Name: ______________________________ Andrew Id: ____________________

15-112 Spring 2022 Quiz 9

Up to 25 minutes (up to 20 minutes for 20% proficiency bonus) . No calculators, no notes, no books, no computers. Show your work!
Do not use try/except on this quiz.

1. (2 points) **Short Answer**: The following function is supposed to sum up the elements in the list `L` in a recursive way. Indicate whether the function is correct or not. If not, indicate what's wrong with it. Place your answer in the box below the code. Answer in **one line**.

```
def recSumList(L):
    return L[0] + recSumList(L[1:])
```

2. (6 points) **Code Tracing**: Indicate what the following program prints. Place your answer (and nothing else) in the box next to the code.

```
def ct(s, depth=0):
    print(depth, "in:", s)
    if len(s) == 1:
        result = '43'
    elif s[0] in "aeiou":
        result = ct(s[1:], depth+1)
        result = result[::-1]
    else:
        result = s[0] + ct(s[1:], depth+1)
    print(depth, "out:", result)
    return result
ct("yaho")
```

3. (4 points) **Reasoning Over Code**: Find an argument, `n`, for the following function to cause it to return `True`. Place your answer (and nothing else) in the box below the code.

```
def rocHelper(n, d):
    if d == 1:
        return (n == 6)
    if d == 2:
        return (n == 42)
    h = d//2
    return rocHelper(n//(10**h), d - h) and rocHelper(n%(10**h), h)

def roc(n):
    return rocHelper(n, 6)
```

4. (8 points) **Free Response: Recursive mergeSorted**

   Write the recursive function `mergeSorted(list1, list2)` that, given two *sorted* lists `list1` and `list2`, merges the two lists into one sorted list. The function must return the *sorted merged* list.

   Note: You can assume `list1` and `list2` are already sorted.

   Important Notes:

   - You must not use loops – no `for` loops or `while`.
   - You may only use builtin functions if they don't iterate over something. So in particular, among many other builtins you may not use, you may not use `sort`, `sorted`, `min`, or `max`.

   Examples:

```
assert(mergeSorted([1,5,6], [3,4]) == [1,3,4,5,6])
assert(mergeSorted([2,4], [1,3,5]) == [1,2,3,4,5])
assert(mergeSorted([7,8], [1,2,3,4]) == [1,2,3,4,7,8])
assert(mergeSorted([], [1,6]) == [1,6])
assert(mergeSorted([], []) == [])
assert(mergeSorted([8], []) == [8])
```

Free Response answers: