#SOURCES:

#https://docs.python.org/2/library/stdtypes.html

#https://wiki.python.org/moin/WhileLoop

#http://www.tutorialspoint.com/python/python\_loop\_control.htm

#Cousin helped me get my while statement to end when -9999 was entered

#http://stackoverflow.com/questions/29836964/error-python-zerodivisionerror-division-by-zero

#https://docs.python.org/3/tutorial/errors.html

#Got help from "Find the longest word" video

#http://stackoverflow.com/questions/8599440/new-line-for-input-in-python

#http://stackoverflow.com/questions/4659524/how-to-sort-by-length-of-string-followed-by-alphabetical-order

#PART 1

#creates variable assigned to the input of the user.

input\_of\_words = input("Enter a few words and I will find the longest:\n")

#split the words into separate elements

wordList = input\_of\_words.split()

#prints the list of words for the user

print("The list of words entered is:\n", wordList)

#creates new function

def find\_longest\_word(wordList):

#sorts the word list according to the length of the word and order entered

#reverse order so that the first longest word is the first in the index

wordList.sort(key=len, reverse=True)

#prints first index in list for the user

print("The longest word in the list is:\n",wordList[0])

#calls the function created above using the wordlist that wast entered

find\_longest\_word(wordList)

#PART 2

#creates a dictionary with three keys-value pairs

diction1 = {'AvgPositive':0,'AvgNonPos':0,'AvgAllNum':0}

#creates an empty list

numList = []

#creates a variable assigned to 0

n\_input = 0

#creates a while loop, since 0 does not equal -9999, do the following...

while n\_input != -9999:

print("Enter a number (-9999 to stop)")

#create an exception in-case the user enters in something other than an integer

try:

#takes the user's integer inputs and assigns them to n\_input

n\_input = int(input())

#checks if the input is equal to -9999

if n\_input != -9999:

#if the input isn't -9999, append the integer to the end of the numList

numList.append(n\_input)

#our exception in case of a Value Error (user enters letters or nothing)

except ValueError:

print("Please enter only integers!")

#prints out the list of numbers that were entered

print("The list of all numbers entered is:\n",numList)

#

#

#create function to take positive numbers from list and average them

def posNumAvg(numList):

#create local variables to store integers from our algorithm

average = 0

sum1 = 0

leng = 0

#for loop that iterates through each number in our list

for n in numList:

#checks if the current number is greater than 0

if n > 0:

#adds the current number to the total of sum1

sum1 += n

#adds 1 to the total of length (tracks how many numbers to divide by)

leng += 1

#create an exception in the case that the formula tries to divide by zero

try:

#gives us our average from the local variables

average = sum1 / leng

#assigns the new value (average) to the key 'AvgPositive' in our dictionary

diction1['AvgPositive'] = (average)

#dividing by zero gives us an error so we tell the program what we want to do

#when we it comes across this error here

except ZeroDivisionError:

#if it tries to divide by zero, this assigns the value to 0

diction1['AvgPositive'] = 0.0

#calls the posNumAvg function using the list inputed by the user

posNumAvg(numList)

#

#

#create function to take 0 and nonpositive numbers from list and average them

def nonPosAvg(numList):

#creates local variables to store integers from our algorithm

average = 0

sum1 = 0

leng = 0

#for loop that iterates through each number in our list

for n in numList:

#checks if the current number is less than or eaqual to 0

if n <= 0:

#adds the current number to the total of sum1

sum1 += n

#adds 1 to the total of length (tracks how many numbers to divide by)

leng += 1

#create an exception in the case that the formula tries to divide by zero

try:

#gives us our average from the local variables

average = sum1 / leng

#assigns the new value (average) to the key 'AvgNonPos' in our dictionary

diction1['AvgNonPos'] = (average)

#dividing by zero gives us an error so we tell the program what we want to do

#when we it comes across this error here

except ZeroDivisionError:

#if it tries to divide by zero, this assigns the value to 0

diction1['AvgNonPos'] = 0.0

#calls the nonPosAvg function using the list inputed by the user

nonPosAvg(numList)

#

#

#create function that calculates the average of all numbers

def allNumAvg(numList):

#create local variables for the average formula

average = 0

sum1 = 0

#for loop that iterates through each number in our list

for n in numList:

#adds the current number to the total of sum1

sum1 += n

#create an exception in the case that the user doesn't enter any numbers

try:

#gives us our average from the local variables, if len is 0 we get an error

average = sum1 / len(numList)

#assigns the new value (average) to the key 'AvgAllNum' in our dictionary

diction1['AvgAllNum'] = (average)

#the only way you'd get a zerodivision error here is if the user entered

#-9999 as their first number, which creates an empty list, so we make this exception

except ZeroDivisionError:

#if it tries to divide by zero, this assigns the value to 0

diction1['AvgAllNum'] = 0

#this lets the user know they didn't add any numbers to the list

print("There were no numbers in your list.")

#calls the allNumAvg fucntion using the list inputed by the user

allNumAvg(numList)

#tells the user what all their averages are and prints the dictionary

print("The dictionary with averages is:\n", diction1)