① s = "egg"
t = "add"

Isomorphic strings have to mirror each other. Meaning s -> t and t -> s.

So if e -> a
g -> d
g -> d because g is assigned to the same letter "d" ~~both~~ times it passes the test.

Then, a -> e
d -> g
d -> g this goes both ways.

∴ egg maps to add.

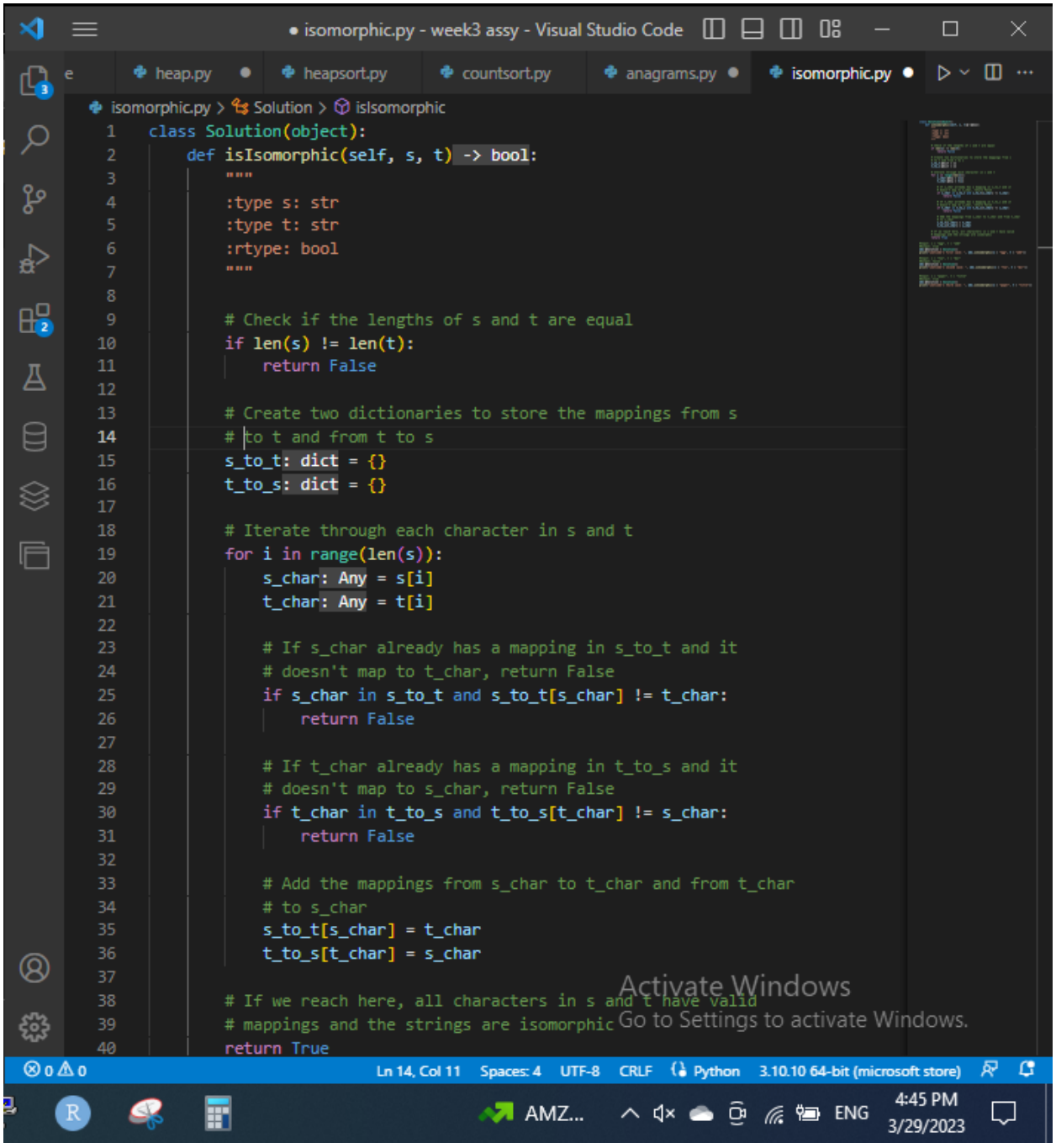
② s = foo
t = bar

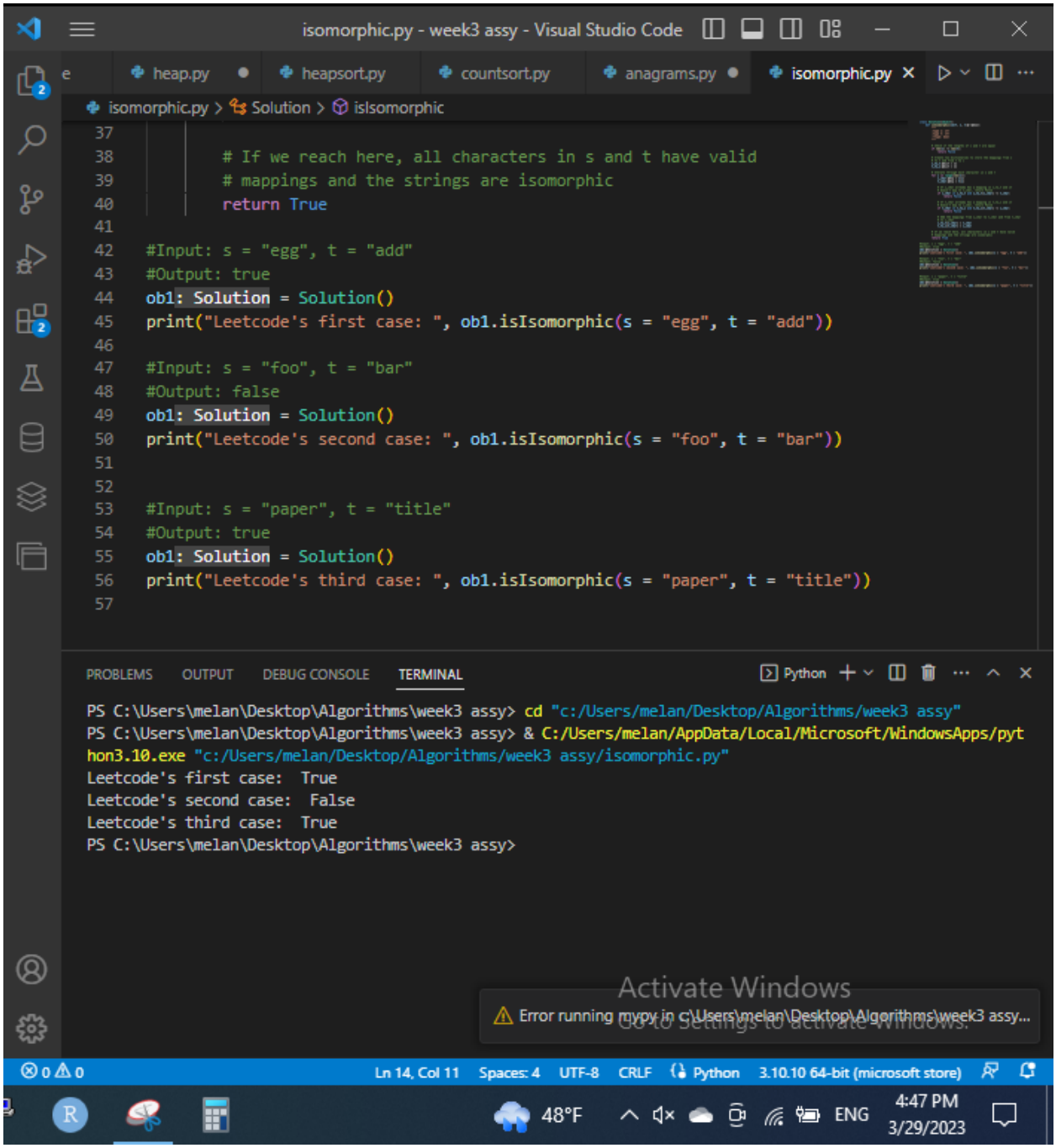
f -> b ✓
o -> a ✓
o -> r ✗ because o maps to two different letters. And no two characters may map to the same character.

check for s => t.

b -> f ✓
a -> o ✓
r -> o ✗

∴ foo does not map to bar.


