

Name: _____ Section: ____ Andrew Id: _____

15-112 Fall 2014 Quiz 8

- * 20 minutes. No calculators, no notes, no books, no computers.
- * You may not discuss any portion of this quiz with anyone until after 5pm today.
- * SHOW YOUR WORK, CIRCLE YOUR ANSWERS.

1. **Code Tracing** [30 pts]: Indicate what this will print:

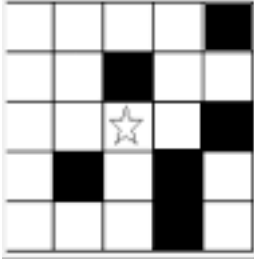
```
def f(x, y=42):
    print (x, y), # don't miss this!
    if (x > y): return (x+y, x-y)
    else: return f(x*2, y/2)
print f(6)
```

```
def g(i, t=""):
    print (i,t), # don't miss this!
    if (i == 0): return t
    else: return chr(ord('A')+i) + g(i/2, t+str(i))
print g(3)
```

```
def h(x, depth=0):
    # This uses depth-based indentation as in the course notes
    print "  *depth, "h(%d)" % x
    if (x < 3): result = x
    else: result = 2*h(x/2, depth+1) + 3*h(x/3, depth+1)
    print "  *depth, "-->", result
    return result
print h(8)
```

2. FloodFill Tracing [20 pts]

Say we start with a 5x5 board and fill the black squares as shown below (in the code from our notes, the black squares would be green, and the white squares would be cyan). Then we right-click where the star is to start a floodFill from there. Write the numeric labels that result in each cell after the floodFill completes.



Hint #1: the numbers are the *depth* at each cell, so some labels may occur more than once.

Hint #2: the first label is 0, not 1 (so a 0 should be where the star is).

Hint #3: Here is an excerpt of the floodFill code:

```
def floodFill(row, col, color, depth=0):  
    ...  
    floodFill(row-1, col, color, depth+1)  
    floodFill(row+1, col, color, depth+1)  
    floodFill(row, col-1, color, depth+1)  
    floodFill(row, col+1, color, depth+1)
```

3. Reasoning Over Code [15 pts]

Find arguments for the following function that make it return True. You only need one set of arguments for the function, even if there are multiple correct answers.

```
def r(a):  
    def f(a):  
        if (a == [ ]):  
            return [ ]  
        else:  
            first = [a[0]]  
            rest = f(a[1:])  
            if (len(a) % 2 == 0): return first + rest  
            else: return rest + first  
    return f(a) == range(5,9) # hint: what is f(range(4))?
```

4. **Free Response: largest(a)** [35 pts]

Write the function `largest(a)` that takes a list of ints but also lists of other such lists, and returns the largest int at any level. So: `largest([1,[2,3],[2,[4,[3,5],4,2]],1])` returns 5. You may assume that every list is non-empty. Note that you may use "for" or "while" here, so long as you also use recursion in some meaningful way.

5. **Bonus/Optional:** [6 pts] Indicate what this will print:

```
import copy
def bonus(d,i=1):
    if (i not in d): return 42
    for key in d:
        d[1].append((".%d"%i) if (type(d[key]) == dict) else str(key))
    return i + 10*bonus(d, i+1)
d = {1:[ ], 3:4}
d[len(d)] = d
print bonus(d), "".join(d[1]) # 6 pts (3 pts each)
```

This page is intentionally blank.