Code Transcript

Q1.

```
import pandas as pd
df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')

df.sort_values(by='subscribers', ascending=False)
print(df.head(10))
```

Output:

	rank	Youtuber	subscribers	video views	category	...	Population	Unemployment rate	Urban_population	Latitude	Longitude
0	1	T-Series	245000000.0	2.280000e+11	Music	...	1.366418e+09	5.36	471031528.0	20.593684	78.962880
1	2	YouTube Movies	170000000.0	0.000000e+00	Film & Animation	...	3.282395e+08	14.70	270663028.0	37.090240	-95.712891
2	3	MrBeast	166000000.0	2.836884e+10	Entertainment	...	3.282395e+08	14.70	270663028.0	37.090240	-95.712891
3	4	Cocomelon - Nursery Rhymes	162000000.0	1.640000e+11	Education	...	3.282395e+08	14.70	270663028.0	37.090240	-95.712891
4	5	SET India	159000000.0	1.480000e+11	Shows	...	1.366418e+09	5.36	471031528.0	20.593684	78.962880
5	6	Music	119000000.0	0.000000e+00	NaN	...	NaN	NaN	NaN	NaN	NaN
6	7	ýýý Kids Diana Show	112000000.0	9.324704e+10	People & Blogs	...	3.282395e+08	14.70	270663028.0	37.090240	-95.712891
7	8	PewDiePie	111000000.0	2.905804e+10	Gaming	...	1.262266e+08	2.29	115782416.0	36.204824	138.252924
8	9	Like Nastya	106000000.0	9.047906e+10	People & Blogs	...	1.443735e+08	4.59	107683889.0	61.524010	105.318756
9	10	Vlad and Niki	98900000.0	7.718017e+10	Entertainment	...	3.282395e+08	14.70	270663028.0	37.090240	-95.712891

Q2.

```
import pandas as pd
df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
category_avg = df.groupby('category')['subscribers'].mean()
category_avg_sorted = category_avg.sort_values(ascending=False)

print("Category with the highest average subscribers:")
print(category_avg_sorted.head(1))
```

Output:

```
Category with the highest average subscribers:
category
Shows    4.161538e+07
```

Q3.

```
import pandas as pd

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
avg_uploads = df.groupby('category')['uploads'].mean().round(0).astype(int)
avg_uploads_sorted = avg_uploads.sort_values(ascending=False)

print("Average number of uploads per category (rounded):")
print(avg_uploads_sorted)
```

Output:

```
Average number of uploads per category
category
News & Politics            112484
Nonprofits & Activism      102912
Shows                       27444
Sports                      19130
Entertainment               12052
People & Blogs               9257
Trailers                     6839
Gaming                       4285
Pets & Animals               3563
Movies                       3553
Education                    3087
Film & Animation             2862
Music                        2326
Science & Technology         2114
Howto & Style                1696
Autos & Vehicles             1551
Comedy                       1203
Travel & Events               766
```

Q4.

```
import pandas as pd

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
top_countries = df['Country'].value_counts().head()

print("Top 5 countries with the highest number of YouTube channels:")
print(top_countries)
```

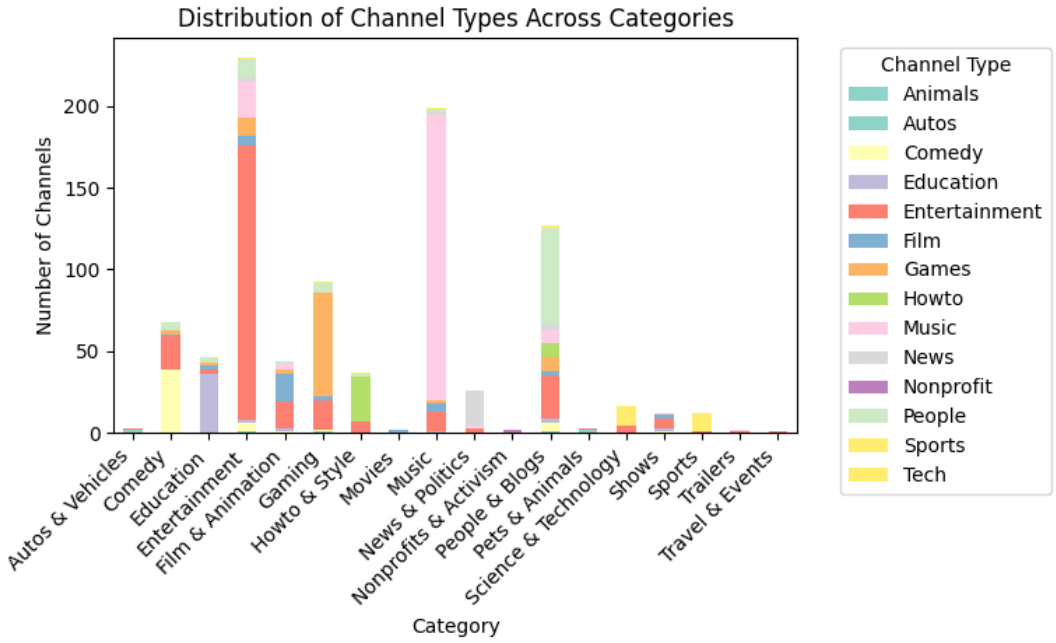
Output:

```
Top 5 countries with the highest number of YouTube channels:
Country
United States     315
India             169
Brazil             62
United Kingdom     44
Mexico             33
```

Q5.

