* List and Tuples
  + Tuples (Ordered Sequence)
    - Tuples are written as comma-separated elements within parentheses.
    - A tuple can include all str,int and float all at the same time
    - Note a tuples is TYPE Tuple regardless of the contents.
    - You can access content in a tuple using index starting at 0 using tuple[0]
    - Can you combine tuples by simply adding them.
    - Can you slice tuple using tuple[0:end+1]
    - Len on tuple will return the length of the tuple.
    - Tuples are immutable (Cannot be changed) thus to manipulate the tuple you must create a new tuple.
    - To sort tuples
      * Sorted = sorted(unsorted tuple name)
    - To find the index of a element
      * Tuple/list name.index(var)
    - Graphical user interface, application

      Description automatically generatedNesting
  + Lists (also ordered sequences)
    - Lists are written with [ ] and are mutable that means it can be changed.
    - Can also contain str, int, floats, lists and tuples
    - Can also be sliced
    - Can be combined
    - Extending list
      * Using the .extend we can add elements into the list
      * Ex: L.extend(“ “, 10)
      * Elements will be added at the end of it.
    - Appending (Used for adding lists or tuples as a single element)
      * L.append([“pop”, 10])
      * This will added the following list as a single element into L.
    - Changing elements
      * Using L[index] = whatever, to set the data to element at index.
    - Deleting elements
      * Using del(list[index]) to delete.
    - Converting strings to a list
      * “hard rock”.split()
      * [“hard”,”rock”]
      * Using a delimited to determine where to split
        + “A,B,C,D”.split(“,”) # note here were are using , to split the string into [“A”,”B”,”C”,”D”]
    - Aliasing (where 2 names references the same list)
    - Clone
      * A=[“hard rock”, 10, 1.2]
      * B=A[:] essentially sets List A to B
    - Help command
      * help(list name or tuple name)
* Dictionaries
  + Unlike a list which has integers indexes a dictionary has keys as an index.
  + Dictionaries are denoted using { }
    - Ex: release\_year\_dict = {"Thriller": "1982", "Back in Black": "1980", \
    - "The Dark Side of the Moon": "1973", "The Bodyguard": "1992", \
    - "Bat Out of Hell": "1977", "Their Greatest Hits (1971-1975)": "1976", \
    - "Saturday Night Fever": "1977", "Rumours": "1977"}
  + Keys can be set as anything (the value on the left is the key, right is the value)
  + Using the .keys() command returns all the keys in the dictionary
  + Using the .values() command returns all the values in the dictionary.
  + To retrieve values
    - Ex: release\_year\_dict['Thriller']
  + To append values into the dictionary
    - Ex: release\_year\_dict['Graduation'] = '2007'
  + To Delete
    - Ex: del(release\_year\_dict['Thriller'])
  + To verify the key
    - 'The Bodyguard' in release\_year\_dict #will return either true or false.
* Sets
  + Type of collection
  + Unlike list and tuples they are unordered thus have no element position index.
  + Sets only have unique elements.
  +  Creating a set (Note duplicate items will be combined to just 1)
  + Text

    Description automatically generated with medium confidenceConverting a List to a Set (Note duplicate items are consolidated into 1)
  + Set operations
    - Add (NOTE THERE CANNOT BE DUPLICATES IN A SET)
      * SetName.add(“ “)
    - Remove
      * SetName.remove(“ “)
    - Find
      * “name” in setName #returns true or false
    - Difference
      * SetName.difference(setname2) 3 returns the elements that are not included in both.
    - Mathematical
      * Graphical user interface, text, application

        Description automatically generatedIntersecting 2 sets (Using &)
      * Union
        + Album\_set\_1.union(album\_set\_2)

The resulting set will have all the elements from both sets.

* + - * Chart, pie chart

        Description automatically generatedIssubset? Finding a set is a subset or another set. Or issuperset?