Name: _____ Andrew Id: _____

15-112 Fall 2024 Quiz 4

Up to 25 minutes. No calculators, no notes, no books, no computers. Show your work!
Do not use lists, tuples, dictionaries, sets, try/except, or recursion on this quiz.

1. **Code Tracing**: Indicate what the following two programs print. Place your answers (and nothing else) in the boxes below the code.

   (a) (4 points) CT1

```
def ct1(s1, s2):
    p = 'A'
    for c in s1:
        if ord(c) < ord(p):
            s2 = s2.replace(c, p)
        p = c
    return s2

a = "BOARDER"
b = "STAND"
c = ct1(a, b)
print(c)
d = ct1(c, a)
print(d)
```

   (b) (4 points) CT2

```
def ct2(s):
    res = ""
    i = 0
    while i < len(s):
        if s[i].isdigit():
            n = int(s[i])
            x = int(s[i+1:i+1+n])
            y = int(s[i+1+n:i+1+2*n])
            z = s[x:y]
            print(x, y, z)
            res += z
            i += 2*n
        i += 1
    return res

print(ct2("158applehero21821caseplot"))
```

2. (12 points) **Free Response**: Matching Pieces

   Write the function `matchingPieces(s1, s2, n)` which, given two strings and a length, returns the count of how many times a substring from `s1` of length `n` occurs in `s2`. You may assume that `n` will always be smaller than the length of both `s1` and `s2`.

   Consider the following example: `matchingPieces("abcdc", "abcdcdc", 3)`

   - There are three substrings of length 3 that can be made from s1: `abc`, `bcd`, and `cdc`.
     - `abc` occurs in `s2` once.
     - `bcd` occurs in `s2` once.
     - `cdc` occurs in `s2` twice.
   - Therefore, the function returns 4.

   Important Note: The built-in string method `count` *does not* do what you want. `"abcdcdc".count("cdc")` returns 1. This implies that you probably shouldn't use `count` and should instead write your own helper function that does what you want.

3. (1 point (bonus)) CT Bonus

```
def ctb(s):
    s.replace('z','a')
    s += "s"
    for c in "012345679":
        s = s.replace(c, "")
    return s

print(ctb("48c12zt"))
```