15-112: Practice Final Exam Questions

Angela Zhang (afzhang)
Recitation I

1. Fill in the following function so it solves the Towers of Hanoi. Specifically, towersOfHanoi should return a list of (from, to) pairs such that making those moves in order would move n disks rom post 0 to post 1.

```
assert(towersOfHanoi(1) == [(0,1)])
assert(towersOfHanoi(2) == [(0,2), (0,1), (2,1)])
assert(towersOfHanoi(4) == [(0, 2), (0, 1), (2, 1), (0, 2), (1, 0),
                            (1, 2), (0, 2), (0, 1), (2, 1), (2, 0),
                            (1, 0), (2, 1), (0, 2), (0, 1), (2, 1)])

def towersOfHanoi(n, frm = 0, to = 1, via = 2):
```

2. Write a function that finds the largest integer out of its arguments, which may or may not be well-formed.

```
assert(findLargest(1,2,3,4) == 4)
assert(findLargest("hello", 5, "look a string :o", 42, "#112forlife") == 42)
assert(findLargest([1,2,3,4], 2, 3) == 4)
assert(findLargest("sup", (2,3), "yes that was a tuple") == 3)

def findSmallest(*args):
```

3. Write a function that satisfies the following. (Hint: optional parameters!)

```
assert(f("abc") == "abc")
assert(f("abc", True) == "ABC")
```

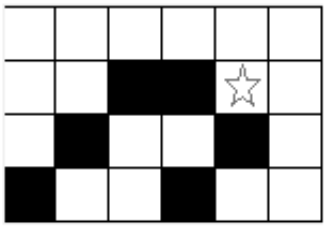
4. A number is a "palindromic wing" if it is of the form xxxxxYxxxxx, so the x's form the "wings" and the Y is the "body" (and where x and Y are single (non-negative) digits, and the two wings have the same non-zero length). With this in mind, write the function isPalindromicWing(n), that takes an integer n and returns True if it is a palindromic wing and False otherwise.

```
assert(isPalindromicWing(2223222) == True)
assert(isPalindromicWing(2232322) == False)
assert(isPalindromicWing(2233322) == False)
assert(isPalindromicWing(22) == False) # too small
assert(isPalindromicWing(2222) == False) # must have odd # of digits (right?)

def isPalindromicWingPrime(n):
```

5. If a Level 0 tree fractal is a straight vertical line, and a Level 1 tree fractal is a Y, draw a Level 2 tree fractal.

6. Say we start with a 4x6 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: 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)
```

7. Here is the powerset code from the class notes, with parts of two lines removed:

```
def powerset(a):
    # returns a list of all subsets of the list a
    if (len(a) == 0):
        return [[]]
    else:
        allSubsets = [ ]
        for subset in ____________________________________:
            allSubsets += [subset]
            allSubsets += ____________________________________________
        return allSubsets
```

Fill in the blanks with the missing code.

8. Find a value of x such that ((~x)**2 - x**2) equals 101 (decimal).

9. For full credit on this problem, you may not use loops or conditionals. If you cannot solve it that way, then for partial credit you may use loops or conditionals.

Write the function halfBlue(rgba) that takes a 32-bit rgba value (where each of the red, green, blue, and alpha parts are 8 bits), and returns another rgba value where the red, green, and alpha are unchanged, but with half as much blue as the original color.

```
def halfBlue(rgba):
```

10. Write a Tkinter program that draws an American flag.

11. Reasoning About Recursive Code: In a few words of English, describe what each of the following functions does *in general*. Do not describe the line-by-line low-level behavior of the code.

```
def f(n):
   # assume n is a non-negative integer
   if (n < 10):
      return 1
   else:
      return 1 + f(n/10)

def g(a):
   # assume a is a list of integers
   if (len(a) == 0):
      return 0
   elif (len(a) == 1):
      return (a[0] % 2)
   else:
      i = len(a)/2
      return g(a[:i]) + g(a[i:])

def h(n):
   # assume n is a non-negative integer
   if (n == 0):
      return 0
   else:
      return h(n-1) + 2*n - 1
```

12. Fill in the following function using string formatting such that these text cases will return True.

```
assert(fmt(4) == "+4")
assert(fmt(-4) == "-4")

def fmt(n):
   return "____________" %n
```

13. What will the following print? `0x15112CA & 0xbeef`

14. Fill in the rest of Polyas 4 steps for problem solving:

    1. _____
    2. _____
    3. Carry out the plan.
    4. _____

15. You have a five-sided die whose sides are: “Diem”, “sin(b)/tan(b)”, “Carpe”, “15-112”, “!”. Write a function that uses Monte Carlo methods to return the probability of rolling the die three times and getting the concatenated result “Carpe Diem!”

```
def carpeMC(trials, sides = [‘Diem’, ‘sin(b)/tan(b)’, ‘Carpe’, ‘15-112’, ‘!’]):
```

16. What is an ASCII value?

17. Write a function, findSubstring(s, substr) that takes in a string s and a substring substr that may or may not occur in s. If substr occurs in s return a tuple (start, end) representing the index where the substring starts (inclusive) and ends (exclusive). If the substring does not occur return None. If the substring occurs multiple times you only need to return the first set of indices where it is found. Do not use any built-in `string` methods.

```
def findSubstring(s, substr):
```

18. What will the following print when called with `a = [42,6,0,4,1,5,0,4,1,5]`?

```
def f(a):
  s = set()
  d = dict()
  for i in range(len(a)):
    if (i%2 == 0):
     if (a[i] in d):
            s.add(a[i+1])
        else:
            d[a[i]] = a[i+1]
  return (sorted(s), d[42])
```

19. Add a class variable and method to the following class such that the assert passes.

```
bovik = Student("Harry Q. Bovik", 71)
burdell = Student("George P. Burdell", 70)

assert(student.getStudents == 2)

class Student(object):
  def __init__(self, name, age):
    self.name = name
    self.age = age
```

20. True or False:

    a. For any positive integers `x` and `y`, if `((x&y) == (x|y))` is True, then `(x^y == 0)` is True.

    b. For a 1d List L that only contains int values, copy.copy(L) is effectively the same as copy.deepcopy(L).

    c. If f(N) runs in O(N**2) time and g(N) runs in O(N**3) time, then f(N) is always faster than g(N) for all positive integers N.

    d. Lambda expressions may not contain additional lambda expressions.

    e. Anything computed with recursion can also be computed without recursion.

    f. A set can be a key in a dictionary, but a dictionary cannot be added to a set.

    g. If we changed floodFill so that sometimes it recursed in the order up/down/left/right, and other times it recursed in the order up/left/right/down, it would sometimes fail to completely floodFill some regions.

21. What is memoization?

22. List two problems that are most easily solved using backtracking.

23. What was the main reason that we used a closure in our animation run function?

If you need more to do, check out:

http://www.kosbie.net/cmu/spring-12/15-112/handouts/final-exam-practice.html

http://www.kosbie.net/cmu/fall-11/15-112/handouts/notes-final-exam-practice.html