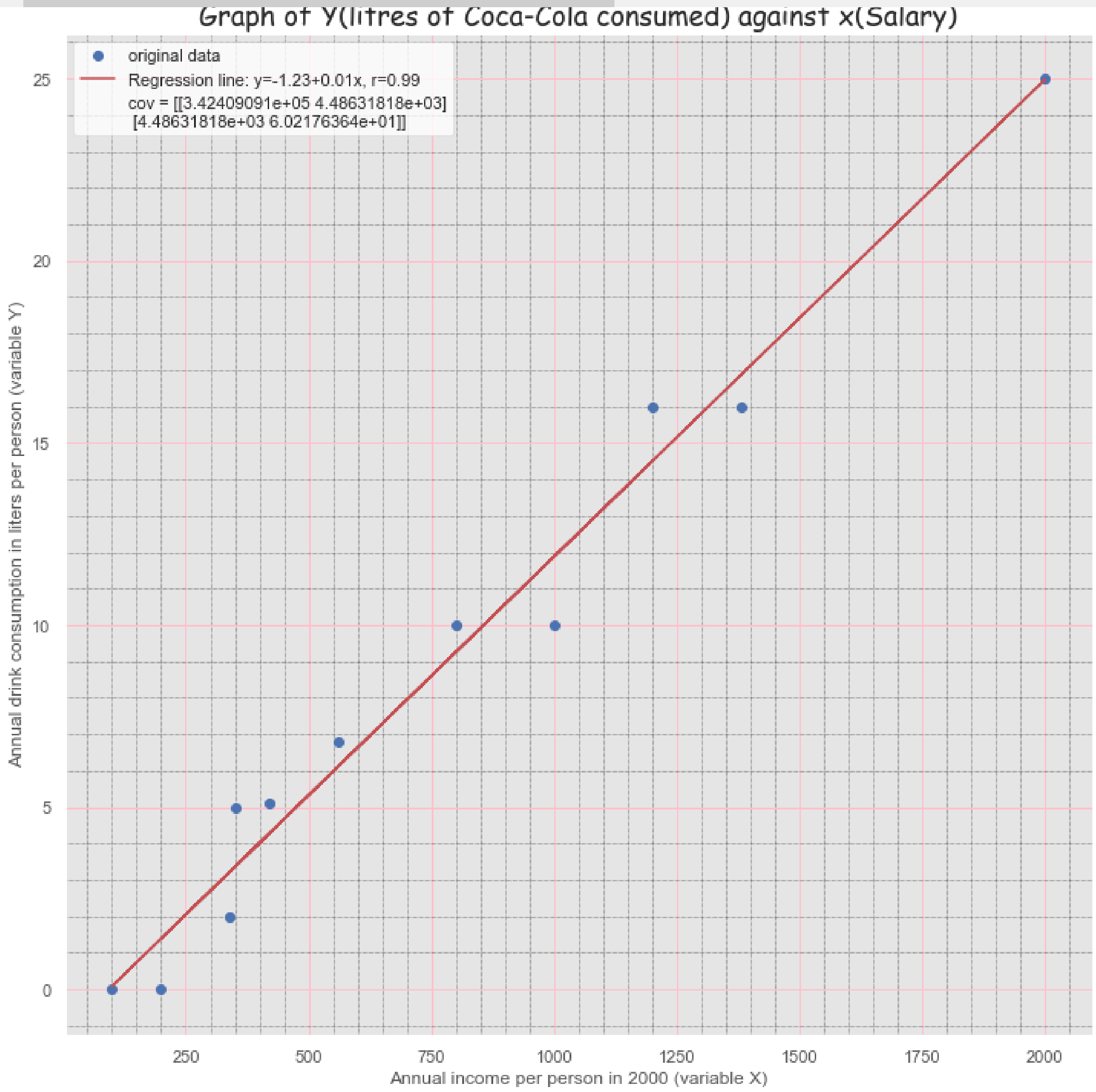
In [66]:

```
#import libraries
import re #use for multilines
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import scipy
from scipy import stats


#interpretation of result
txt=""" The supposition that the consumption of Coca-Cola is higher the higher th
of the correlation coefficient is positive, this implies that as X increases so c
This implies a very strong relationship and X and Y are nearly perfectly linear.
change in positive manner and the regression coefficient indicates that a mean in
unit increase in the mean of Y (litres of Coca-Cola consumed)."""


