def main():

print("What's your file name?")

myfile = input("> ")

print("Processing ... \n")

read\_file = open(myfile, 'r').readlines()

# print(read\_file)

line\_count = 0

word\_count = 0

char\_count = 0

for line in read\_file:

sentence = read\_file[line\_count]

word\_count = len(sentence.split()) + word\_count

char\_count = len(read\_file[line\_count]) + char\_count

line\_count = line\_count + 1

print(f"Your file {myfile} has {line\_count} lines, {word\_count} words, and {char\_count} characters.")

# print(f"""{myfile}

# Lines: {line\_count}

# Words: {word\_count}

# Charactes: {char\_count}

# """)

main()

import math

def numbers():

print("Starting the Process")

myfile = open("numbers.txt", 'w')

myfile.write("Numbers Squared Sq. Root\n")

myfile.write("------- ------- --------\n")

for i in range(1, 101):

myfile.write("%6d %7d %7.3f\n" % (i, i\*i, math.sqrt(i)))

myfile.close()

print("Finished writing the File")

def read\_numbers():

print()

infile = open("numbers.txt", 'r')

# allstr = infile.read()

# print(allstr)

for i in range(10):

s = infile.readline()

# remove trailing newline character

print(s[:-1])

infile.close()

def read\_numbers2():

infile = open("numbers.txt", 'r')

# lines = infile.readlines()

# --- Print the lines in the file in a list ---

# print(lines)

# --- Read the file as a list of lines ---

# for s in lines:

# print(s[:-1])

# --- Use the file as if it is a list of lines ---

for s in infile:

print(s[:-1])

infile.close()

numbers()

read\_numbers()

read\_numbers2()

|  |
| --- |
| Numbers Squared Sq. Root |
|  | ------- ------- -------- |
|  | 1 1 1.000 |
|  | 2 4 1.414 |
|  | 3 9 1.732 |
|  | 4 16 2.000 |
|  | 5 25 2.236 |
|  | 6 36 2.449 |
|  | 7 49 2.646 |
|  | 8 64 2.828 |
|  | 9 81 3.000 |
|  | 10 100 3.162 |
|  | 11 121 3.317 |
|  | 12 144 3.464 |
|  | 13 169 3.606 |
|  | 14 196 3.742 |
|  | 15 225 3.873 |
|  | 16 256 4.000 |
|  | 17 289 4.123 |
|  | 18 324 4.243 |
|  | 19 361 4.359 |
|  | 20 400 4.472 |
|  | 21 441 4.583 |
|  | 22 484 4.690 |
|  | 23 529 4.796 |
|  | 24 576 4.899 |
|  | 25 625 5.000 |
|  | 26 676 5.099 |
|  | 27 729 5.196 |
|  | 28 784 5.292 |
|  | 29 841 5.385 |
|  | 30 900 5.477 |
|  | 31 961 5.568 |
|  | 32 1024 5.657 |
|  | 33 1089 5.745 |
|  | 34 1156 5.831 |
|  | 35 1225 5.916 |
|  | 36 1296 6.000 |
|  | 37 1369 6.083 |
|  | 38 1444 6.164 |
|  | 39 1521 6.245 |
|  | 40 1600 6.325 |
|  | 41 1681 6.403 |
|  | 42 1764 6.481 |
|  | 43 1849 6.557 |
|  | 44 1936 6.633 |
|  | 45 2025 6.708 |
|  | 46 2116 6.782 |
|  | 47 2209 6.856 |
|  | 48 2304 6.928 |
|  | 49 2401 7.000 |
|  | 50 2500 7.071 |
|  | 51 2601 7.141 |
|  | 52 2704 7.211 |
|  | 53 2809 7.280 |
|  | 54 2916 7.348 |
|  | 55 3025 7.416 |
|  | 56 3136 7.483 |
|  | 57 3249 7.550 |
|  | 58 3364 7.616 |
|  | 59 3481 7.681 |
|  | 60 3600 7.746 |
|  | 61 3721 7.810 |
|  | 62 3844 7.874 |
|  | 63 3969 7.937 |
|  | 64 4096 8.000 |
|  | 65 4225 8.062 |
|  | 66 4356 8.124 |
|  | 67 4489 8.185 |
|  | 68 4624 8.246 |
|  | 69 4761 8.307 |
|  | 70 4900 8.367 |
|  | 71 5041 8.426 |
|  | 72 5184 8.485 |
|  | 73 5329 8.544 |
|  | 74 5476 8.602 |
|  | 75 5625 8.660 |
|  | 76 5776 8.718 |
|  | 77 5929 8.775 |
|  | 78 6084 8.832 |
|  | 79 6241 8.888 |
|  | 80 6400 8.944 |
|  | 81 6561 9.000 |
|  | 82 6724 9.055 |
|  | 83 6889 9.110 |
|  | 84 7056 9.165 |
|  | 85 7225 9.220 |
|  | 86 7396 9.274 |
|  | 87 7569 9.327 |
|  | 88 7744 9.381 |
|  | 89 7921 9.434 |
|  | 90 8100 9.487 |
|  | 91 8281 9.539 |
|  | 92 8464 9.592 |
|  | 93 8649 9.644 |
|  | 94 8836 9.695 |
|  | 95 9025 9.747 |
|  | 96 9216 9.798 |
|  | 97 9409 9.849 |
|  | 98 9604 9.899 |
|  | 99 9801 9.950 |
|  | 100 10000 10.000  Numbers Squared Sq. Root  ------- ------- --------  1 1 1.000  2 4 1.414  3 9 1.732  4 16 2.000  5 25 2.236  6 36 2.449  7 49 2.646  8 64 2.828  9 81 3.000  10 100 3.162  11 121 3.317  12 144 3.464  13 169 3.606  14 196 3.742  15 225 3.873  16 256 4.000  17 289 4.123  18 324 4.243  19 361 4.359  20 400 4.472  21 441 4.583  22 484 4.690  23 529 4.796  24 576 4.899  25 625 5.000  26 676 5.099  27 729 5.196  28 784 5.292  29 841 5.385  30 900 5.477  31 961 5.568  32 1024 5.657  33 1089 5.745  34 1156 5.831  35 1225 5.916  36 1296 6.000  37 1369 6.083  38 1444 6.164  39 1521 6.245  40 1600 6.325  41 1681 6.403  42 1764 6.481  43 1849 6.557  44 1936 6.633  45 2025 6.708  46 2116 6.782  47 2209 6.856  48 2304 6.928  49 2401 7.000  50 2500 7.071  51 2601 7.141  52 2704 7.211  53 2809 7.280  54 2916 7.348  55 3025 7.416  56 3136 7.483  57 3249 7.550  58 3364 7.616  59 3481 7.681  60 3600 7.746  61 3721 7.810  62 3844 7.874  63 3969 7.937  64 4096 8.000  65 4225 8.062  66 4356 8.124  67 4489 8.185  68 4624 8.246  69 4761 8.307  70 4900 8.367  71 5041 8.426  72 5184 8.485  73 5329 8.544  74 5476 8.602  75 5625 8.660  76 5776 8.718  77 5929 8.775  78 6084 8.832  79 6241 8.888  80 6400 8.944  81 6561 9.000  82 6724 9.055  83 6889 9.110  84 7056 9.165  85 7225 9.220  86 7396 9.274  87 7569 9.327  88 7744 9.381  89 7921 9.434  90 8100 9.487  91 8281 9.539  92 8464 9.592  93 8649 9.644  94 8836 9.695  95 9025 9.747  96 9216 9.798  97 9409 9.849  98 9604 9.899  99 9801 9.950  100 10000 10.00 |

