```python
# -*- coding: utf-8 -*-
"""
Created on Tue Jun 13 12:02:40 2023

@author: marte
"""

import numpy as np
import matplotlib.pyplot as plt

"""
Wassuup Marten back at it again met de P&D uitwerkingen :)
"""

# Exercise 1
"""
A = 3
B = 4
C = 5
D = 2
E = 1
"""

# Exercise 2
# a
x = np.random.normal(0, 1, (100,200))

# b
y = np.array([np.sum(x[i,:]) for i in range(len(x))])

# c
y1 = y[:50]
y2 = y[50:]

# Exercise 3
# a
kwadraat = [i**2 for i in range(1,26)]

# b
kwadraat2 = [i for i in kwadraat if 200<i<400]


# Exercise 4
def second_order(wt, zeta):
    y = 1 - np.exp(-zeta*wt)* (np.cos(wt*np.sqrt(1-zeta**2)) +
                               zeta/(np.sqrt(1-zeta**2))*np.sin(wt*np.sqrt(1-zeta**2)))
    return y

wt = np.linspace(0,20,1000)
plt.figure()
plt.plot(wt, second_order(wt, 0.2),label=r'$\zeta$ = 0.2', )
plt.plot(wt, second_order(wt, 0.7),'--', label=r'$\zeta$ = 0.7')
plt.plot(wt, second_order(wt, 0.9), ':r', label=r'$\zeta$ = 0.9')
plt.ylim([0,2])
plt.xlim([0,20])
plt.grid()
plt.legend()
plt.xlabel(r'$\omega_n$t')
plt.ylabel('y(t)')
plt.show()


# Exercise 5
# a
def taxicab(p1,p2):
    s = np.abs(p2[1]-p1[1]) + np.abs(p2[0]-p1[0])
    return s

# Check:
p1 = np.array([-2,1])
p2 = np.array([3,4])
print(taxicab(p1,p2))

# b
route = np.loadtxt('route.csv', delimiter=',')
```

```
# c
length = 0
for i in range(len(route[:,0])-1):
    length += taxicab(route[i,:], route[i+1,:])


# Exercise 6
def Madelung(m,n):
    alpha = 0                      # Start with alpha at zero
    for i in range(1,m+1):         # Sum over values of m
        if i%2==1:                 # Only take the odd values of m
            for j in range(1,n+1): # Sum over values of n
                if j%2==1:         # Only take the odd values of n
                    alpha += np.cosh(np.pi/2 * (np.sqrt(i**2 + j**2)) )**(-2)
                                   # Add the calculated value to alpha
    alpha = alpha * 12 * np.pi     # Add the 12pi prefactor
    return alpha
# Check calculated value:
print(Madelung(13,13))
# Calculated value is almost 1.7476 as expected


# Exercise 7
def lengthlongestincreasing(x):
    longest = 1  # Starting longest series is 1
    count = 1    # Counter of how long the current series is

    for i in range(0,len(x)-1): # Go through the array from left to right
        if x[i+1] > x[i]:       # Check if the series is still increasing
            count += 1          # If yes: update counter

        elif count > longest:  # If not increasing and the current series is
                               # longer than the longest series, update longest
            longest = count    # Update the longest series
            count = 1          # Reset the counter

    if count > longest:        # After the entire for-loop is done last-check
        longest = count        # if the counter is higher than longest
                               # if that's the case: update longest
    return longest             # this is important for the case that we have
                               # a constantly increasing series and the elif
                               # statement is not

xtest = [23,2,3,4,88,-1,2,3] # Should give 4
xtest2 = [1,2,3,4,5,6,7,8,9] # Should give 9
xtest3 = [1,2,3,4,5,6,7,6,8,9,10,11,12,13,14,15,16,-3,4] # Should give 10
print(lengthlongestincreasing(xtest))
print(lengthlongestincreasing(xtest2))
print(lengthlongestincreasing(xtest3))
```