Name:_____ Section:___ Andrew Id: _____ a

15-112 Fall 2019 Quiz 10a
*** Up to 20 minutes. No calculators, no notes, no books, no computers. * Show your work!**
Do not unstaple the pages!

1. Free Response: powerSum(n, k) [25 pts]
Write the function powerSum(n, k) that takes two non-negative integers, n and k, and returns the result of the following power-sum sequence: 1**k + 2**k + ... + n**k. For example, powerSum(4, 2) would return 30, because 1**2 + 2**2 + 3**2 + 4**2 = 1 + 4 + 9 + 16 == 30.

Note: You must solve this recursively or you will not receive any credit. You may not use loops or list comprehensions (or anything we haven't taught you, i.e. generators). You may only use builtin functions or methods if they run in O(1)

2. Free Response: limitedPowerSet(L, limit) [30 pts]
Write the function limitedPowerSet(L, limit), where L is a list of positive integers, and limit is a positive integer. This modified powerset function returns a list of all subsets of L where the sum of each subset is less than limit.

For example, limitedPowerSet([1, 2, 3], 5) should return [[], [1], [2], [1, 2], [3], [1, 3]]. Order of elements does not matter for this problem.

If you generate the full powerset first and then check the subset sums, you will only get partial credit.

Note: You must solve this recursively or you will not receive any credit. You **may** use loops and loop-like builtins we've shown you!

3. CT [25 pts]
Indicate what this prints in the box to the right (and nothing else):

```
def ct(s, depth = 0):
    print(f"{depth}->{s}")
    if len(s) == 1:
        result = s
    elif s[0] in "aeiou":
        result = ct(s[1:], depth+1) + s[0]
    else:
        result = s[0] + ct(s[1:], depth+1)
    print(f"{depth}<-{result}")
    return result

print(ct("one"))
```

4. RC [20 pts]
Find an argument for the function rc(L) that makes it return True. Place your answers (and nothing else) in the box beside the code:

L =

```
def f(L, n):
    if len(L) < 2:
        return True
    else:
        return ((L[0]*n == L[-1]) and
                f(L[1:-1], n+1))

def rc(L):
    for val in L:
        if (not isinstance(val, int)) or (val < 2):
            return False

    return ((len(L) == 6) and
            (len(set(L)) == 6) and
            f(L, 2) == True)
```

5. Bonus/Optional: Code Tracing [2 pts]
Indicate what this prints. Very clearly circle your answer (and nothing else):

```
def g(x):
    if (x < 5): print(f"b{x}", end = "")
    if (x < 1): return 1
    return x + g(x - 2) if (x%2 == 0) else x + g(x + 2)
def bonusCT(x):
    print(f"a{x}", end = "")
    try: return g(x + 1)
    except: return bonusCT(x + 3)
print(f" --> {bonusCT(2)}")
```