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:

   ```python
   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:

   ```python
   dt = {'pickle':'iris','rex':'saul','tex':'doug'}
   print(dt['rex'])
   ```

   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:

```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:

```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:

```python
>>> 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 one 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:

```python
>>> date = {'month': 'dec', 'day': 9, 'year': 1906}
>>> for mykey in date:
...     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.
```python
def appendDictList(d, key, value):
    # Your code here
```

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.
```python
def dataEntry():
    # Your code here
```