Name: _____ Andrew Id: _____

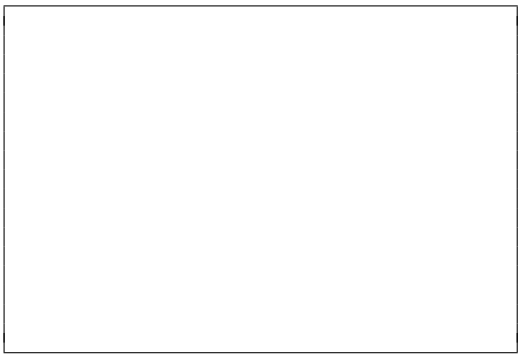
15-112 Spring 2025 Quiz 4

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

1. (6 points) **Code Tracing**: Indicate what the following program prints. Place your answer (and nothing else) in the box next to the code.

```
def ct1(s):
    result = ""
    while len(s) > 1:
        result = result + s[:1] + s[2:4]
        s = s[1:-1:2]
        print('s:', s)
    print(result.replace('.', '#'))
    return result + '\n' + s

s = "ic.mau9q2"
print(ct1(s))
print(s)
```



2. (7 points) **Free Response: Quasi-Palindrome**

A palindrome is a string that reads the same forward and backward. In a previous homework assignment, you explored palindrome strings. This time, we will focus on **quasi-palindromes**.

A string is called a quasi-palindrome if:

- It is not a palindrome by itself, but
- it becomes a palindrome after removing exactly one character from it.

Here are some examples:

- `"abcdba"` is a quasi-palindrome because removing `'d'` makes it `"abcba"`, which is a palindrome.
- `"bbaaab"` is a quasi-palindrome because removing one `'b'` from the left makes it `"baaab"`, which is a palindrome.
- `"racescar"` is a quasi-palindrome because removing `'s'` makes it `"racecar"`, which is a palindrome.
- `"racecar"` is **not** a quasi-palindrome because it is already a palindrome.
- `"abcdef"` is not a quasi-palindrome because removing any character does not make it a palindrome.

Write a function `isQuasiPalindrome(s)` that determines whether a given string `s` is a quasi-palindrome. The function should return `True` if s is a quasi-palindrome and `False` otherwise. You can assume `s` is always a string value.

3. (7 points) **Free Response: Smallest Integer Inside Square Brackets**

   Write the function `smallestIntInsideBrackets(s)` that finds the smallest integer among all numbers appearing **inside square brackets `[ ]`** in the given string `s`.

   If there are no valid integers inside square brackets, return `None`. A valid integer consists only of digits and may have an optional leading − (negative sign). Decimal numbers (floats) are not considered valid integers.

   Here are some test cases:

```
assert smallestIntInsideBrackets("15[112] Spring [25]") == 25
assert smallestIntInsideBrackets("15[112] Spring [25] -42") == 25
assert smallestIntInsideBrackets("15[112] Spring [25] [-42]") == -42
assert smallestIntInsideBrackets("NoBracketsHere") == None
assert smallestIntInsideBrackets("Floats are not Ints [15.112]") == None
assert smallestIntInsideBrackets("Be [care]ful with [non]digits [42]") == 42
```