

1. (15 pts) **Recursion.**

(a) What is the output of the following chunk of code?

```
def R(s) :  
    if len(s) == 0 :  
        return ""  
    else :  
        return s[0] + R(s[1:]) + s[0]  
  
print(R(cat))
```

(b) Write a recursive function `lessThan(L,e)` that takes a list `L` and an element `e`, and returns `True` if all of the elements of `L` are less than `e`, and `False` otherwise. You cannot make any assumptions about what type of objects `e` and the elements of `L` are, but you can assume that `<`, `>`, `=` etc work. Do not use loops.

2. (10 pts) **Odds and Ends.**

(a) What is wrong with the following code?

```
def __init__(a,b,c):  
    a = self.a  
    b = self.b  
    c = self.c
```

(b) Suppose A , B and C are boolean variables. Write a boolean expression that evaluates to true if and only if one or more of these variables are False.

3. (5 pts) **Runtime Evaluation.**

(a) Consider the following stupid function that takes in a list of integers:

```
def s(L) :  
    x = 0  
    for i in range(len(L)) :  
        for j in range(50) :  
            L[i] = L[i] + j  
    for i in range(len(L)) :  
        print(L[i])
```

What is the runtime of this function in terms of n , where n is the length of the list L . Express your answer using Big O notation.

4. (15 pts) **Searching and Sorting.**

- (a) Which has the best Big O runtime: Selection sort, Insertion sort, or Bubble sort?
 - (b) Will selection sort or insertion sort perform better on a list that is already partially sorted?
 - (c) Write pseudocode to do a binary search over a sorted list.
-

5. (15 pts) **Multiple Processes (a)** Assume you have a very large number, n , and a list, P , that consists of the primes from 2 to the first prime larger than the square root of n . Use a process pool with 4 processes to determine if n is prime.

(b) Write code where 10 processes each add 1 to `r.value` 10 times, ending with `r.value` equal to 100. Make sure you use the lock to ensure you end up with the correct result.

```
r = RawValue("i",0)
l = Lock()
```

6. (10 pts) **Objects and Inheritance.** Create a subclass of S, R, which says "Wheee!" instead of "Hi", and eats pizza instead of kale. Its sleep should remain the same.

```
class S:
    def talk(self):
        return "Hi!"

    def nom(self):
        return "Delicious kale!"

    def sleep(self):
        return "zzzzzz"
```

7. (15 pts) **Patterns.** Write a function `tree(n)`, which takes in a number and draws a lovely tree like the ones pictured below.

`tree(3)`

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

`tree(5)`

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


8. (15 pts) **2D Lists.** Write a Python function called `diagFlip(L)` that takes in a 2-dimensional $n \times n$ list L and flips it around the diagonal. For example, if the list L was

```
1  2  3  4
5  6  7  8
9  10 11 12
13 14 15 16
```

then $n = 4$ and the function should update the list to

```
1  5  9  13
2  6  10 14
3  7  11 15
4  8  12 16
```

If you're having trouble getting started, try to figure out where the value at a general location (i, j) should be placed in terms of i , j and n . Sometimes its easier to think about bigger examples. Check whether your answer is consistent with the above example. Hint: it may be useful to create a new copy of L to avoid accidentally overwriting data.