```python
# File: Benford.py
# Student: Jennifer Truong
# UT EID: Jat5244
# Course Name: CS303E
#
# Date Created: 4/21/2021
# Date Last Modified: 4/23/2021
# Description of Program: Verify Benford's law for the U.S. census data from 2009

import os.path

# Checking if the user's name of the file exist
fileName = input( "Enter the name of a file of census data: ").strip()
if not os.path.isfile( fileName ):
    print( "File does not exist" )
elif os.path.isfile( fileName ):
    print( "Output written to benford.txt")

# Open the file
infile = open( "Census_2009.txt", "r")
uniquePop = set()
digits = list()
cityCounter = 0
leadDigitCount = list()

# Read each line, except header, in the file
# and store it in the empty set

readText = infile.readlines()[1 : ]
for line in readText:
    uniquePop.add( ( line.split() ).pop() )     # To only get digits
    digits.append( line.split().pop() )
    cityCounter += 1

# A very inefficent way of counting the leading digits of population
count1 = 0
count2 = 0
count3 = 0
count4 = 0
count5 = 0
count6 = 0
count7 = 0
count8 = 0
count9 = 0

for numbers in digits:
    if numbers.startswith("1"):
        count1 += 1
    elif numbers.startswith("2"):
        count2 += 1
    elif numbers.startswith("3"):
        count3 += 1
    elif numbers.startswith("4"):
        count4 += 1
    elif numbers.startswith("5"):
        count5 += 1
    elif numbers.startswith("6"):
        count6 += 1
    elif numbers.startswith("7"):
        count7 += 1
    elif numbers.startswith("8"):
        count8 += 1
    elif numbers.startswith("9"):
        count9 += 1

# Percentage calculations, also inefficent
percent1 = round( ((count1 / cityCounter) * 100), 1 )
percent2 = round( ((count2 / cityCounter) * 100), 1 )
percent3 = round( ((count3 / cityCounter) * 100), 1 )
percent4 = round( ((count4 / cityCounter) * 100), 1 )
percent5 = round( ((count5 / cityCounter) * 100), 1 )
percent6 = round( ((count6 / cityCounter) * 100), 1 )
percent7 = round( ((count7 / cityCounter) * 100), 1 )
percent8 = round( ((count8 / cityCounter) * 100), 1 )
percent9 = round( ((count9 / cityCounter) * 100), 1 )



# Create a dictionary
leadingDigitDict = {'1' : [count1, percent1], '2' : [count2, percent2], '3' :
[count3, percent3],
                    '4' : [count4, percent4], '5' : [count5, percent5], '6' :
[count6, percent6],
                    '7' : [count7, percent7], '8' : [count8, percent8], '9' :
[count9, percent9],}


infile.close()


# Creating the benford.txt and formating the text
f = open("benford.txt", "w")


f.write( "Total number of cities: " + str(cityCounter) + "\n")
f.write( "Unique population counts: " + str(len(uniquePop)) + "\n")
f.write( "First digit frequency distributions:" + "\n")
f.write( "Digit   Count Percentage" + "\n")
for key, value in leadingDigitDict.items():
    f.write( (format(key, "<7s")) )
    f.write(' {0:<7} {1:<2} '.format(value[0], value[1]) + "\n")

f.close()
```