• isomorphic.py - week3 assy - Visual Studio Code
heap.py
heapsort.py
countsort.py
anagrams.py
isomorphic.py
isomorphic.py > Solution > isIsomorphic
class Solution(object):
def isIsomorphic(self, s, t) -> bool:
"""
:type s: str
:type t: str
:rtype: bool
"""
# Check if the lengths of s and t are equal
if len(s) != len(t):
return False
Create two dictionaries to store the mappings from s
to t and from t to s
s_to_t: dict = {}
t_to_s: dict = {}
# Iterate through each character in s and t
for i in range(len(s)):
s_char: Any = s[i]
t_char: Any = t[i]
# If s_char already has a mapping in s_to_t and it
# doesn't map to t_char, return False
if s_char in s_to_t and s_to_t[s_char] != t_char:
return False
# If t_char already has a mapping in t_to_s and it
# doesn't map to s_char, return False
if t_char in t_to_s and t_to_s[t_char] != s_char:
return False
# Add the mappings from s_char to t_char and from t_char
# to s_char
s_to_t[s_char] = t_char
t_to_s[t_char] = s_char
# If we reach here, all characters in s and t have valid
# mappings and the strings are isomorphic
return True
Activate Windows
Go to Settings to activate Windows.
Ln 14, Col 11
Spaces: 4
UTF-8
CRLF
Python
3.10.10 64-bit (microsoft store)
AMZ...
ENG
4:45 PM
3/29/2023


isomorphic.py - week3 assy - Visual Studio Code
heap.py
heapsort.py
countsort.py
anagrams.py
isomorphic.py
isomorphic.py > Solution > isIsomorphic
If we reach here, all characters in s and t have valid
mappings and the strings are isomorphic
return True
#Input: s = "egg", t = "add"
#Output: true
ob1: Solution = Solution()
print("Leetcode's first case: ", ob1.isIsomorphic(s = "egg", t = "add"))
#Input: s = "foo", t = "bar"
#Output: false
ob1: Solution = Solution()
print("Leetcode's second case: ", ob1.isIsomorphic(s = "foo", t = "bar"))
#Input: s = "paper", t = "title"
#Output: true
ob1: Solution = Solution()
print("Leetcode's third case: ", ob1.isIsomorphic(s = "paper", t = "title"))
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Python
PS C:\Users\melan\Desktop\Algorithms\week3 assy> cd "c:/Users/melan/Desktop/Algorithms/week3 assy"
PS C:\Users\melan\Desktop\Algorithms\week3 assy> & C:/Users/melan/AppData/Local/Microsoft/WindowsApps/python3.10.exe "c:/Users/melan/Desktop/Algorithms/week3 assy/isomorphic.py"
Leetcode's first case: True
Leetcode's second case: False
Leetcode's third case: True
PS C:\Users\melan\Desktop\Algorithms\week3 assy>
Activate Windows
Error running mypy in c:\Users\melan\Desktop\Algorithms\week3 assy...
Ln 14, Col 11
Spaces: 4
UTF-8
CRLF
Python
3.10.10 64-bit (microsoft store)
48°F
ENG
4:47 PM
3/29/2023

CODE

```
class Solution(object):
    def isIsomorphic(self, s, t):
        """
        :type s: str
        :type t: str
        :rtype: bool
        """

        # Check if the lengths of s and t are equal
        if len(s) != len(t):
            return False

        # Create two dictionaries to store the mappings from s
        # to t and from t to s
        s_to_t = {}
        t_to_s = {}

        # Iterate through each character in s and t
        for i in range(len(s)):
            s_char = s[i]
            t_char = t[i]

            # If s_char already has a mapping in s_to_t and it
            # doesn't map to t_char, return False
            if s_char in s_to_t and s_to_t[s_char] != t_char:
                return False

            # If t_char already has a mapping in t_to_s and it
            # doesn't map to s_char, return False
            if t_char in t_to_s and t_to_s[t_char] != s_char:
                return False

            # Add the mappings from s_char to t_char and from t_char
            # to s_char
            s_to_t[s_char] = t_char
            t_to_s[t_char] = s_char

        # If we reach here, all characters in s and t have valid
        # mappings and the strings are isomorphic
        return True

#Input: s = "egg", t = "add"
```

```
#Output: true
ob1 = Solution()
print("Leetcode's first case: ", ob1.isIsomorphic(s = "egg", t = "add"))

#Input: s = "foo", t = "bar"
#Output: false
ob1 = Solution()
print("Leetcode's second case: ", ob1.isIsomorphic(s = "foo", t = "bar"))


#Input: s = "paper", t = "title"
#Output: true
ob1 = Solution()
print("Leetcode's third case: ", ob1.isIsomorphic(s = "paper", t = "title"))
```