```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Dec 17 16:35:34 2018

@author: leasantos
"""

# Fourier Transform are used to analyze, smooth or filter signals and functions

# We can write a periodic function as a Fourier series
# If the function is symmetric, we can use a cosine series
# f(x) = sum_k a_k cos(2*pi*k*x/L)
# where the frequency of each cosine is k/L

# With the Fourier transform we can identify the
# dominant frequencies

# See more in the PDF file Lec14_notes01.pdf


#%%

# The function dft below gives as the coefficients c's for the function y

# NOTE that it gives ONLY
#                       N/2 + 1 c's for EVEN y  (from c_0 to c_N/2)
# and
#                      (N+1)/2 c's for ODD y  (from c_0 to c_{(N+1)/2} )

# so the loop below goes from k=0 to k=N//2 + 1

# NOTE: // is the integer division operator (it rounds DOWN the integer result)

# NOTE: cmath is used instead of math, so that exp can handle complex numbers


import numpy as np
import cmath



def dft(y):
    Ntot = len(y)
    cc = np.zeros(Ntot//2 + 1, complex )
#    cc = np.zeros(Ntot, complex )
    for k in range(Ntot//2 + 1):
#    for k in range(Ntot):
        for n in range(Ntot):
            cc[k] = cc[k] + y[n]*cmath.exp(-2j*cmath.pi*k*n/Ntot)
    return cc
```

```
#%%

#          AN EXAMPLE
# UNDERSTANDING THE FOURIER TRANSFORM

# We are given below a sine function "y" with period = 50,
# therefore frequency = 1/50 = 0.02
# The function is Sin[4.Pi.x/100]

# We are given N=100 samples for y_n
# After the discrete Fourier transform (dft) of "y", we get that
# the largest c_k happens at k = 2,
# so the frequency is indeed k/N = 2/100 = 0.02
# and the period is N/k = 100/2 = 50


import numpy as np
import cmath
import math
import matplotlib.pyplot as plt
from numpy.fft import rfft


def dft(y):
    Ntot = len(y)
    cc = np.zeros(Ntot//2 + 1, complex )
#    cc = np.zeros(Ntot, complex )
    for k in range(Ntot//2 + 1):
#    for k in range(Ntot):
        for n in range(Ntot):
            cc[k] = cc[k] + y[n]*cmath.exp(-2j*cmath.pi*k*n/Ntot)
    return cc

# The function is Sin[4.Pi.x/100]
# Study the function: what is the period? What is the frequency?
# We could then write the function as
#           Sin[2.Pi.f.x]
#               or
#           Sin[2.Pi.x/T]

nt=100
x = np.arange(0, nt, 1)
y = np.sin(4*math.pi*x/nt)
# AXES has [left, bottom, width, height]
plt.axes([0.05, 0.3, 0.4, 0.4])
plt.plot(y)
plt.title('y=sin(4*pi*x/100),   T=50')
```

```
# The function can be written as
#       (1/N) sum_k c_k exp(i 2.pi.k.n/N)
#               so each f_k = k/N

# NOTE: We plot the absolute value of the c's,
# because they are complex values.
c = dft(y)
plt.axes([0.55, 0.6, 0.4, 0.3])
plt.plot(abs(c),'b-', label = 'dft')
plt.title('f=k/N=2/100 or T=N/2=100/2')
plt.axis([0, 3, -1.5, 60])
plt.xlabel("k")
plt.ylabel("c")
plt.legend()

# -----  UNITS ------
# We need to be given what N=100 corresponds to in seconds.
# Suppose N=100 corresponds to 0.3.s,
# in other words, the interval between two points is 0.3/100.
# Then the period is (N/k)*(0.3/N) seconds
# And the frequency is (k/N)*(N/0.3) Hz


# -----  FAST FOURIER TRANSFORM of REAL FUNCTION ------
plt.axes([0.55, 0.2, 0.4, 0.3])
cfast = rfft(y)
plt.plot(abs(cfast), 'r--',label = 'rfft')
plt.axis([0, 3, -1.5, 60])
plt.xlabel("k")
plt.ylabel("c")
plt.legend()

plt.show()


#%%
# EXAMPLE from the book

# Estimate the frequency by looking at the function.
# Then confirm it by studying the DFT

import numpy as np
import cmath
import matplotlib.pyplot as plt


def dft(y):
    Ntot = len(y)
    cc = np.zeros(Ntot//2 + 1, complex )
    for k in range(Ntot//2 + 1):
        for n in range(Ntot):
```

```
            cc[k] = cc[k] + y[n]*cmath.exp(-2j*cmath.pi*k*n/Ntot)
    return cc

y = np.loadtxt("Lec14_pitch.txt",float)


# AXES has [left, bottom, width, height]
plt.axes([0.05, 0.2, 0.4, 0.4])
plt.plot(y)
plt.axis([0, 200, -1.5, 1.5])
plt.title('y - function')


c = dft(y)


plt.axes([0.55, 0.2, 0.4, 0.4])
plt.plot(abs(c))
#plt.axis([0, 50, 0, 600])
plt.xlabel("k")
plt.ylabel("c")
plt.title('Main Frequency (1st) and Harmonics')

plt.show()

print("The main frequency can be estimated from the y-plot: T~", 64," and f~",
 1/64)
print("It can be obtained directly from the c-plot,")
print("by identifying the k corresponding to the largest c, which is ~",16)
print("and dividing this k by Ntot = length of y, which is", len(y))
print("So from the c-plot, f~", 16./len(y))
```