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

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
distribution = pd.crosstab(df['category'], df['channel_type'])


plt.figure(figsize=(14, 7))
distribution.plot(kind='bar', stacked=True, colormap='Set3')

plt.title('Distribution of Channel Types Across Categories')
plt.xlabel('Category')
plt.ylabel('Number of Channels')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Channel Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()


plt.show()
```

Output:



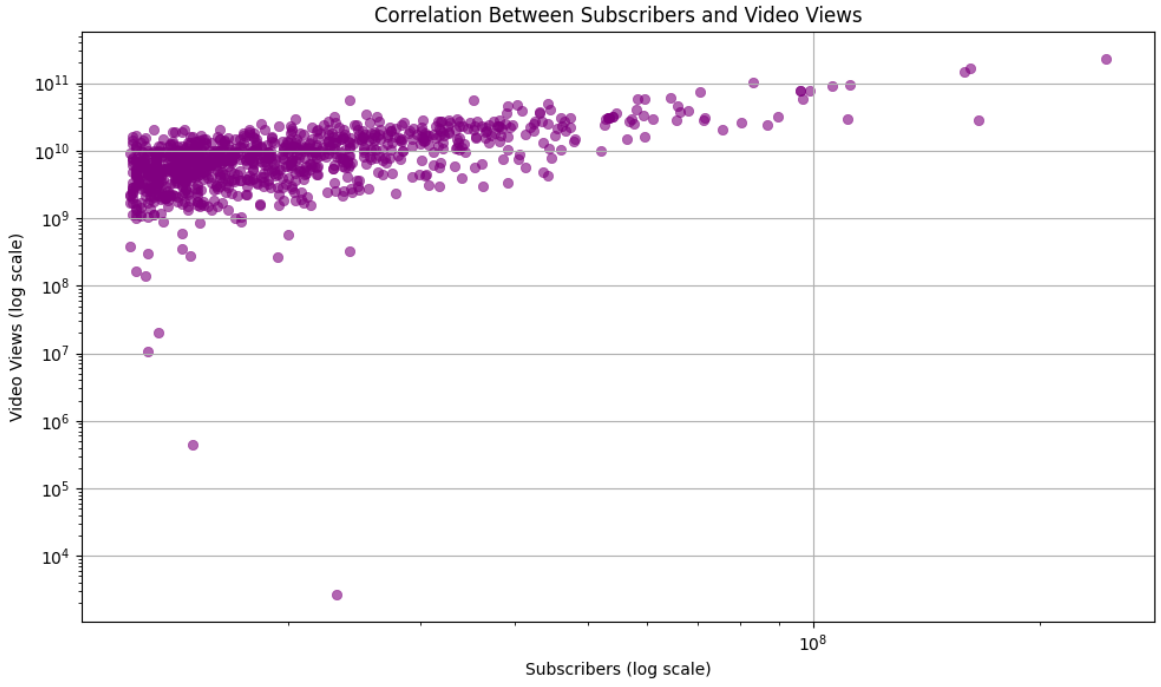
Q6.

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')

df_clean = df.dropna(subset=['subscribers', 'video views'])
df_clean = df_clean[(df_clean['subscribers'] > 0) & (df_clean['video views'] > 0)]

correlation = df_clean['subscribers'].corr(df_clean['video views'])
print(f"Correlation between subscribers and video views: {correlation:.2f}")

plt.figure(figsize=(10, 6))
sns.scatterplot(data=df_clean, x='subscribers', y='video views', alpha=0.6, color='purple', edgecolor=None)

plt.xscale('log')
plt.yscale('log')

plt.title('Correlation Between Subscribers and Video Views')
plt.xlabel('Subscribers (log scale)')
plt.ylabel('Video Views (log scale)')
plt.grid(True)
plt.tight_layout()

plt.show()
```

Output:

Strong Positive Correlation observed

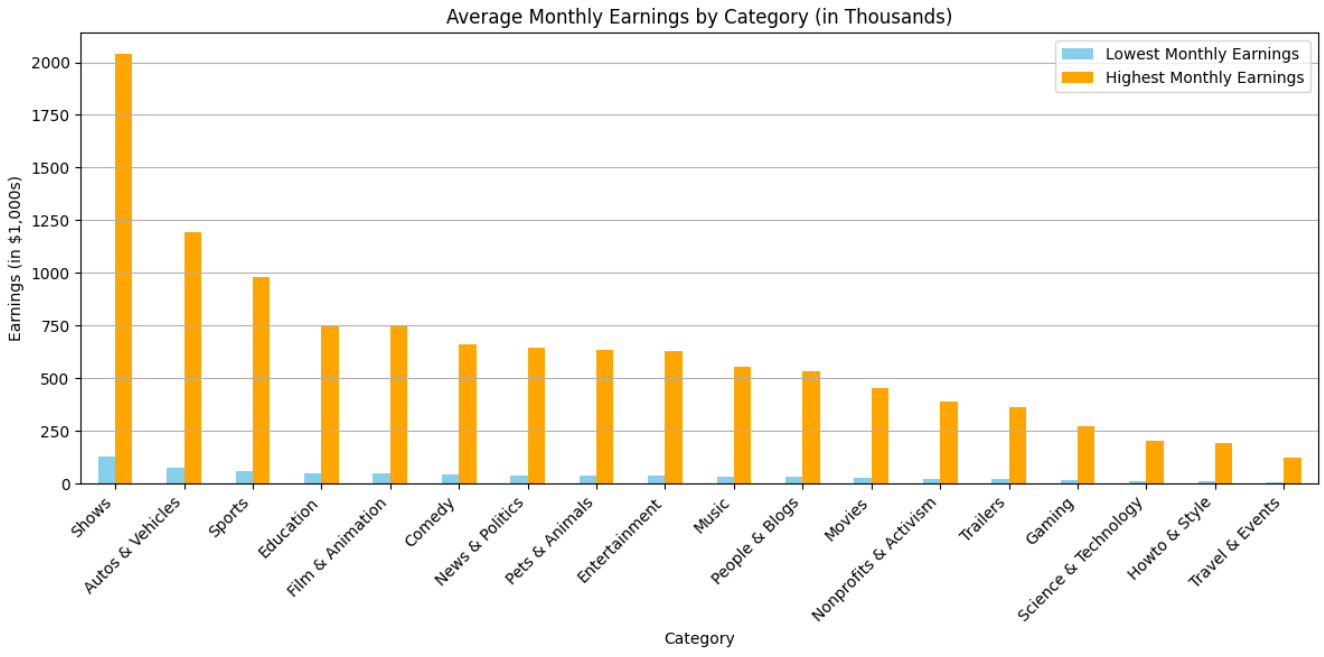
```
Correlation between subscribers and video views: 0.83
```



