Dictionaries:

* Objects retrieved by KEY name
* Unordered and cannot be sorted

Lists:

* Objects retrieved by location
* Ordered sequence can be indexed or sliced

Dictionary example

prices={'apples':1.99,'oranges':2.00,'milk':3.50}

prices['apples']

Tuples

Does not support item re-assignment

Why bother using them? Passing around objects in programs, data integrity, when we need to ensure elements aren’t changed

Sets are unordered collections of unique elements

Myset=set()

Myset.add(1)

No repeatability

If else if statements

loc = 'park'

if(loc== 'bank'):

print('At bank')

elif(loc=="park"):

print('Not at bank')

else:

print("At home")

a=False

if(a==True):

print('~Feed me~')

else:

print("Don't feed me, not hungry")

Tuple unpacking with for loops

Suppose mylist = [(1,2),(3,4),(5,6),(7,8)], list of tuples

For a,b in mylist:

Print(a)

Print(b)

//gives us access to both

Can do tuple unpacking with dictionaries as well

d={'k1':1,'k2':2,'k3':3}

for key,value in d:

print(value)

While loop with python

while x!=10:

print(x)

x+=1

break: breaks out of current enclosing loop

continue: Goes to the top of the closest enclosing loop

pass: does nothing at all

Range function

for num in range(4,13,4):

print(num)

list(range(0,10))

for letter in word:

print(f'the letter is {letter} and the index is {count}')

count+=1

count =0

word='abcde'

for item in enumerate(word):

print(item)

//enumerate does our index count for us automatically

if 'x' in ['y']:

print(True)

else:

print(False)

//in keyword operator

Convert string to list style

mylist=[letter for letter in mystring] or mylist=[x for lx in ‘hello’] or mylist = [num for num in range(0,11)]

can perform actions like squaring from this line too

mylist = [num\*\*2 for num in range(0,11)]

mylist = [num\*\*2 for num in range(0,11) if num%2==0]

fahrenheit = [(9/5)\*x+32 for x in celsius]