#On a multiline sub, \s* will match any number of \n and any other whitespace
text= re.sub(r'\n\s*\n','\n',txt,re.MULTILINE)


#set ggplot style, provides for better outlook
plt.style.use('ggplot')

#setting up figure dimension
fig,ax = plt.subplots(figsize=(12, 12))

#Building data frame from dataset
Coke=pd.DataFrame({'X':[350,420,200,800,1200,560,100,2000,1380,1000,340],'Y':[5,!

#statistica regression calculation using scipy
slope, intercept, r_value, p_value, std_err = stats.linregress(Coke['X'], Coke['Y

#covariance calculation
cov = np.cov(Coke['X'], Coke['Y'])


#regression equation printing
fitted_line = f'Regression line: y={intercept:.2f}+{slope:.2f}x, r={r_value:.2f}

#covariance printing
covariance = f'cov = {cov}'

#plotting of data and presentation of statistical results
ax.plot(Coke['X'], Coke['Y'], 'bo', label='original data')
ax.plot(Coke['X'], intercept + slope*Coke['X'], 'r', label=fitted_line)
ax.plot([], [], ' ', label=covariance) #setting up label for covariance result on

#setting up title, axes label, and legend
ax.set_title('Graph of Y(litres of Coca-Cola consumed) against x(Salary)',fontnar
ax.set_ylabel('Annual drink consumption in liters per person (variable Y)')