Q7.

```python
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df['lowest_monthly_earnings'] = pd.to_numeric(df['lowest_monthly_earnings'], errors='coerce')
df['highest_monthly_earnings'] = pd.to_numeric(df['highest_monthly_earnings'], errors='coerce')
df_clean = df.dropna(subset=['category', 'lowest_monthly_earnings', 'highest_monthly_earnings'])

monthly_earnings = df_clean.groupby('category')[['lowest_monthly_earnings', 'highest_monthly_earnings']].mean()
monthly_earnings = (monthly_earnings / 1000).round(1)  # Divide by 1000 and round to 1 decimal place
monthly_earnings_sorted = monthly_earnings.sort_values(by='highest_monthly_earnings', ascending=False)
monthly_earnings_sorted.plot(kind='bar', figsize=(12, 6), color=['skyblue', 'orange'])

plt.title('Average Monthly Earnings by Category (in Thousands)')
plt.xlabel('Category')
plt.ylabel('Earnings (in $1,000s)')
plt.xticks(rotation=45, ha='right')
plt.legend(['Lowest Monthly Earnings', 'Highest Monthly Earnings'])
plt.grid(True, axis='y')
plt.tight_layout()
plt.show()
```

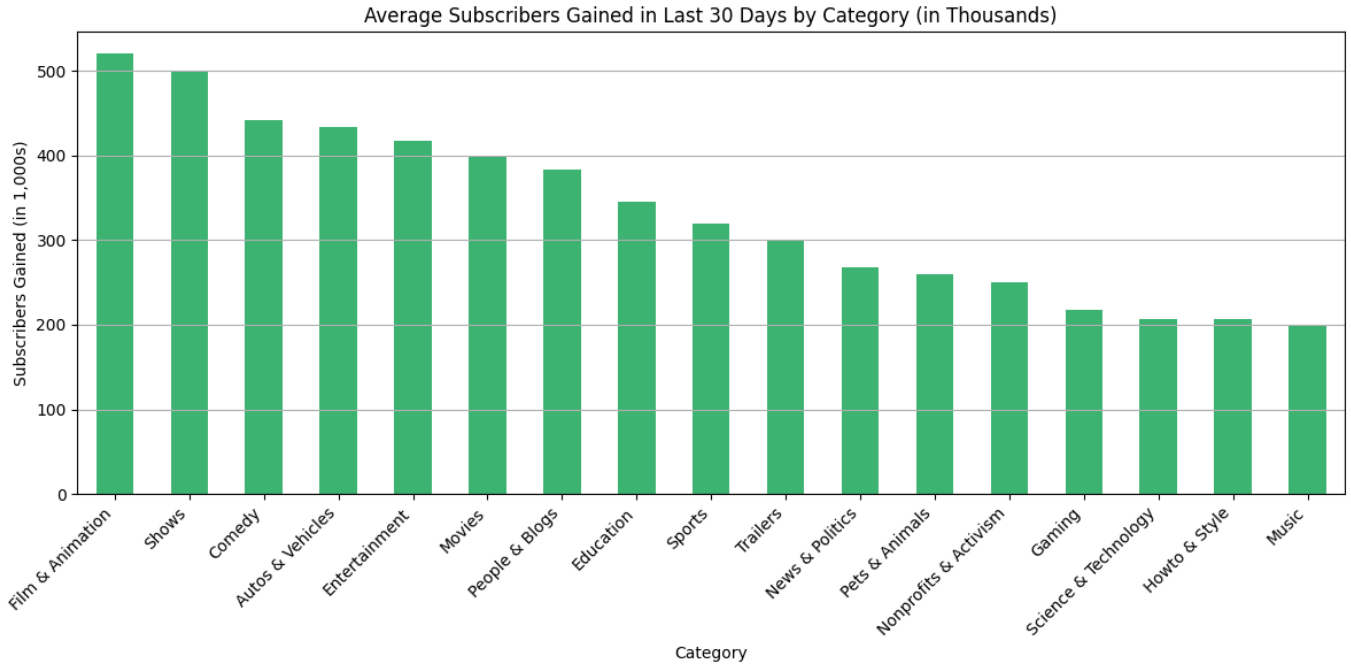
Output:



Q8.

```python
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df['subscribers_for_last_30_days'] = pd.to_numeric(df['subscribers_for_last_30_days'], errors='coerce')
df_clean = df.dropna(subset=['category', 'subscribers_for_last_30_days'])

sub_trend = df_clean.groupby('category')['subscribers_for_last_30_days'].mean()
sub_trend = (sub_trend / 1000).round(1)

sub_trend_sorted = sub_trend.sort_values(ascending=False)

plt.figure(figsize=(12, 6))
sub_trend_sorted.plot(kind='bar', color='mediumseagreen')

plt.title('Average Subscribers Gained in Last 30 Days by Category (in Thousands)')
plt.xlabel('Category')
plt.ylabel('Subscribers Gained (in 1,000s)')
plt.xticks(rotation=45, ha='right')
plt.grid(True, axis='y')
plt.tight_layout()
plt.show()
```

Output:



Q9.

```python
import pandas as pd

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df['highest_yearly_earnings'] = pd.to_numeric(df['highest_yearly_earnings'], errors='coerce')
df_clean = df.dropna(subset=['highest_yearly_earnings'])

Q1 = df_clean['highest_yearly_earnings'].quantile(0.25)
Q3 = df_clean['highest_yearly_earnings'].quantile(0.75)
IQR = Q3 - Q1
upper_bound = Q3 + 1.5 * IQR
outliers = df_clean[df_clean['highest_yearly_earnings'] > upper_bound]

print(f"Upper bound for outliers: {round(upper_bound):,} USD\n")
print("YouTube Channels with Outlier Yearly Earnings:\n")
print(outliers[['Youtuber', 'category', 'highest_yearly_earnings']].sort_values(by='highest_yearly_earnings', ascending=False))
```

