Name:_____ **Partner:** _____

Python Activity 19: Tuples

Learning Objectives
Students will be able to:
Content:
- Define a **tuple**
- Identify **elements** of a tuple
- Explain how to access individual elements of a tuple
- Explain how to replace an item in a tuple

Process:
- Write code that prints a tuple
- Write code that edits a tuple – add, remove, and insert items

Prior Knowledge
- Python concepts from Activities 1-19

Folks, this is a brand new activity. If you encounter any issues/typos, please let Iris know!

Critical Thinking Questions:

FYI: A ***tuple*** is an immutable object that stores multiple data items in a contiguous manner. Each value stored in a tuple is called an **element.**

1. Examine the sample lists below.

Sample Tuples in Python

```
0 >>> mylist = [“pixel”, “tally”, 2]
1 >>> mytup = (“pixel”, “tally”, 2)
2 >>> mylist == mytup
3 False
```

a. How many **elements** does the list named `mylist` contain? _____
b. How many **elements** does the tuple named `mytuple` contain? _____
c. What element is at `mytuple[1]`? _____ And at `mytuple[2]`? _____
d. How do the elements of `mylist` and `mytuple` differ?
_____
e. Why does the comparison on line 2 return `False`?
_____
f. How does the syntax for defining a tuple differ from the syntax for a list?
_____

2. The following code continues from the previous example:

```
4 >>> mytup1 = (“pixel”, “tally”, 2)
5 >>> mytup2 = “pixel”, “tally”, 2
6 >>> mytup1 == mytup2
7 True
```

a. Write a line of code to access the last element of `mytup1` with indexing: _____

b. How do the elements of `mytup1` and `mytup2` differ? _____

c. How do lines 4 and line 5 differ?

_____

d. Write a line of code to create a new tuple using the parentheses notation style:

_____

e. Write some code that iterates through two tuples, `t1` and `t2`, and compares values at each index. It prints “Not Equal!” when it encounters two values that are different, and “Equal!” when the 2 values are equivalent:

_____

_____

_____

_____

3. The following code continues from the previous examples:

```
8  >>> mylist.append(42)
9  >>> mytup.append(42)
10 AttributeError: 'tuple' object has no attribute 'append'
```

a. What is stored in `mylist` after line 8? _____

b. What is stored in `mytup` after line 9?

_____

c. What might the `AttributeError` on line 10 mean?

_____

FYI: Tuples are **immutable**, and so they do not have any methods to modify the tuple itself. You’ll need to construct a new tuple in order to change a tuple.

4. Examine the following code:

```
0 >>> mytup = “pixel”, “tally”, 2
1 >>> mytup += 72,
2 >>> mytup
3 (‘pixel’, ‘tally’, 2, 72)
```

a. How does what is stored in `mytup` at line 2 differ from what it contains at line 0?

_____

b. What type of object is `mytup`? _____

c. What type of object is `72,` ? _____

d. Rewrite `72,` in its alternative format: _____

e. Why does line 1 append an item to a tuple, while `.append(obj)` throws an error?

_____

5. Examine the following code, it is similar to the previous example:

```
0 >>> mytup = “pixel”, “tally”, 2
1 >>> mytup += 72
2 TypeError: can only concatenate tuple (not ‘int’) to tuple
```

a. What type of object is `mytup`? _____

b. What type of object is `72`? _____

c. How should we modify line 1 to append `72` to our tuple?

_____

d. Write a line of code to append the string “second” to the tuple, `mytup`:

_____

6. Examine the following code, it is similar to the previous example:

```
0 >>> mytup = (“pixel”, “tally”, 2)
1 >>> mytup[1] = “wally”
2 TypeError: ‘tuple’ object does not support item assignment
```

a. What is the programmer trying to do on line 1?

_____

b. Write some lines of code to replace the second element of `mytup` with “wally”:

_____

_____

_____

7. Examine the following code for creating new tuples:

```
0 >>> etup = ()
1 >>> len(etup)
```

a. How do we know that `etup` is a tuple?

_____

b. What will the output of line 1 be? _____

c. Write some code that *iterates* through the list, `mylist`, and adds the items to a new tuple, `etup`:

```
mylist = range(0,100)
```

8. Examine the following code for using tuples to create new variables:

```
0 >>> a, b, c = 99, 77, 55
1 >>> a
2 99
3 >>> c
55
```

a. What value is stored in the variable, `b`?

b. Explain what is occurring on line 0:

c. Write *one* line of code to assign 5 different values to 5 different variables:

FYI: Tuple assignment allows for a tuple of variables on the lefthand side of an assignment operator to be assigned the values of a tuple on the righthand side.

9. Examine the following code for parentheses use:

```
0 >>> s = ('cheese')      3 >>> s = ('cheese',)      6 >>> s = 'cheese',
1 >>> type(s)             4 >>> type(s)              7 >>> type(s)
2 <class 'str'>           5 <class 'tuple'>          8 <class 'tuple'>
```

a. How do lines 0, 3, and 6 differ?

b. Why is the s from line 3 & 6 a tuple, but not line 3?

Application Questions: Use the Python Interpreter to check your work

1. Write some lines of code that use tuple assignment to swap two variables' values:

2. Write a python program that checks whether a tuple contains the value "winner" by iterating through the items of the tuple:

3. Write a python program that finds the repeated values inside of a tuple:

4. Given a **list** of **tuples**, write a program that changes the last element of each tuple to "last":

Excerpt from Python Activity 10: Looping Structures -- Nested Loops

1. Enter and execute the following code:

```
name = input("What is your name: ")
for x in range(5):
    for x in range(3):
        print(name + " ", end=" ")
    print()
```

a. What might the program display?

_____

_____

b. How many FOR loops are in this code? _____ Is one loop completely executed before the next loop begins? _____ What do you call this type of loop? _____

c. How many times is the following line of code executed in the program? _____

```
print(name + " ", end=" ")
```

d. Label the **inner loop** and the **outer loop**.

e. What does the **inner loop** do? _____

f. What does the **outer loop** do? _____

2. If you were asked to create a Python program that displayed the adjacent rectangle, you could easily do it with a set of print statements. You can also create it with a FOR loop and a print statement. This exercise will go through the steps to create a program that will print similar output but allows the user to determine the length and width of the figure when they execute the program.

```
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

a. Create a code segment that prompts the user for a number between 1 and 10 and then prints that many asterisks (*) on one line. Use a FOR loop.

_____

_____

_____

b. You want the program to create several lines of asterisks. Extend the code in "a." to also prompt the user for how many rows to print. Use an "outer" loop to print that many lines of asterisks. Write the revised code below.

_____

_____

_____

_____

_____

c. Edit the program so that it prints numbers instead of asterisks. Write the line of code that was changed.

_____

```
1 1 1 1 1 1
2 2 2 2 2 2
3 3 3 3 3 3
4 4 4 4 4 4
```