ax.set_xlabel('Annual income per person in 2000 (variable X)')
fig.text(.5, 0.00005, text, ha='center')

ax.legend(facecolor='white',loc='upper left')
```

```
# Turn on the minor TICKS, which are required for the minor GRID
ax.minorticks_on()

# Customize the major grid
ax.grid(which='major', linestyle='-', linewidth='1', color='pink')
# Customize the minor grid
ax.grid(which='minor', linestyle=':', linewidth='0.5', color='black')

plt.show()
#plt saving and saving in very high quality
fig.savefig('q1.png', format='png', dpi=1200)
```

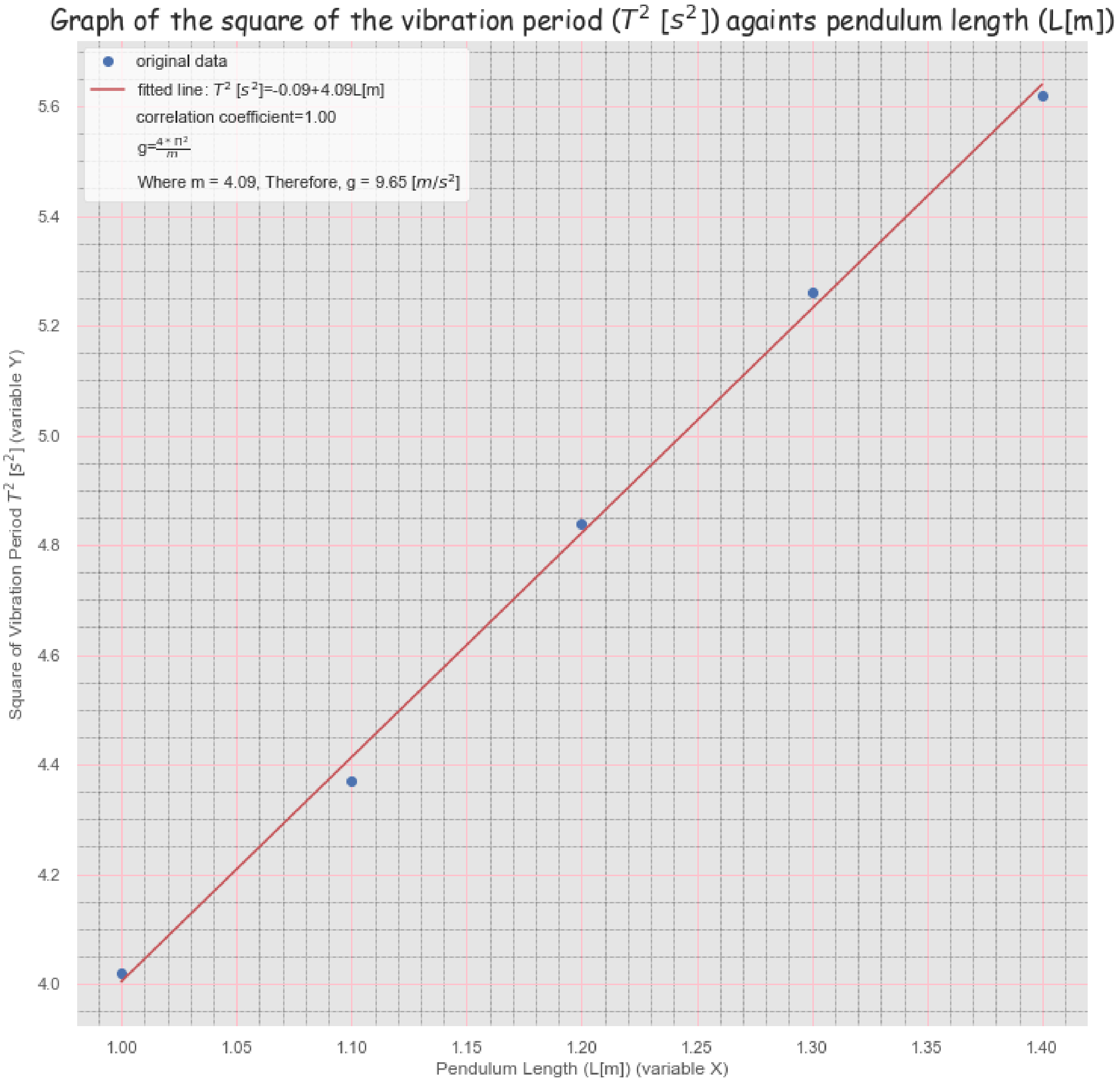


The supposition that the consumption of Coca-Cola is higher the higher the income of the population, is true. This is because the sign of the correlation coefficient is positive, this implies that as X increases so does. Also, the vale of the correlation is 0.99. This implies a very strong relationship and X and Y are nearly perfectly linear. Futhermore, the covariance shows that X and Y change in positive manner and the regression coefficient indicates that a mean increase of 0.01 in X (Salary) will result to a unit increase in the mean of Y (litres of Coca-Cola consumed).