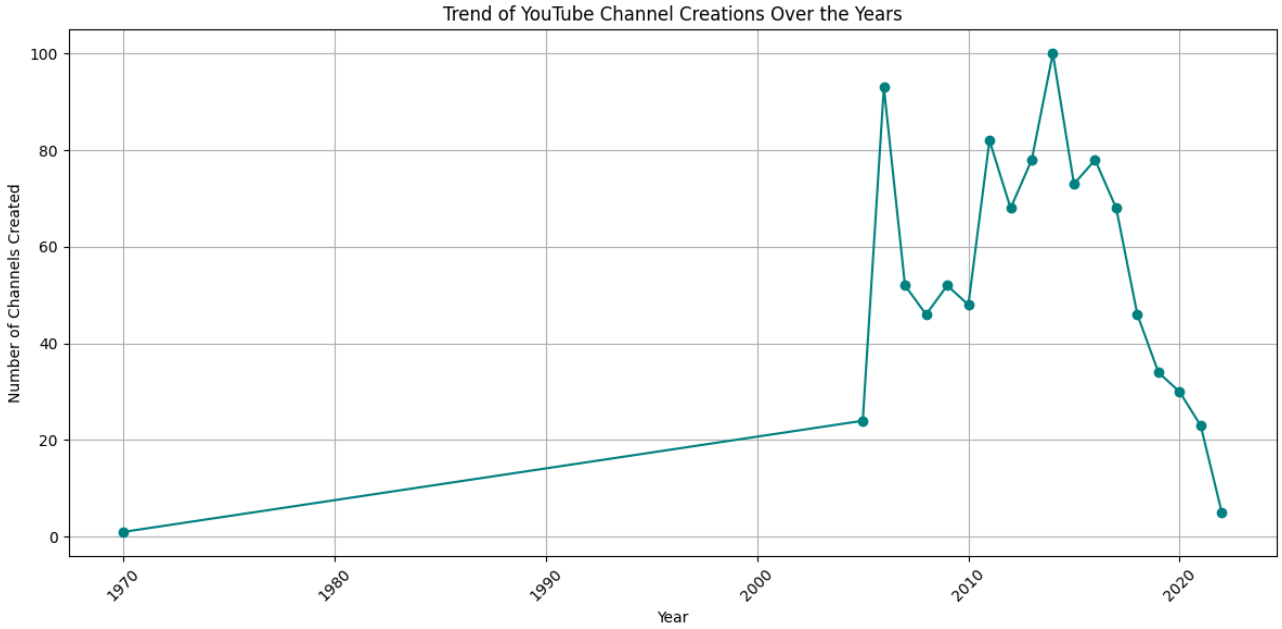
Output:

```
Upper bound for outliers: 17,468,875 USD

YouTube Channels with Outlier Yearly Earnings:

                    Youtuber          category  highest_yearly_earnings
495            ýýýýýýýýý KIMPRO               NaN                163400000.0
417                DaFuq!?Boom!  Film & Animation                110600000.0
0                     T-Series             Music                108400000.0
0                     T-Series             Music                108400000.0
0                     T-Series             Music                108400000.0
302        KL BRO Biju Rithvik     Entertainment                 97600000.0
0                     T-Series             Music                108400000.0
0                     T-Series             Music                108400000.0
302        KL BRO Biju Rithvik     Entertainment                 97600000.0
3    Cocomelon - Nursery Rhymes         Education                 94800000.0
..                          ...               ...                       ...
134                The Weeknd             Music                 17900000.0
502                  News 24   News & Politics                 17800000.0
779                SEVENGERS             Comedy                17700000.0
908               Susy Mouriz     Entertainment                17700000.0
283                     Tilak  Film & Animation                17600000.0
```

Q10.

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

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df['created_year'] = pd.to_numeric(df['created_year'], errors='coerce')
df = df.dropna(subset=['created_year'])
channel_trend = df['created_year'].value_counts().sort_index()

plt.figure(figsize=(12, 6))
plt.plot(channel_trend.index, channel_trend.values, marker='o', linestyle='-', color='teal')

plt.title('Trend of YouTube Channel Creations Over the Years')
plt.xlabel('Year')
plt.ylabel('Number of Channels Created')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

Output:



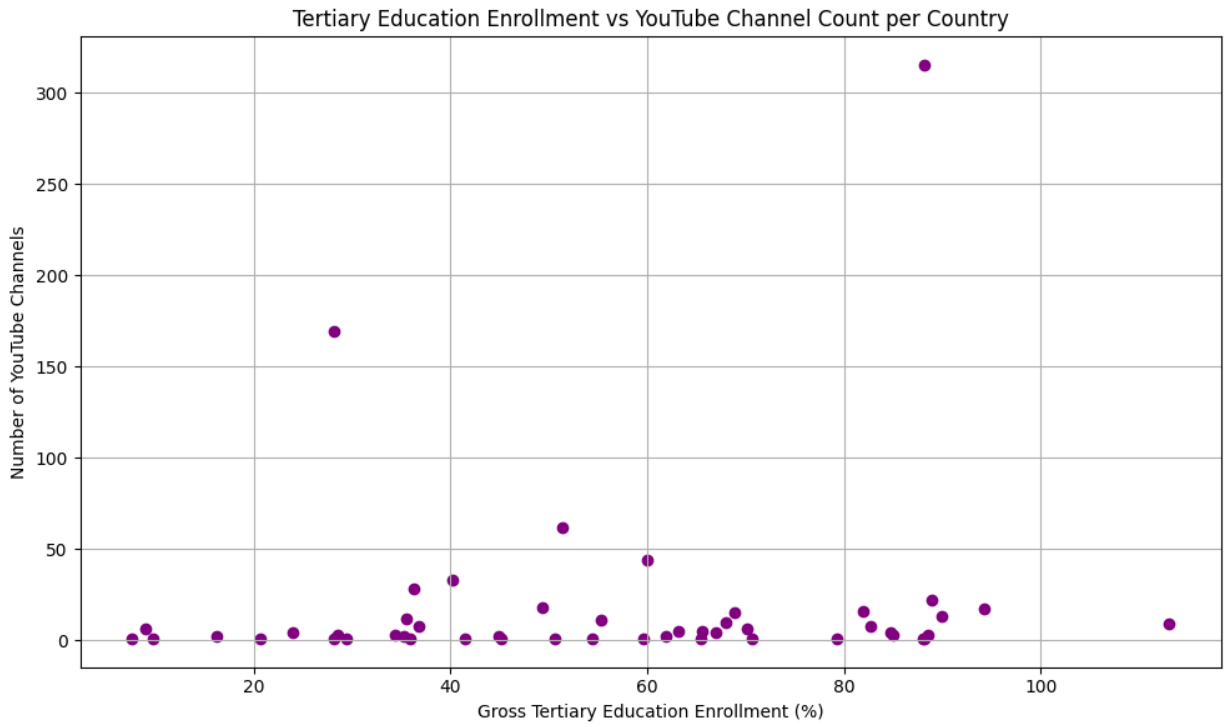
Q11.

```python
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df.dropna(subset=['Country', 'Gross tertiary education enrollment (%)'])
channel_counts = df_clean['Country'].value_counts().rename_axis('Country').reset_index(name='Channel_Count')
edu_avg = df_clean.groupby('Country')['Gross tertiary education enrollment (%)'].mean().reset_index()
merged = pd.merge(channel_counts, edu_avg, on='Country')

plt.figure(figsize=(10, 6))
plt.scatter(merged['Gross tertiary education enrollment (%)'], merged['Channel_Count'], color='purple')
plt.title('Tertiary Education Enrollment vs YouTube Channel Count per Country')
plt.xlabel('Gross Tertiary Education Enrollment (%)')
plt.ylabel('Number of YouTube Channels')
plt.grid(True)
plt.tight_layout()
plt.show()

correlation = merged['Gross tertiary education enrollment (%)'].corr(merged['Channel_Count'])
print(f"Correlation coefficient: {correlation:.2f}")
```

