15-112 Summer 1 Practice Quiz 2

Code Tracing

```
def ct1(s,  n):
    result  =   ""
    d   =   0
    while   (n  >   0)  and (len(s) >   0):
        if  (s[-1].isdigit()):
            result +=  str((n%10)%int(s[-1]))
        else:
            result  +=  chr(ord('D')    +   d)
        n   //= 10
        s   =   s[1:-1]
        d   +=  1
    return  result
print(ct1("abc3c3", 2468))
```

```
def ct2(a):
    a = [42]
a = [1, 2, 3]
b = ct2(a)
print(a)
print(b)
```

Reasoning Over Code

```
def   rc1(s,t):
      assert((s != "" and t != "") and (s in t))
      result =    ""
      for i in range(len(s)):
            If (i % 2) == 0:
                  result += t[i]
            else:
                  result+= t[-1-i]
      return(result == s)
```

```
def rc2(M):
    assert((type(M) == list) and (len(M) == 5))
    for i in range(-1,  3):
        assert(M[i] == M[i-1] + i)
    return (sum(M) == 15)
```

Free Response

nondestructiveRotateList(a, n)

Write the function nondestructiveRotateList(a, n) which takes a list a and an integer n, and nondestructively modifies the list so that each element is shifted to the right by n indices (including wraparound). The function should then return this new list. For example:

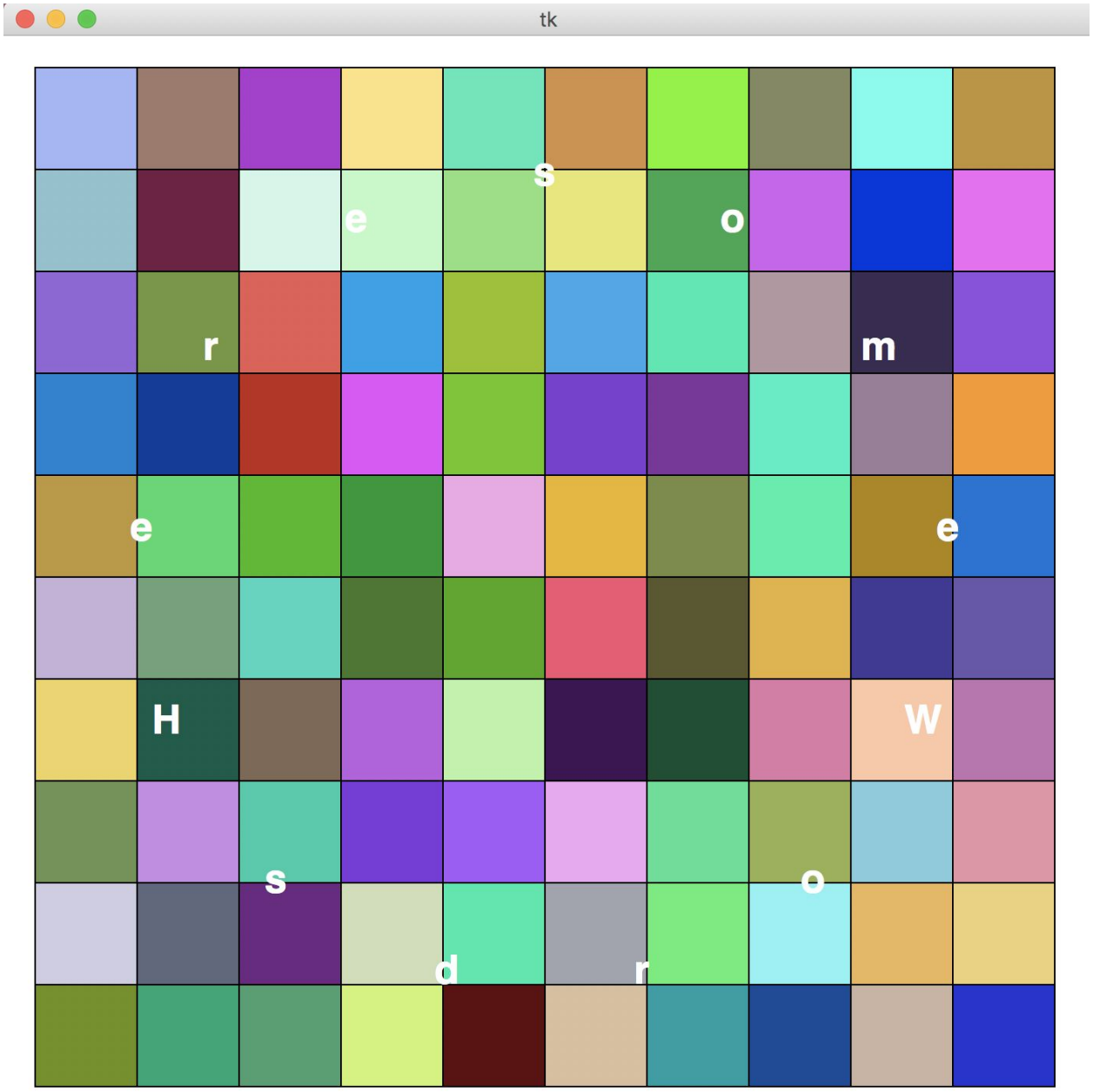
```
nondestructiveRotateList([1,2,3,4], 1) -> [4,1,2,3]
nondestructiveRotateList([4,3,2,6,5], 2) -> [6, 5, 4, 3, 2]
nondestructiveRotateList([1,2,3], 0) -> [1,2,3]
nondestructiveRotateList([1, 2, 3], -1) -> [2, 3, 1]
```

```
Def nonDestructiveRotateList(a, n):
    b = [0]*len(a)
    For i in range(len(a)):
        B[i] = a[(i+n)%len(a)]
    Return b
```

drawColorfullyBackgroundedCircleOfText()

For this problem, assume the runDrawing function is already defined. Create the function drawColorfullyBackgroundedCircleOfText(canvas, width, height, inputString, rows, cols, margin) as follows. We want the inputString to be placed as if making a circle centered on the screen, with each letter equidistant from its neighbors (with no actual circle being drawn). Additionally, make the background a grid with as many rows and columns as specified by the arguments, and bordered on each side (top, bottom, left, and right) by exactly the dimension specified by the ‘margin’ input. Finally, using the rgbString(red, green, blue) function (as defined in the notes) and the function random.randint(lower, upper) (which returns a random integer between upper and lower, including both the bounds as the possible result), make the color of every cell in the background grid random.

An example:



Bonus

destructiveRotateList(a, n)

This function works the same as the previous function, only here it is destructive. That is, it directly changes the list a, so after the call, that exact list is rotated n indices to the right with wraparound, and a new list is not created. As usual for destructive functions, this function returns None. Also: you may not call the nondestructive version here, and in fact, you may not even create a new list (or tuple or other similar data structure) that is longer than 2 elements! While you must be space-efficient here, we do not expect the most time-efficient approach; anything reasonable (for 15-112) will do