In [67]:

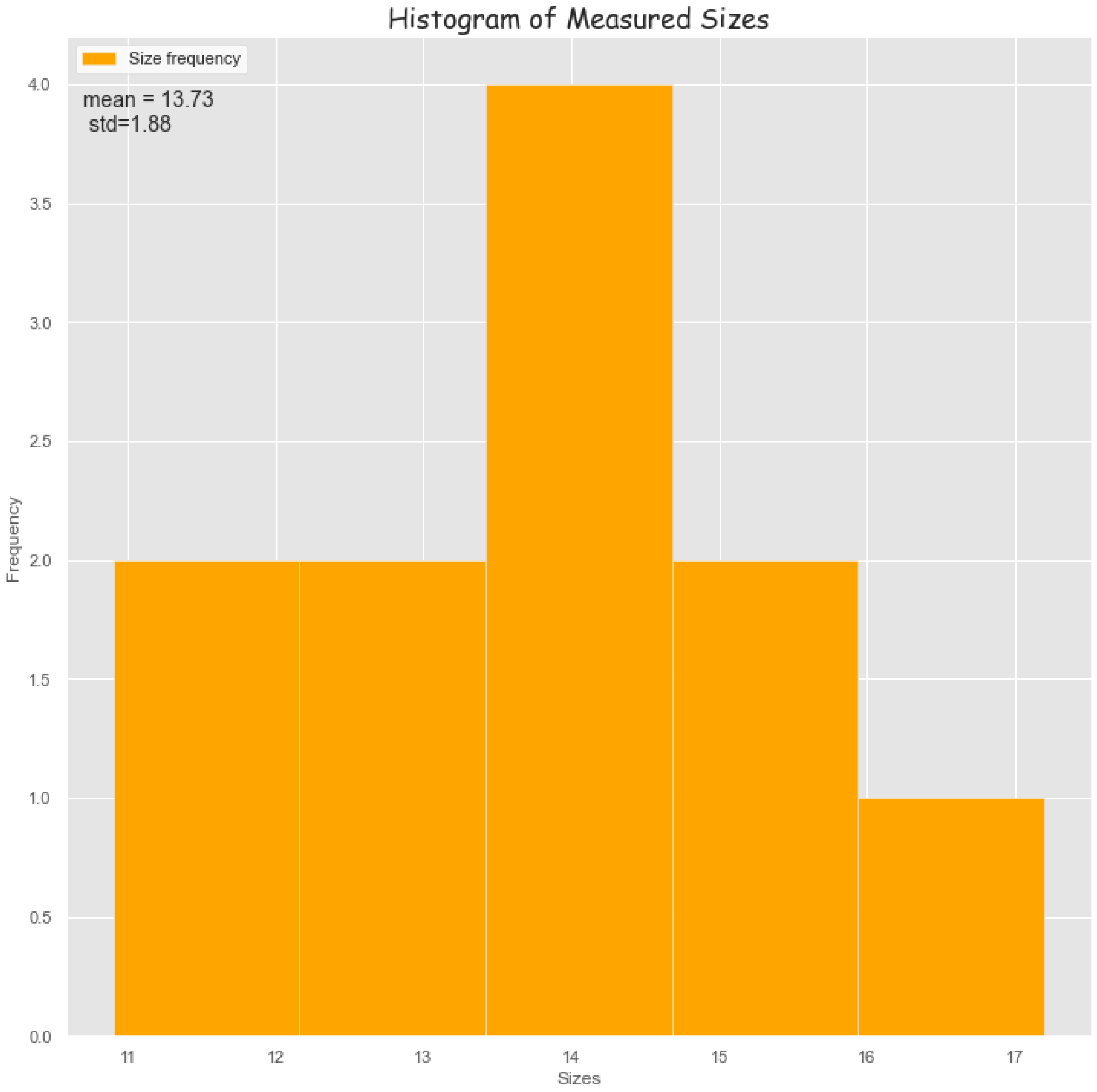
```
#import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from numpy import pi
import scipy
from scipy import stats

#set ggplot style
plt.style.use('ggplot')

#setting up figure
fig,ax = plt.subplots(figsize=(12, 12))

#Building data frame
Pendulum=pd.DataFrame({'X':[1.0, 1.10, 1.20, 1.30, 1.40], 'Y':[4.02, 4.37, 4.84,

#statistica calculation
slope, intercept, r_value, p_value, std_err = stats.linregress(Pendulum['X'], Pe

##fitted line and it's equation
fitted_line = f'fitted line: $T^{2}$ [$s^{2}$]={intercept:.2f}+{slope:.2f}L[m]'


##correlation coefficient printing
correlaion_coef = f'correlation coefficient={r_value:.2f}'

#calculating the gravitational acceleration
g=(4*(np.pi)**2)/slope
gra= f'Where m = {slope:.2f}, Therefore, g = {g:.2f} [$m/s^{2}$]'

#plotting of data and presentation of statistical results
ax.plot(Pendulum['X'], Pendulum['Y'], 'bo', label='original data')
ax.plot(Pendulum['X'], intercept + slope*Pendulum['X'], 'r', label=fitted_line)
ax.plot([], [], ' ', label=correlaion_coef) #label for correlation coefficient
ax.plot([], [], ' ',label=r'g=$\frac{4*\Pi^{2}}{m}$') #label for calculating gra
