#!/usr/bin/env python

# Note : Handle all other corner cases which are not handled here

import sys

import os

from functools import reduce

# Write a python program to read contents of a file (filename as argument)

# and store the number of occurrences of each word in a dictionary.

dict = {}

wordLen = []

if(len(sys.argv) != 2):

print ("Invalid Arguments")

sys.exit()

if(not(os.path.exists(sys.argv[0]))):

print ("Invalid File Path")

sys.exit()

if(sys.argv[1].split('.')[-1] != "txt"):

print ("Invalid File Format. Only TXT files allowed")

sys.exit()

with open(sys.argv[1]) as file:

for line in file:

for word in line.split():

dict[word] = dict.get(word,0) + 1

# print dict

# Display the top 10 words with most number of occurrences in descending order.

# Food for thought - Does a dictionary maintain order? How to print 10 words with most frequency?

# Ans - extract dict items as Tuples and sort them based on value in dictionary

#(second item of the tuple / index 1)

sortedDict = sorted(dict.items(), key=lambda dictItem: dictItem[1], reverse=True)

for i in range(len(sortedDict)):

print(sortedDict[i])

for i in range(10):

try:

wordTuple = sortedDict[i]

wordLen.append(len(wordTuple[0]))

print (wordTuple[0], ", Frequency: " , wordTuple[1] , ", Length " , len(wordTuple[0]))

except IndexError:

print ("File has less than 10 words")

break

print ("Lengths of 10 most frequently occuring words:")

print (wordLen)

# Write a one-line reduce function to get the average length

sum = reduce(lambda x,y: x+y, wordLen)

print ("Average length of words: " , sum/len(wordLen))

# Write a one-line list comprehension to display squares of all odd numbers

squares = [x\*\*2 for x in wordLen if x%2 != 0]

print ("Squares of odd word lengths: ")

print (squares)