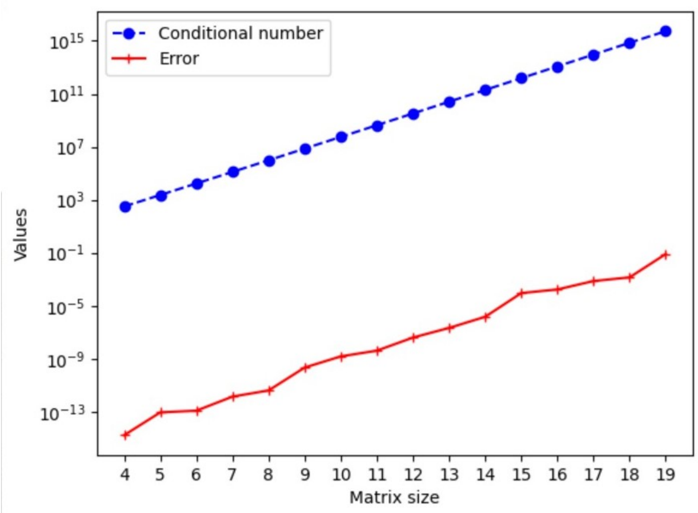
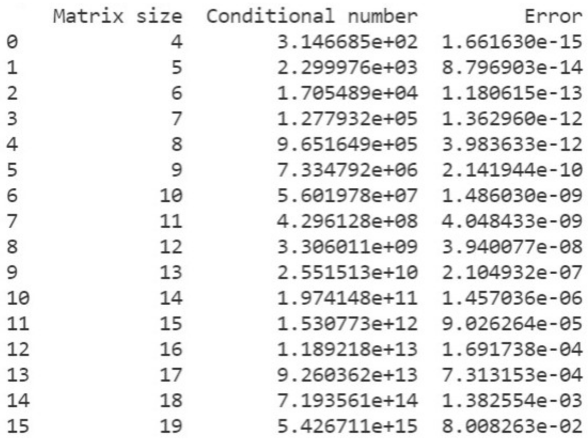
Q3

The code is written in Python

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

def Van_mat_gen(n):
    """Creates the Vandermonde matrix of
    of the inputter size"""
    x = 1/n
    initial  = 1/n
    matrix_list = []
    mult = 1
    x_list = []
    for j in range(n):
        x = mult*initial
        x_list.append(x)
        mult += 1

    for x in x_list:
        row = [1]
        for e in range(1, n):
            row.append(x**e)
        matrix_list.append(row)
    return np.array(matrix_list)

def cond_num(matrix):
    """Calculates the conditional
    number of the matrix"""
    inverse = np.linalg.inv(matrix)
    normA = np.linalg.norm(matrix, ord=2)
    norm_invA = np.linalg.norm(inverse, ord=2)
    return norm_invA*normA

def solver(matrix):
    """Solves the matrix for some
    vector for which the solution is [1,1..1]^T"""
    output = []
    for e in matrix:
        output.append(sum(e))
    output_vect = np.transpose(output)
    solution = np.linalg.solve(matrix, output_vect)
    return solution

def relative_error(exp, act):
    """Calculates the rellative error of a vector
     as ||expected-actual||/||actual||"""
    numerator = np.linalg.norm(act-exp, ord=2)
    denom = np.linalg.norm(act, ord=2)
    return numerator/denom

def constructor(Van=Van_mat_gen, cond=cond_num, solver=solver, error=relative_error):
    data = []
    x_val = list(range(4, 20))
    cond_list = []
    error_list =[]
    for i in range(4, 20):
        matrix = Van(i)
        act = solver(matrix)
        exp = np.array([1]*i)
        error_num = error(exp, act)
        cond_num = cond(matrix)

        cond_list.append(cond_num)
        error_list.append(error_num)
        data.append([i, cond_num, error_num])

    df = pd.DataFrame(data, columns = ['Matrix size', 'Conditional number', 'Error'])

    x = np.array(x_val)
    y = np.array(cond_list)
    z = np.array(error_list)
    default_x_ticks = range(len(x))
    plt.plot(default_x_ticks, y, color='blue', linestyle='dashed', marker='o')
    plt.plot(default_x_ticks, z, color='red', linestyle='solid', marker='+')
    plt.xticks(default_x_ticks, x)
    plt.xlabel('Matrix size')
    plt.ylabel('Values')
    plt.legend(['Conditional number', 'Error'])
    plt.yscale("log")
    plt.show()

    return df

print(constructor())
```





```
    Matrix size  Conditional number         Error
0             4        3.146685e+02  1.661630e-15
1             5        2.299976e+03  8.796903e-14
2             6        1.705489e+04  1.180615e-13
3             7        1.277932e+05  1.362960e-12
4             8        9.651649e+05  3.983633e-12
5             9        7.334792e+06  2.141944e-10
6            10        5.601978e+07  1.486030e-09
7            11        4.296128e+08  4.048433e-09
8            12        3.306011e+09  3.940077e-08
9            13        2.551513e+10  2.104932e-07
10           14        1.974148e+11  1.457036e-06
11           15        1.530773e+12  9.026264e-05
12           16        1.189218e+13  1.691738e-04
13           17        9.260362e+13  7.313153e-04
14           18        7.193561e+14  1.382554e-03
15           19        5.426711e+15  8.008263e-02
```

As we can see from the graph, both the conditional number and the relative error increase exponentially (even thought the graphs look linear, the scale on the y-axis is logarithmic). And the larger the size of the matrix, the less the accurate the computation of the solution and the higher the conditional number.