ax.plot([], [], ' ', label=gra) #label for gravity calculation answer

#setting up title, axes, and legend
ax.set_title('Graph of the square of the vibration period ($T^{2}$ [$s^{2}$]) ag
             fontname='Comic Sans MS', fontsize=18)
ax.set_ylabel('Square of Vibration Period $T^{2}$ [$s^{2}$] (variable Y)')
ax.set_xlabel('Pendulum Length (L[m]) (variable X)')

ax.legend(facecolor='white',loc='upper left')

# Turn on the minor TICKS, which are required for the minor GRID
ax.minorticks_on()

# Customize the major grid
ax.grid(which='major', linestyle='-', linewidth='1', color='pink')
# Customize the minor grid
ax.grid(which='minor', linestyle=':', linewidth='0.5', color='black')

plt.show()
```

```
# Do the plot code saving in very high quality
fig.savefig('q2.png', format='png', dpi=1200)
```



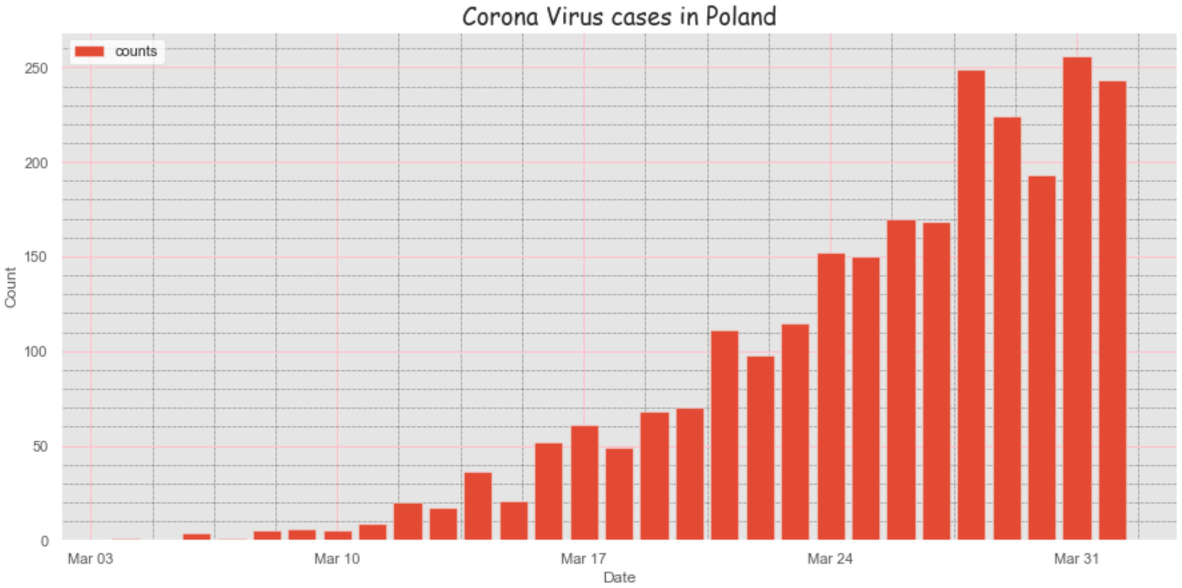
In [68]:

```
#import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from numpy import pi
import scipy
from scipy import stats

#set ggplot style
plt.style.use('ggplot')

#using seaborn to make output nicer
sns.set()

#setting up figure
fig,ax = plt.subplots(figsize=(12, 12))

#building up dataframe from dataset
d = pd.DataFrame({'X':[13.2, 13.9, 13.65, 14.1, 12.3, 15.6, 17.2, 14.3,14.9, 10.
