

Name: _____ Andrew Id: _____

15-112 Spring 2022 Quiz 10

Up to 25 minutes. No calculators, no notes, no books, no computers. Show your work!

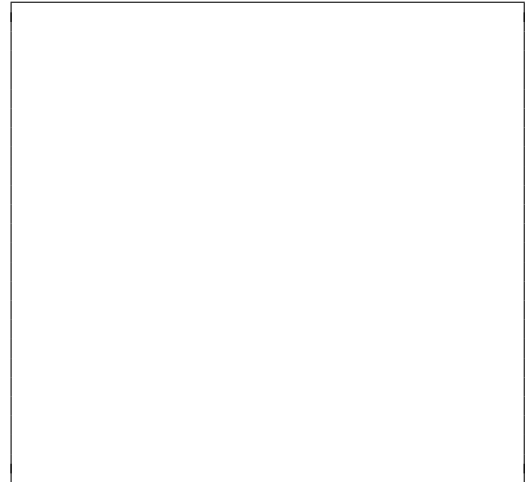
1. **Code Tracing:** Indicate what the following program prints. Place your answer (and nothing else) in the box next to the code. **NO PARTIAL GRADING**

(a) (4 points) CT1 (from Exam 2 - Fall 21)

```
def f(u):
    if 8 in u:
        print(f'8: {u[8]}')
        del u[8]
    return u

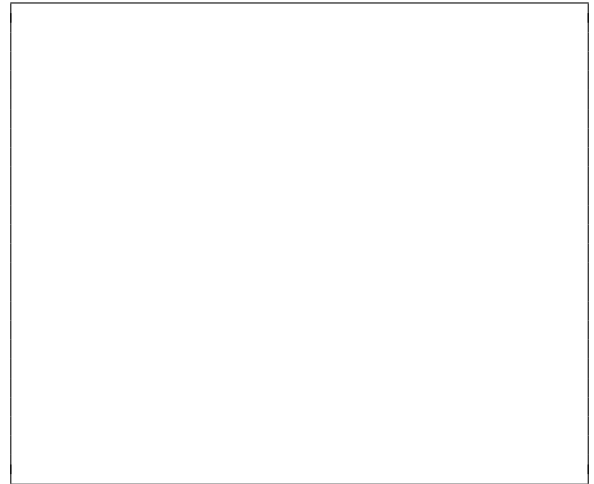
def ct(L):
    s = set(L)
    d = dict()
    for v in L:
        d[v] = d.get(v,v) + min(s)
        s.add(d[v])
    u = f(d)
    print(f's = {s}')
    print(f'd = {d}')
    print(f'u = {u}')

ct([8,4,8,4,2])
```



(b) (4 points) CT2 (from Exam 2 - Spring 22)

```
def ct2(n, d):
    print(f"IN n: {n} d: {d}")
    if (n <= 2):
        return 2+abs(n)
    else:
        ans = n + ct2(n-2, d+1)
        print(f"MID: {ans}")
        ans += ct2(n//2, d+1)
        print(f"OUT {ans} d: {d}")
        return ans
print(ct2(5, 0))
```



2. (6 points) **Free Response: isHandyWord(word, hand)** (from Quiz 6 - Spring 22)

Write the function `isHandyWord(word, hand)` that takes two strings and returns `True` if the string `word` can be formed by some arrangement of some subset of the characters in `hand`, ignoring case (so “A” and “a” are the same), `False` otherwise.

Both strings consists of **only alphabetical characters**.

```
assert( isHandyWord("Read", "adre") == True)
assert( isHandyWord("hello", "Llohe") == True)
assert( isHandyWord("hello", "Lohe") == False) # one L is missing
assert( isHandyWord("notes", "") == False)
assert( isHandyWord("in", "zz") == False)
assert( isHandyWord("lecture", "eetucrlabydf") == True)
assert( isHandyWord("a", "abcdef") == True)
```

NOTE: Your solution should run in $O(N)$ time, where N is the sum of the length of the strings

3. (6 points) **Free Response: largestSumOfPairs(a)** (derived from Homework 8 - Spring 22)

Write the function `largestSumOfPairs(a)` that takes a list of integers, and returns the largest sum of any two elements in that list, or `None` if the list is of size 1 or smaller. So,

`largestSumOfPairs([8,4,2,8])` returns the largest of $(8+4)$, $(8+2)$, $(8+8)$, $(4+2)$, $(4+8)$, and $(2+8)$, or 16.

NOTE: Your solution should run in $O(N)$ time where N is the length of `a`.

The naive solution is to try every possible pair of numbers in the list. This runs in $O(n^2)$ time and is much too slow. Another solution is to sort the list in increasing order and take the last two elements. This runs in $O(N \log N)$ and is too slow.