# Code for testing the Polya conjecture (just change the "n=..." below)
# Author: Chris Connell 10-3-21

from sympy.ntheory import factorint
import time  # FOR TIMING HOW LONG SOMETHING TAKES
import matplotlib.pyplot as plt
import numpy as np

# Number of prime factors counted with multiplicities
def num_prime_factors1(n):
i = 2
factors = 0
while i * i <= n:
    if n % i:
i += 1
    else:
n //= i
    factors += 1
if n > 1:
factors += 1
return factors

# This does the same thing as above but uses a package optimized for speed.
def num_prime_factors(n):
    return sum(factorint(n).values())

# Number of numbers at most n with an odd number of prime factors (counted with multiplicities)
def num_odd_factors(n):
t0 = time.process_time()
i=2
num_odd = 0
while i<n:
    if num_prime_factors(i) % 2:
        num_odd += 1
    i += 1
# Put the timer in here and return it too.
fraction = num_odd/n
elapsed_time = time.process_time() - t0

return [num_odd, fraction, elapsed_time]
n=10000

print("There are ", num_odd_factors(n)[0], \\
" numbers less than ", n, ".\n", sep="")
print("The fraction of numbers less than ", n, " with \\
an odd number of prime factors is ", num_odd_factors(n)[1], ".\n", sep="")
print("The time it took was ", num_odd_factors(n)[2], ".\n", sep="")

#Eric Rasmusen's supplemental code follows.
numbers = [10, 20, 30, 40, 50, 60, 100, 200, 250, 300, 400, 500 ]
#numbers = [10,100,1000,10000,100000,1000000,2000000]
#numbers= [5000000,10000000]
#numbers = [10,20,40,80,10,100,1000,1e+6,1e+9]
print("The numbers to be tested are: ", numbers,".\n", sep="")
times=[]
fractions=[]
for item in numbers:
    fractions.append(num_odd_factors(item)[1])
    times.append(num_odd_factors(item)[2])

#How do I convert integers to exponential form if over a certain level?

#Label the horizontal and vertical axes and give a title.
plt.xlabel("The Number $N$")
plt.ylabel("Fraction Odd", rotation= "horizontal", horizontalalignment="left", labelpad=6)
plt.title("The Polya Conjecture", color="red", fontsize = 20 )
plt.plot(numbers, fractions,linestyle="dotted", marker= ".")
plt.show()
plt.plot(numbers, times,linestyle="dotted", marker= ".")