Output:

No meaningful relationship observed, only a few anomalous plots seen on the scatter plot.

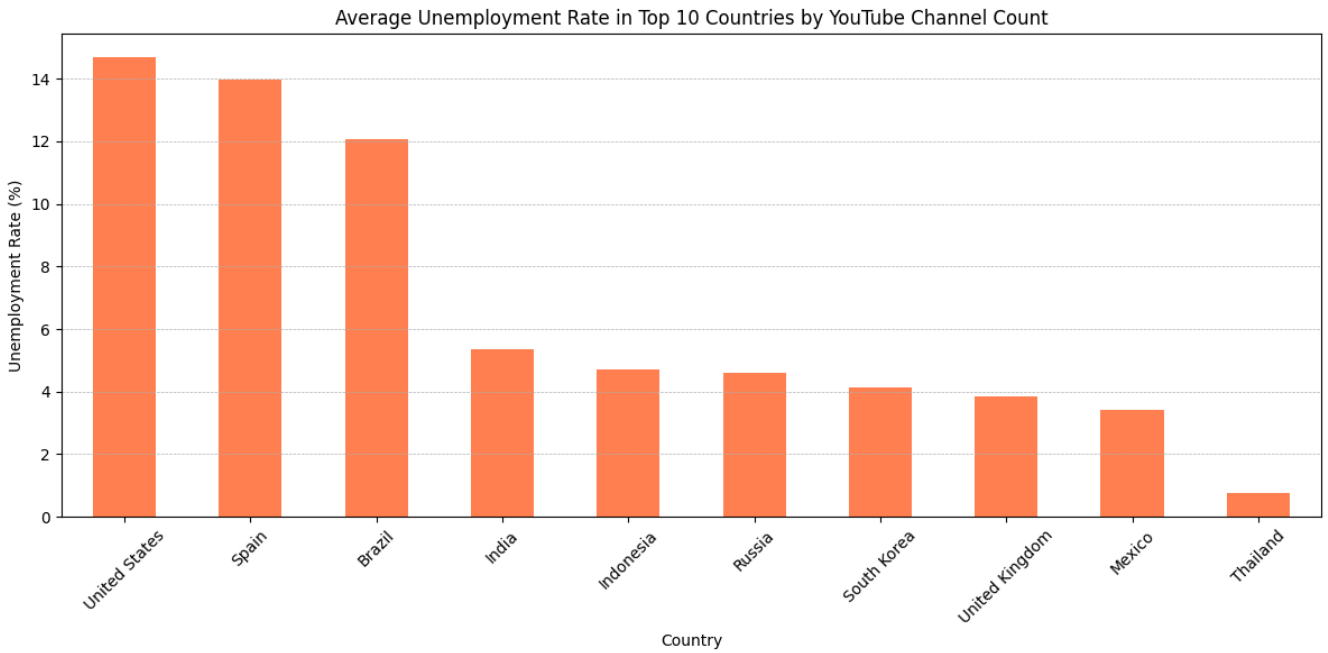
```
Correlation coefficient: 0.11
```



Q12.

```python
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df.dropna(subset=['Country', 'Unemployment rate'])
channel_counts = df_clean['Country'].value_counts().head(10)
top_countries = channel_counts.index.tolist()
top_df = df_clean[df_clean['Country'].isin(top_countries)]
avg_unemployment = top_df.groupby('Country')['Unemployment rate'].mean().sort_values(ascending=False)

plt.figure(figsize=(12, 6))
avg_unemployment.plot(kind='bar', color='coral')

plt.title('Average Unemployment Rate in Top 10 Countries by YouTube Channel Count')
plt.ylabel('Unemployment Rate (%)')
plt.xlabel('Country')