mean=d['X'].mean()
std=d['X'].std()

#making histogram
ax.hist(d['X'], color=['orange'],bins=5)

#setting up legend, axes and title
plt.title('Histogram of Measured Sizes',fontname='Comic Sans MS', fontsize=18)
legend = ['Size frequency']
plt.xlabel("Sizes")
plt.ylabel("Frequency")
plt.legend(legend,facecolor='white',loc='upper left')

#printing results of statistica calculation on graph
plt.text(10.7, 3.8, f'mean = {mean:.2f} \n std={std:.2f}',fontsize=14)

#saveing image in high quality
fig.savefig('q3.png', format='png', dpi=1200)
```


Histogram of Measured Sizes
Size frequency
mean = 13.73
std=1.88
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Frequency
11
12
13
14
15
16
17
Sizes

In [74]:

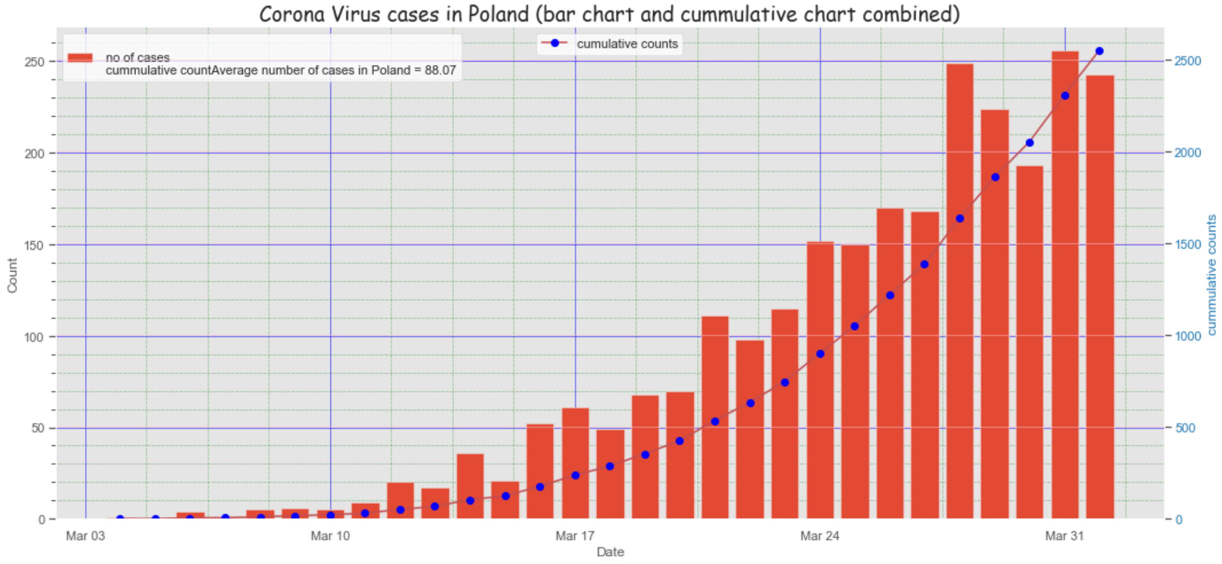
```
#import libraries
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates

#set ggplot style
plt.style.use('ggplot')

#read data from csv
korona = pd.read_excel ("D:\Korona_PL.xlsx", usecols=['data','no of cases'], par
#set date as index
korona.set_index('data',inplace=True)

#plot data
fig, ax = plt.subplots(figsize=(15,7))
ax.bar(korona.index, korona['no of cases'])

#setting up legend, title and axes lable
ax.set_title('Corona Virus cases in Poland',fontname='Comic Sans MS', fontsize=1
ax.set_ylabel('Count')
ax.set_xlabel('Date')

legend = ['counts']
ax.legend(legend,facecolor='white',loc='upper left')

# Turn on the minor TICKS, which are required for the minor GRID
ax.minorticks_on()

# Customize the major grid
ax.grid(which='major', linestyle='-', linewidth='1', color='pink')
# Customize the minor grid
ax.grid(which='minor', linestyle=':', linewidth='0.5', color='black')

#set ticks every week
ax.xaxis.set_major_locator(mdates.WeekdayLocator())
#format date
