

15-112 Fall 2021 Quiz 4

Up to 20 minutes. No calculators, no notes, no books, no computers. Show your work!
Do not use string indexing, loops, lists, dictionaries, try/except, or recursion on this quiz.

1. (4 points) **Code Tracing:** Indicate what the following program prints. Place your answer (and nothing else) in the box below the code.

```
def ct1(L):
    ret = []
    for i in range(0, len(L)-1):
        if isinstance(L[i], type(L[-1])):
            print("A:", L[i])
            ret.append(L[i])
        else:
            print("B:", L[L[i]])
            ret.insert(0, L[i])
    return ret

print(ct1([1, "purple", 4, "monkey", "dishwasher", 3, "bob"]))
```

2. (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.

Hint: Start by trying a 4 digit number like 1234, and see what values *a* and *b* hold.

```
def rc1(n):
    assert(n >= 1000 and n < 10000 and isinstance(n, int))
    a = b = count = 0
    while n > 0:
        a = 10*a + n%10
        b = 10*b + (n//10)%10
        n //= 100
    for i in range(a, b):
        if (i % 10 == 0):
            count += 1
    return a > 75 and (a % 10) * (b % 10) == 6 and count == 2
```

3. Free Response

Do not use strings, lists, dictionaries, sets, try/except, or recursion on this problem. If you do, you will receive a 0.

We'll say that an integer is a `hybrindt` number (coined term) if it is a positive integer and the most frequently occurring digit in the number is a 9.

Some examples of `hybrindt` numbers are: 9, 99, 199, 19997773, 912121999.

Some numbers that are *not* `hybrindt` are: 7 (no nines at all), 292 (more 2s than 9s), 19 (same number of 1s as 9s), 92121212999 (same number of 2s as 9s).

- (a) (8 points) Write the function `isHybrindt(n)`, which takes a positive integer `n` and returns `True` if `n` is a `hybrindt` number and `False` otherwise. You can write any additional helper functions that you need.
Hint: We found it very useful to write a helper function capable of counting the frequency of a digit in a number.

- (b) (4 points) Write the function `nthHybrindt(n)` which takes a positive integer `n` and returns the `n`th hybrindt number. `nthHybrindt(0)` should return 9, the first hybrindt number. You may assume that your implementation of `isHybrindt(n)` functions properly, even if yours does not.