plt.grid(axis='y', linestyle='--', linewidth=0.5)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

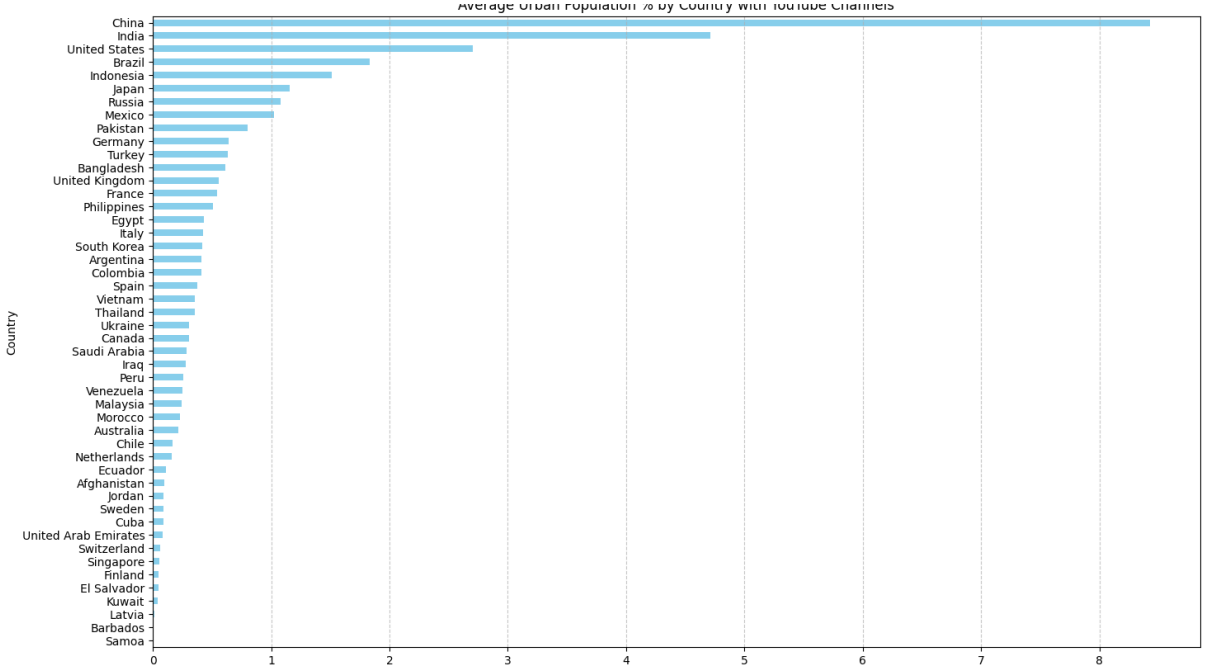
Output:



Q13.

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

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df.dropna(subset=['Country', 'Urban_population'])
df_clean['Country'] = df_clean['Country'].str.strip().str.title()
urban_pop_by_country = (
    df_clean.groupby('Country')['Urban_population']
    .mean()
    .round(2)
    .sort_values(ascending=True)
)
plt.figure(figsize=(12, 20))
urban_pop_by_country.plot(kind='barh', color='skyblue')

plt.title('Average Urban Population % by Country with YouTube Channels')
plt.xlabel('Urban Population (%)')
plt.ylabel('Country')
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

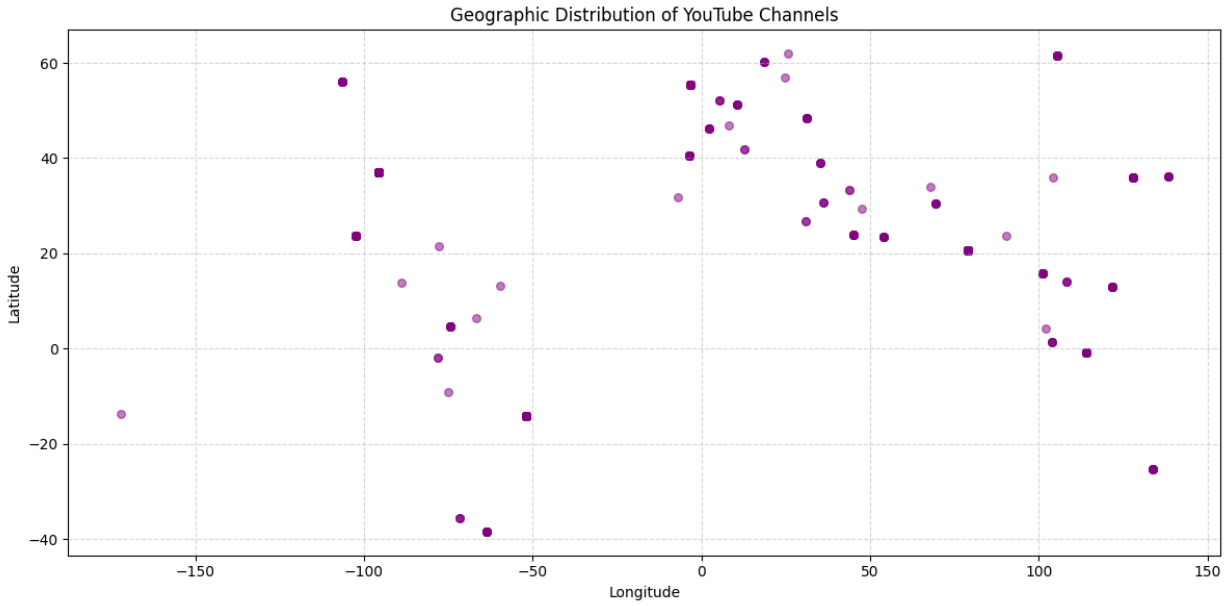
Output:



Q14.