ax.xaxis.set_major_formatter(mdates.DateFormatter('%b %d'))


plt.show()

#saving in high quality
fig.savefig('q4a.png', format='png', dpi=1200)
```


Corona Virus cases in Poland
counts
Count
0
50
100
150
200
250
Mar 03
Mar 10
Mar 17
Mar 24
Mar 31
Date

In [75]:

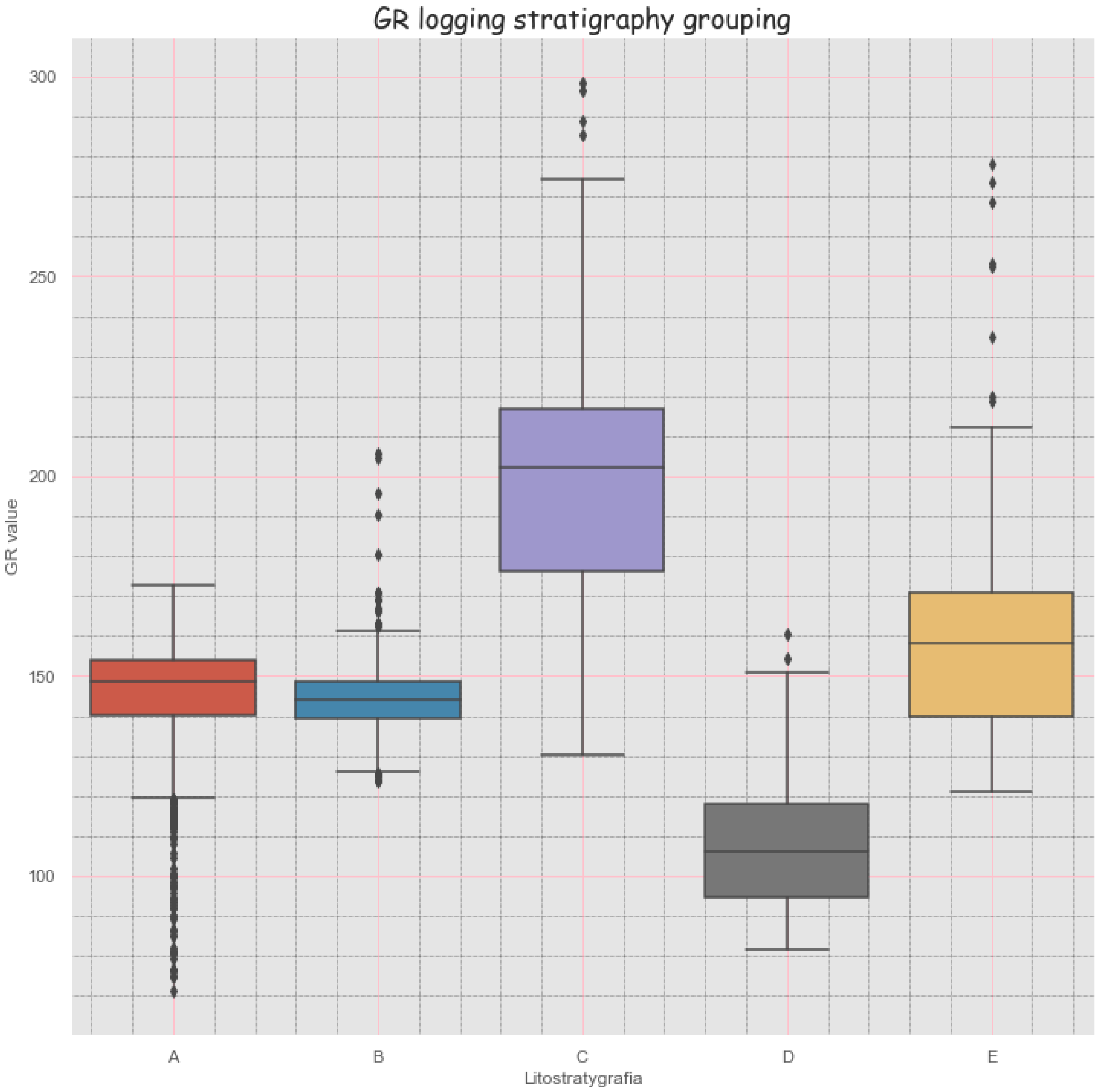
```
#import libraries
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates

#set ggplot style
plt.style.use('ggplot')

#read data from csv
korona = pd.read_excel ("D:\Korona_PL.xlsx", usecols=['data','no of cases'], pars
#set date as index
korona.set_index('data',inplace=True)

#cumylative sum
korona['cumulative counts']=korona['no of cases'].cumsum()

#mean
mean = korona['no of cases'].mean()

#plot data
fig, ax = plt.subplots(figsize=(15,7))
ax.bar(korona.index, korona['no of cases'],label='no of cases')

#setting up tile, legend and labels
ax.set_title('Corona Virus cases in Poland (bar chart and cummulative chart comb
ax.set_ylabel('Count')
ax.set_xlabel('Date')

#in legend, the mean cases of coronavirus is printed also
leg = ax.legend([f'\n no of cases \n cummulative countAverage number of cases in

# instantiate a second axes that shares the same x-axis
ax2 = ax.twinx()

#setting up cummulative chart and properties
ax2.plot(korona.index, korona['cumulative counts'],color='r', label='cumulative 
         marker='o',markerfacecolor='blue', markeredgecolor='blue')

color = 'tab:blue' #making colour for ylable
ax2.set_ylabel('cummulative counts', color=color)  # we already handled the x-la
ax2.tick_params(axis='y', labelcolor=color)
ax2.set_ylim(bottom=0)
leg2 = ax2.legend(facecolor = 'white',loc='upper center')


fig.tight_layout()  # otherwise the right y-label is slightly clipped

# Turn on the minor TICKS, which are required for the minor GRID
ax.minorticks_on()

