**Exercise 1**

**Solution with the “listener pattern”**

inp = input("Please enter integer:")

total\_lst = []

while inp.lower() != "exit":

inp = int(inp) #tricky

total\_lst.append(inp)

inp = input("Please enter integer:")

**Solution with the "flag pattern":**

flag = True

total\_lst = []

inp = input("Please enter integer:")

while flag: # or: while flag == True:

inp = int(inp) #tricky

total\_lst.append(inp)

inp = input("Please enter integer:")

if inp.upper() == "EXIT":

flag = False

**Solution with control statement break:**

total\_lst = []

inp = input("Please enter integer:")

while True: # or: while flag == True:

inp = int(inp)

total\_lst.append(inp)

inp = input("Please enter integer:")

if inp.upper() == "EXIT":

break

**Exercise 2:**

**Solution 1 – with lists and dictionaries**

def most\_common\_elem(lst):

counts\_diction = {}

for elem in lst:

if elem in counts\_diction:

counts\_diction[elem] = counts\_diction[elem] + 1

else:

counts\_diction[elem] = 1

diction\_keys = list(counts\_diction.keys())

key\_with\_max\_val\_so\_far = diction\_keys[0] # remember can't do this in dictionaries, just lists! So have to do above line

# Now iterate through the rest

for k in diction\_keys[1:]:

if counts\_diction[k] > counts\_diction[key\_with\_max\_val\_so\_far]:

key\_with\_max\_val\_so\_far = k

return key\_with\_max\_val\_so\_far

# Figuring out that this is what to return and WHERE it should go is often tricky

**Solution 2 — with lists, dictionaries, and tuples**

def most\_common\_elem(lst):

# a dictionary of list elements

# the values associated to it are the element count

d = {}

# assigned the elements of lst to the dictionary

for item in lst:

if item not in d:

d[item] = 1

else:

d[item] += 1

# find the maximum value in the dictionary

mcount = 0 # to keep track of the max val so far

mkey = "" # an empty string for starters,

# but it can be reassigned to another type

# to the key time in the dictionary

for k, v in d.items(): # d.items() returns a list of tuples

if mcount < v:

mcount = v

mkey = k

return mkey

res = most\_common\_elem([1,2,2,2,3,4,2])

print(res) # should print 2

lp = [5, "six", "six", 6, 6, 6, 6, 6, 7]

print(most\_common\_elem(lp)) # should print 6

l = [3,4,5,65,7,8,8,8,8,8,9,9,2]

print(most\_common\_elem(l)) # should print 8