```python
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df.dropna(subset=['Latitude', 'Longitude'])

plt.figure(figsize=(12, 6))
plt.scatter(df_clean['Longitude'], df_clean['Latitude'], alpha=0.5, s=30, color='purple')

plt.title('Geographic Distribution of YouTube Channels')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

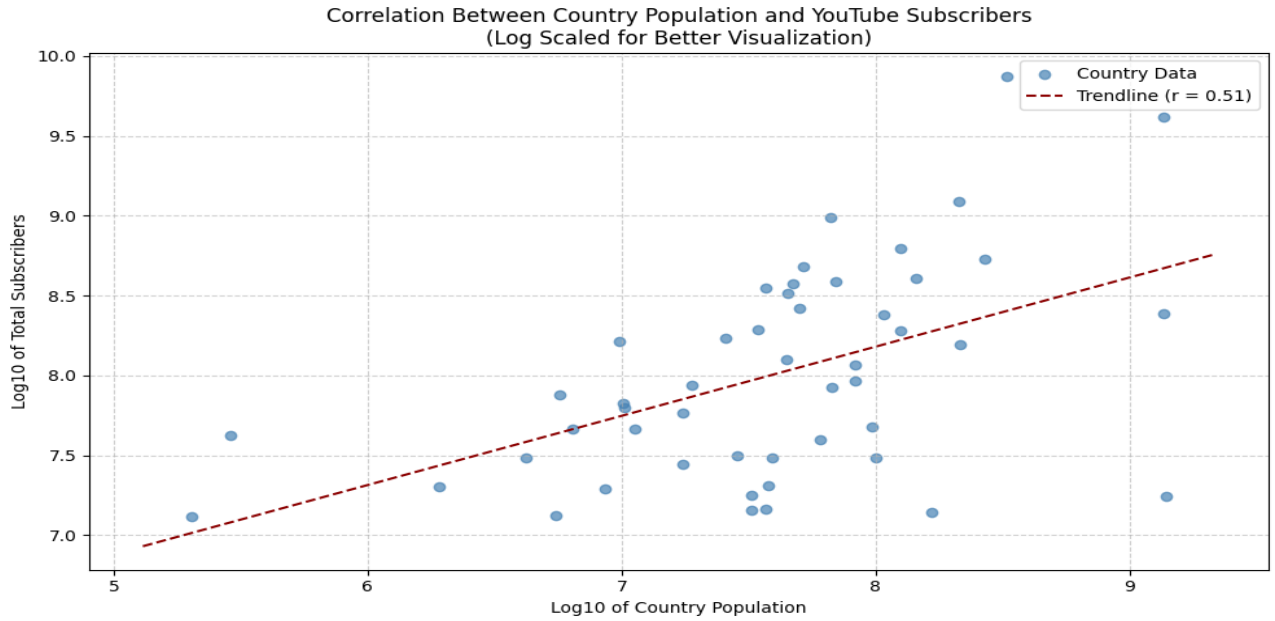
Output:



Q15.

```python
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import linregress

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df.dropna(subset=['subscribers', 'Population', 'Country'])
country_group = df_clean.groupby('Country').agg({
    'subscribers': 'sum',
    'Population': 'mean'
}).reset_index()
country_group = country_group[(country_group['subscribers'] > 0) & (country_group['Population'] > 0)]
country_group['log_subs'] = np.log10(country_group['subscribers'])
country_group['log_pop'] = np.log10(country_group['Population'])

slope, intercept, r_value, _, _ = linregress(country_group['log_pop'], country_group['log_subs'])

plt.figure(figsize=(10, 6))
plt.scatter(country_group['log_pop'], country_group['log_subs'], color='steelblue', alpha=0.7, label='Country Data')

x_vals = np.array(plt.gca().get_xlim())
y_vals = intercept + slope * x_vals
plt.plot(x_vals, y_vals, '--', color='darkred', label=f'Trendline (r = {r_value:.2f})')

plt.xlabel('Log10 of Country Population')
plt.ylabel('Log10 of Total Subscribers')
plt.title('Correlation Between Country Population and YouTube Subscribers\n(Log Scaled for Better Visualization)')
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend()
plt.tight_layout()
plt.show()
```

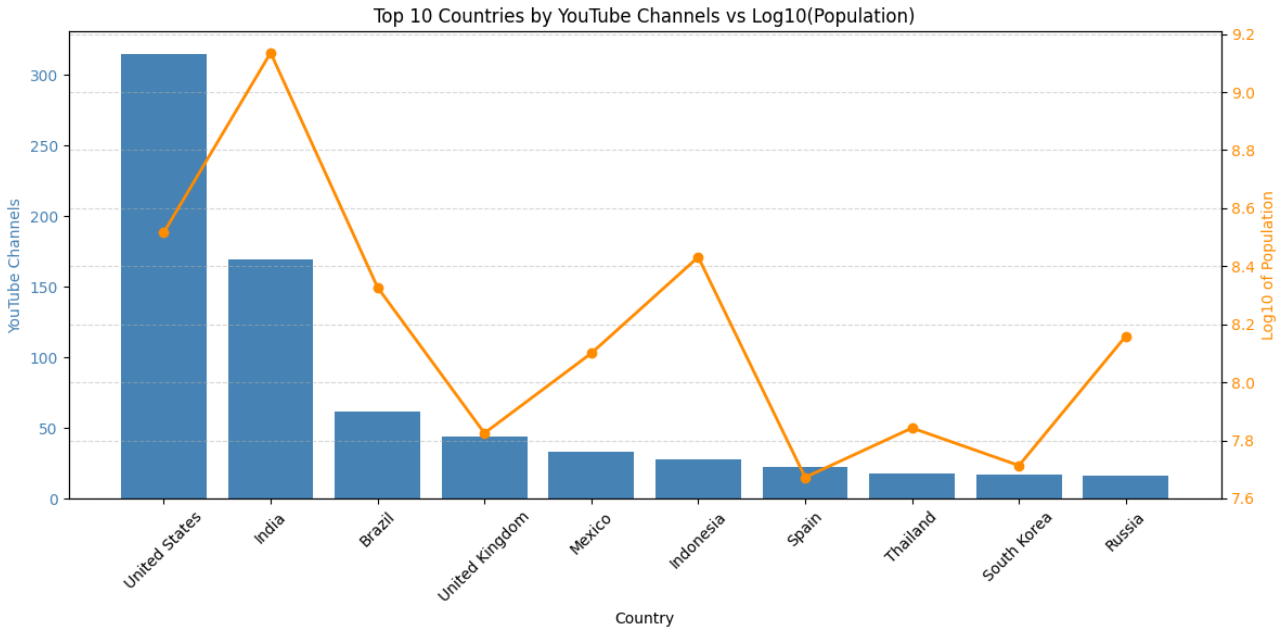
Output: There is a moderate positive correlation with a value of r=0.51 as shown in the graph.