# Customize the major grid for bar chart
ax.grid(which='major', linestyle='-', linewidth='0.5', color='blue')
# Customize the minor grid
ax.grid(which='minor', linestyle=':', linewidth='0.5', color='green')

# Customize the major grid for cummulative chart
```

```
ax2.grid(which='major', linestyle='-', linewidth='0.5', color='pink')
# Customize the minor grid
ax2.grid(which='minor', linestyle=':', linewidth='0.5', color='green')

#set ticks every week
ax.xaxis.set_major_locator(mdates.WeekdayLocator())
#format date
ax.xaxis.set_major_formatter(mdates.DateFormatter('%b %d'))

plt.show()

#saving in high quality
fig.savefig('q4b.png', format='png', dpi=1200)
```



In [76]:

```
#import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

#set ggplot style
plt.style.use('ggplot')

#read data from csv
GRlog = pd.read_excel ("D:\GR_logging.xlsx")

#setting up plotting dimension
plt.figure(figsize=(12, 12))

#using sns to generate boxplot
plot=sns.boxplot( x=GRlog["Litostratygrafia"], y=GRlog["GR"])

#setting up title, legend and axes label
plot.set_title('GR logging stratigraphy grouping',fontname='Comic Sans MS', font
plot.set_ylabel('GR value')
plot.set_xlabel('Litostratygrafia')
leg = ax.legend('Litostratygrafia')

# Turn on the minor TICKS, which are required for the minor GRID
plot.minorticks_on()

# Customize the major grid
plot.grid(which='major', linestyle='-', linewidth='1', color='pink')
# Customize the minor grid
plot.grid(which='minor', linestyle=':', linewidth='0.5', color='black')

plt.show()

#saving in high quality
fig = plot.get_figure()
fig.savefig('q5.png', format='png', dpi=1200)
```


GR logging stratigraphy grouping
300
250
200
150
100
GR value
A
B
C
D
E
Litostratygrafia