Connected to base (Python 3.12.5)

```
In [ ]: """Requirements:

        Python script using Pandas for descriptive statistics
        Read a dataset (CSV or Excel)
        Generate summary statistics (mean, median, standard deviation)
        Create at least one data visualization
        Deliverable:

        Python script
        Generated summary report (PDF or markdown)"""

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


        df = pd.read_csv(
            "https://raw.githubusercontent.com/anlane611/datasets/main/population.csv"
        )

        df_y = df["Y"]
        print(df_y)
```

```
0        21.973610
1        12.387638
2        12.665114
3        16.753335
4        22.435229
           ...
99995    23.310289
99996    20.406937
99997    25.335073
99998    29.479947
99999    16.473850
Name: Y, Length: 100000, dtype: float64
```

```
In [ ]: # calculate the mean of variable Y:
        def get_mean(script):
            sum_script = 0
            for i in range(len(script)):
                sum_script += script[i]

            mean_script = sum_script / len(script)
            return round(mean_script, 3)


        print(get_mean(df_y))
```

```
19.976
```

```
In [ ]: # calculate the median of variable Y:
        def get_median(script):
            sorted_script = np.sort(script)
            num = len(sorted_script)
            if num % 2 == 0:
                median_script = (sorted_script[num // 2] + sorted_script[num // 2 + 1]) / 2
            else:
                median_script = sorted_script[np.ceil(num / 2)]
            return round(median_script, 3)


        print(get_median(df_y))
```

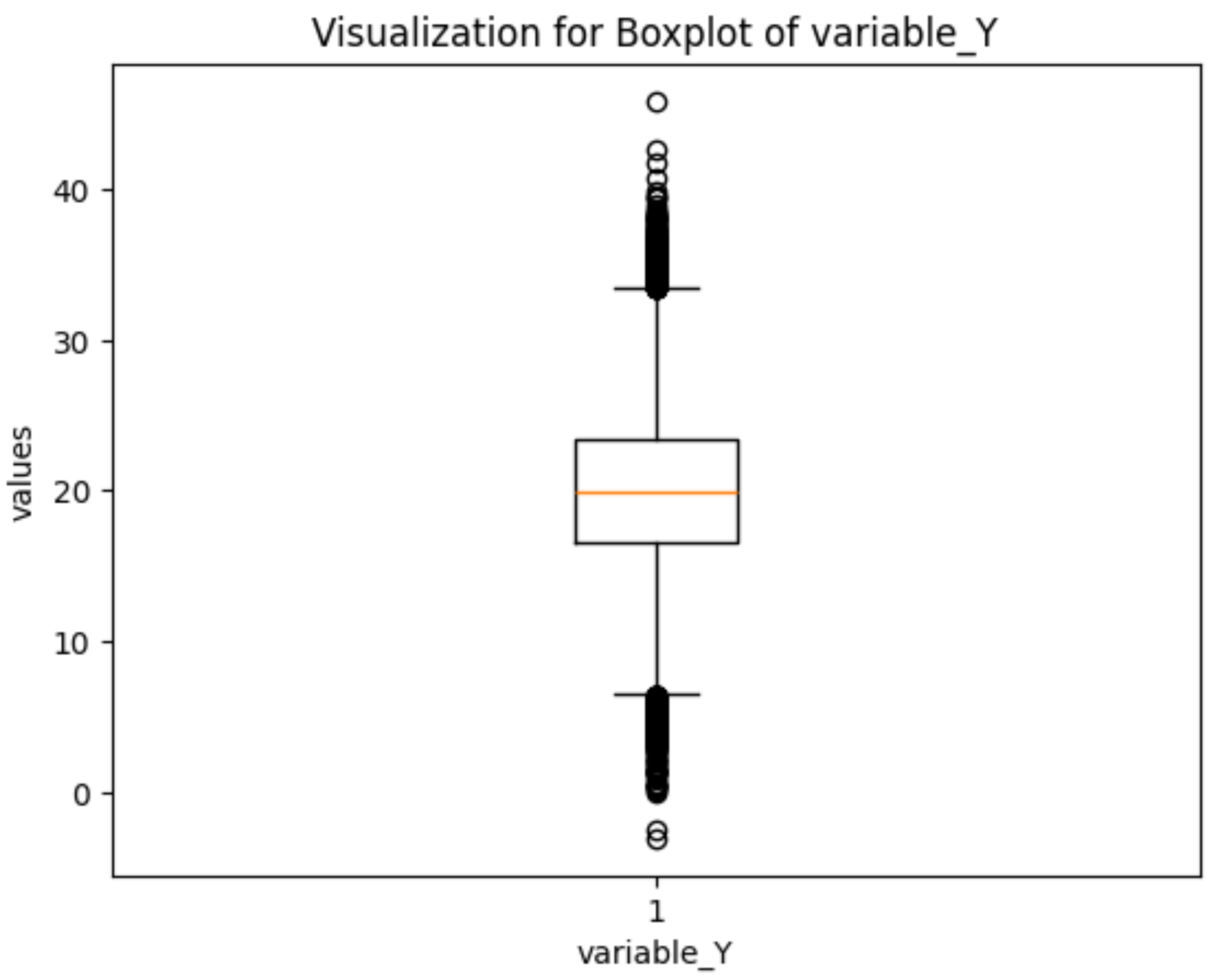
```
19.971
```

```
In [ ]: # calculate the standard deviation of variable Y:
        def get_std(script):
            num = len(script)
            mean_s = np.mean(script)
            sum_error = 0
            for i in range(num):
                sum_error += (mean_s - script[i]) ** 2
            std_script = np.sqrt(sum_error / (num - 1))
            return round(std_script, 3)


        print(get_std(df_y))
```

```
5.005
```

```
In [ ]: # data visualization: boxplot for variable Y"
        plt.boxplot(df_y)
        plt.xlabel("variable_Y")
        plt.ylabel("values")
        plt.title("Visualization for Boxplot of variable_Y")
        plt.show()
```


Visualization for Boxplot of variable_Y
values
variable_Y