Part of a rearrangement for Fall 2023 semester.

Name:_____ **Partner:** _____

Python Activity 28a: Sorting

Learning Objectives
Students will be able to:
Content:
- Explain the difference between the ***sorted(..)*** built-in function
- Describe the relationship between the built-in functions ***ord(..)*** and ***chr(..)***

Process:
- Write code that sorts sequences by creating new sequences
- Write code that converts characters to ASCII values and vice versa
- Write code that sorts mutable sequences *in place*

Prior Knowledge
- Python concepts: sequences, tuples, methods

Critical Thinking Questions:

1. Examine the sample code below.

Interactive Python

```
0 >>> nums = {42, -20, 13, 10, 0, 11}
1 >>> sorted(nums)
2 [-20, 0, 10, 11, 13, 42]
3 >>> letters = ['a', 'z', 'c', 'Z', 'A']
4 >>> sorted(letters)
5 ['A', 'Z', 'a', 'c', 'z']
```

a. What *type* of value is `nums`? _____
What *type* of value is the output on line 2? _____
How else does the output from line 3, differ from the input of `nums`?
_____

b. What might the built-in function ***sorted(..)*** be doing on line 1?
_____

c. How does the output from line 5 differ from the input of `letters`?
_____

d. How might the built-in function ***sorted(..)*** determine the ordering of characters?
_____

2. Examine the following interactive Python session:

```
0 >>> sorted("Pixel!")
1 ['!', 'P', 'e', 'i', 'l', 'x']
```

a. What *type* of value is `"Pixel"`? _____
b. What *type* of value is the output on line 1? _____
_____

c. How might the built-in function ***sorted(..)*** order strings?
_____

3. The following code continues from the previous example:

```
0 >>> ord('!')
1 33
2 >>> ord('P')
3 80
4 >>> ord('p')
5 112
6 >>> chr(33)
7 '!'
```

a. What is similar about lines 0 and 7? _____

What is similar about lines 1 and 6? _____

b. What might be returned by `chr(80)`? _____

c. What might the built-in function ***ord(..)*** do?

_____

d. What might the built-in function ***chr(..)*** do?

_____

FYI: Characters in Python (and most programming languages) have a numerical representation or *encoding*. In Python, this mapping from character to number follows the ***ASCII*** (American Standard Code for Information Interchange) protocol. Special characters come first in the mapping, then capital letters, then lowercase letters. Characters are encoded using integers from 0-127.

4. Examine the interactive Python session:

```
0 >>> fruits = [[12, "plum"], [4, "grape"], [27, "lime"]]
1 >>> sorted(fruits)
2 [[4, 'grape'], [12, 'plum'], [27, 'lime']]
```

a. What type of list is `fruits`? A list of _____

b. How does `fruits` differ from what's returned by ***sorted(..)*** on line 2?

_____

c. What rules might Python use to sort a sequence of sequences?

_____

FYI: Python uses ***lexicographical*** sorting to sort sequences of sequences (i.e., ascending order by their first item). If there is a tie, Python breaks the tie by comparing the second items. If the second items are also tied, it compares the third items and so on.

6. Examine the following code, that continues from the previous example:

```
3 >>> fruits = [[12, "plum"], [4, "grape"], [27, "lime"]]
4 >>> sorted(fruits, reverse=True)
5 [[27, 'lime'], [12, 'plum'], [4, 'grape']]
```

a. Circle what is different about the code written in this example, from the previous question.

b. How does the value returned on line 5 differ from the value returned on line 2 in the previous example? _____

c. What might the `reverse=True` argument value do? _____

_____

Application Questions: Use the Python Interpreter to check your work

1. Write a function, `shift(letter, shift_by)` that takes a character, `letter`, and shifts it in the alphabet by `shift_by`. This is called a ***Caesar Cipher***. Use the built-in functions ***ord(..)*** and/or ***chr(..)***. You may need to look up ASCII values to shift around the end of the alphabet. `shift('a', 3)` should return `'d'`. `shift('Z', 4)` should return `'E'`.

```
def shift(letter, shift_by):
```