Mutable structure can be updated in-place

List and dictionaries

Immutable: cannot be updated in-place, we have to re-create the structure

String and Tuple

Aliasing: two or more names refer to the same (memory location)

In [1]: list1 = [1, 2, 3]
   print(list1)
   print(list1)
   [1, 2, 3]

In [2]: print(list1 is list1)
   True

In [3]: mylist1 = ['a', 'b', 1, 2]
mylist1 is mylist1
   print(mylist1)
   ['a', 'b', 1, 2]
   print(mylist1 is mylist1)
   False

In [4]: print(mylist1 is mylist1)
   False

In [5]: mylist1 = ['a', 'b', 1, 2, ['x', 'x', 'cs101', 'cs101']]
mylist1 is mylist1
   print(mylist1)
   ['a', 'b', 1, 2, ['x', 'x', 'cs101', 'cs101']]
   print(mylist1 is mylist1)
   False

In [6]: print(mylist1 is mylist1)
   False

In [7]: num = 10
   count = 0
   while num > 0:
      letter = input("letter")
      if letter in myLst1:
         count += 1
      num -= 1
   print(count)
10

In [8]: # we are breaking the loop
   break

break terminates a loop when a termination condition is satisfied

In [9]: for x in range(10):
   if x == 10:
      print("we are breaking the loop")
      break
1

read 10 values from the user, stop if the total reaches 100
or the end of the loop. Also print the number of iterations

In [10]: count = 0
   while num > 0:
      letter = input("letter")
      if letter in myLst1:
         count += 1
      num -= 1
   print(count)
10

continue: skips the remainder of the loop body

In [11]: for x in range(10):
   if x == 10:
      print("we are skipping the remainder of this iteration")
      continue
1

continue: skips the remainder of the loop body

In [12]: word = input("Enter a word: ")
for letter in word:
   if letter in list2:
      print(letter)
we are skipping the remainder of this iteration

zip allows traversing two lists at the same time

In [13]: list1 = ['a', 'b', 1, 2]
   list2 = [1, 2, 3, 4, 5]
   zip(list1, list2)
   [('a', 1), ('b', 2), (1, 3), (2, 4), (3, 5)]

enumerate helps with keeping track of elements and their locations

In [14]: list1 = ['a', 'b', 1, 2]
   enumerate(list1)
   [(0, 'a'), (1, 'b'), (2, 1), (3, 2)]
   print(index, value)
   0 a
   1 b
   2 1
   3 2

Search for an element in the list, return the search outcome (True/False) and
the number of iterations. Your solution should use a break