Howley, Iris. (2020). "Adapting Guided Inquiry Learning Worksheets for Emergency Remote Learning."
Supplementary materials for the *Journal of Information and Learning Sciences*.

Name: _____ **Partner:** _____

Python Activity 20a: Dictionaries, Part 1

Learning Objectives

Students will be able to:

Content:

- Define a **dictionary**.
- Identify the **key** and **value** pair of a dictionary.
- Explain why a dictionary is a good data structure for organizing data.

Process:

- Write code that accesses the keys, values, and length of a dictionary.
- Write code to create and modify dictionaries.
- Write code that iterates over a dictionary's keys.

Prior Knowledge

- Python concepts from Activities 1-19.

Critical Thinking Questions:

1. Examine the sample code defining a list of lists, below:

Sample Code

```
dog2owner = [['pickle','iris'],['rex','saul'],['tex','doug']]
print(dog2owner[0][0]) # prints: 'pickle'
```

a. What's stored at `dog2owner[0][0]`?_____

b. What might be stored at `dog2owner[0][1]`? _____

c. Write a line of code to print the name of Rex's owner using `dog2owner`:

_____

d. Write a line of code to access and print the name of Doug's dog via `dog2owner`:

_____

e. As `dog2owner` gets bigger and bigger (the CS department is growing!), will a list of a lists be an accessible way to continue storing this information?

_____

2. The following code occurs in interactive Python and introduces a new data structure:

```
0 >>> dt = {'pickle':'iris','rex':'saul','tex':'doug'}
1 >>> dt['rex']
2 'saul'
```

a. What does `dt['rex']` do?

_____

b. How might python know that Rex (the dog) is mapped to Saul (the owner)? Where is that relationship defined?

_____

c. In the line, `dt['rex']`, what does the value in the square brackets represent?

_____

FYI: A ***dictionary*** is a data structure that is similar to a list, but instead of storing ***values*** at numerical indices, ***values*** are mapped to ***keys.*** Keys must be an immutable data type.

d. Write a line of code to print the name of your CS134 instructor's name, accessed via the dictionary, `dt`:

_____

e. Why might a dictionary be a better data structure for this data than a list of lists?

_____

f. How would you describe the ***keys*** and ***values*** for this dictionary, `dt`?

keys:_____ values:_____

g. What type of data is stored in the keys and the values for `dt`?

keys:_____ values:_____

3. Examine the following code from interactive Python:

```
0 >>> dt = {'pickle':'iris','rex':'saul','tex':'doug'}
1 >>> dt['lilac'] = 'jenn'
2 >>> dt
3 {'pickle':'iris','rex':'saul','tex':'doug','lilac':'jenn'}
```

a. What does the line `dt['lilac'] = 'jenn'` do?

_____

b. What might this imply about the ***mutability*** of dictionaries?

_____

c. What does the object in square brackets on the left hand side of the assignment operator in line 1 represent? (*Circle one*) key or value

d. What does the object on the right hand side of the assignment operator in line 1 represent? (*Circle one*) key or value

e. Write a line of code to add `Bob` and his dog, `Alpha`, to our dictionary.

_____

4. Examine the following code from interactive Python:

```
0 >>> csPets = {'dogs':6, 'cats':3, 'bees':20000}
1 >>> len(csPets)
2 3
```

a. What type of data is stored in the keys and the values for `csPets`?

keys:_____ values:_____

b. How many keys does `csPets` have? _____

c. What is the length `csPets`? _____

d. How does python determine the length of a dictionary object?

_____

e. If we added a line 3 of code, `csPets['others'] = ['hamster', 'ferret']`, what might `len(csPets)` return? _____

5. Examine the following example code from interactive python:

Interactive Python

```
0 >>> d = dict()     # can also do: d = {}
1 >>> d
2 {}
```

a. If we wrote line 3 of code, `len(d)`, what might be the output? _____

b. Write some code to create an empty dictionary, then ask the user for `input(..)` for today's month, then day, then year. Place the data into `month, day, year` keys, mapped to the user's input values, into the empty dictionary:

_____

_____

_____

_____

_____

_____

_____

6. Examine the following example code:

```
>>> coll = {}        # can also do: coll = dict()
>>> coll['colleges'] = 'williams'
>>> coll['colleges'] = 'amherst'
```

a. If we wrote a fourth line of code, `print(coll)`, what might be the output?

_____

b. At the end of this code execution, `coll` only has: `{'colleges': 'amherst'}` Why might this be?

_____

FYI: Dictionaries can only have <u>one</u> key of its value, any replicated key:value mappings added will simply overwrite the previous one!

7. Examine the following example code from interactive python:

```
0 >>> date = {'month':'dec', 'day':9, 'year':1906}
1 >>> for mykey in date:
2 ...     print("The {} is {}.".format(mykey, date[mykey]))
```

a. What data does the dictionary, `date`, appear to hold?

_____

b. If you had to guess, what might the programmer want to be output by line 2?

_____

c. For the first defined item of `date` what might `mykey` and `date[mykey]` refer to on lines 1 & 2?

`mykey:`_____ `date[mykey]:`_____

d. The first time through the loop defined on line 1, line 2 might print `'The month is dec.'` What might be printed the second time through the loop?

_____

e. What does line 1, `for mykey in date:`, do?

_____

f. Write some code that will iterate over the items in `date` and print *only* the values:

_____

_____

_____

_____

Application Questions: Use the Python Interpreter to check your work

1. Write a function that checks if a given dictionary, `d`, has a given `key`. If it doesn't, create a new list at `key` with the given `value` as its only element. If it does already have the `key`, append `value` to the existing list mapped to `key`.

```
def appendDictList(d, key, value):
```

2. Write a function, `dataEntry` that collects data from the user to put into a dictionary. The user should be prompted for a key, and then value data to be added to a dictionary, and this process should be repeated until they enter the text `'done'`. For extra bonus points, use your previous function, `appendDictList`, to ensure that no data is overwritten, even if a key is duplicated! The `dataEntry` function should return the dictionary when the process is done.

```
def dataEntry():
```