|  |
| --- |
| def main(): |
|  | print("Enter the name of your file below.") |
|  | myfile = input("> ") |
|  | read\_file = open(myfile, 'r').readlines() |
|  |  |
|  | # Initialize counters |
|  | line\_count = 1 |
|  | column1, column2, column3 = 0, 0, 0 |
|  |  |
|  | # print(range(1, len(read\_file) - 1)) |
|  | # line\_count = 1 |
|  | # column1 = 0 |
|  | # column1 = float(((read\_file[line\_count]).split())[0]) + column1 |
|  | # print(column1) |
|  |  |
|  | for line in range(1, len(read\_file) - 1): |
|  | # Get line |
|  | # Split line |
|  | # Get column number |
|  | # print(line\_count) |
|  |  |
|  | column1 = float(((read\_file[line\_count]).split())[0]) + column1 |
|  | column2 = float(((read\_file[line\_count]).split())[1]) + column2 |
|  | column3 = float(((read\_file[line\_count]).split())[2]) + column3 |
|  |  |
|  | line\_count = line\_count + 1 |
|  |  |
|  | print("Column Sum Average") |
|  | print("------ --------------- -------------") |
|  |  |
|  | column = [column1, column2, column3] |
|  | line\_count = line\_count - 1 |
|  |  |
|  | for i in column: |
|  | print("%6d %15.2f %13.3f\n" % (1, i, i / line\_count), end='') |
|  |  |
|  | # print("%6d %15.2f %13.3f\n" % (1, column1, column1 / line\_count), end='') |
|  | # print("%6d %15.2f %13.3f\n" % (2, column2, column2 / line\_count), end='') |
|  | # print("%6d %15.2f %13.3f\n" % (3, column3, column3 / line\_count), end='') |
|  |  |
|  | # print(f""" |
|  | # File: {myfile} |
|  | # Count: {line\_count} |
|  | # Total Column 1: {column1} |
|  | # Average: {column1 / line\_count} |
|  | # Total Column 2: {column2} |
|  | # Average: {column2 / line\_count} |
|  | # Total Column 3: {column3} |
|  | # Average: {column3 / line\_count} |
|  | # """) |
|  |  |
|  |  |
|  | main() |