Q16.

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

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df.dropna(subset=['Country', 'Population'])
channel_counts = df_clean['Country'].value_counts().head(10)
population_data = df_clean[df_clean['Country'].isin(channel_counts.index)][['Country', 'Population']]
population_data = population_data.drop_duplicates(subset='Country')

merged = pd.DataFrame({
    'Country': channel_counts.index,
    'YouTube Channels': channel_counts.values
})
merged = pd.merge(merged, population_data, on='Country')
merged['Log Population'] = np.log10(merged['Population'])
fig, ax1 = plt.subplots(figsize=(12, 6))

ax1.bar(merged['Country'], merged['YouTube Channels'], color='steelblue', label='YouTube Channels')
ax1.set_xlabel('Country')
ax1.set_ylabel('YouTube Channels', color='steelblue')
ax1.tick_params(axis='y', labelcolor='steelblue')
ax1.set_xticklabels(merged['Country'], rotation=45)

ax2 = ax1.twinx()
ax2.plot(merged['Country'], merged['Log Population'], color='darkorange', marker='o', linewidth=2, label='Log10(Population)')
ax2.set_ylabel('Log10 of Population', color='darkorange')
ax2.tick_params(axis='y', labelcolor='darkorange')

plt.title('Top 10 Countries by YouTube Channels vs Log10(Population)')
fig.tight_layout()
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.show()
```

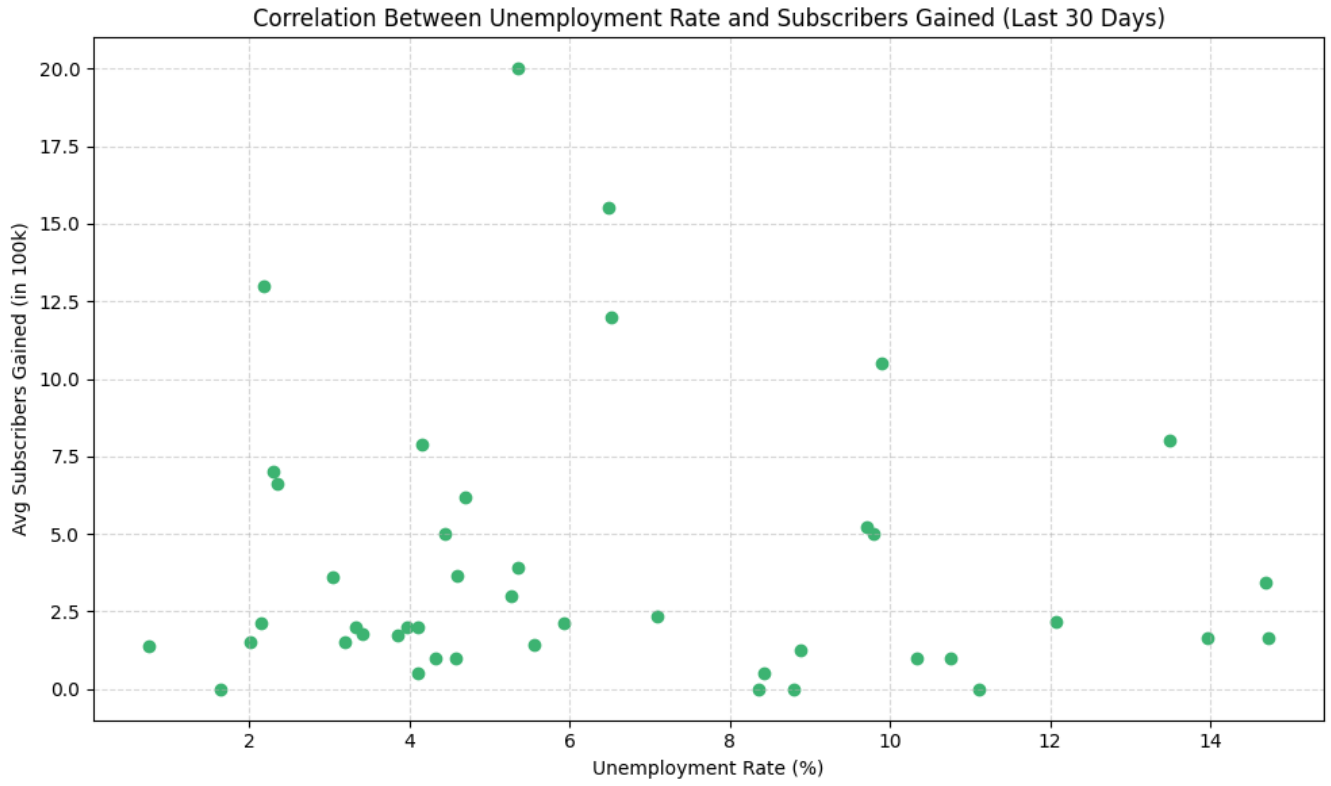
Output:



Q17.

```python
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import pearsonr

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df[['Country', 'subscribers_for_last_30_days', 'Unemployment rate']].dropna()
df_clean['subscribers_for_last_30_days'] = pd.to_numeric(df_clean['subscribers_for_last_30_days'], errors='coerce')
df_clean['Unemployment rate'] = pd.to_numeric(df_clean['Unemployment rate'], errors='coerce')
df_clean = df_clean.dropna()
grouped = df_clean.groupby('Country').agg({
    'subscribers_for_last_30_days': 'mean',
    'Unemployment rate': 'mean'
}).reset_index()

corr, _ = pearsonr(grouped['subscribers_for_last_30_days'], grouped['Unemployment rate'])
print(f"Pearson correlation: {corr:.2f}")

plt.figure(figsize=(10,6))
plt.scatter(grouped['Unemployment rate'], grouped['subscribers_for_last_30_days'] / 100000, color='mediumseagreen')
plt.title('Correlation Between Unemployment Rate and Subscribers Gained (Last 30 Days)')
plt.xlabel('Unemployment Rate (%)')
plt.ylabel('Avg Subscribers Gained (in 100k)')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

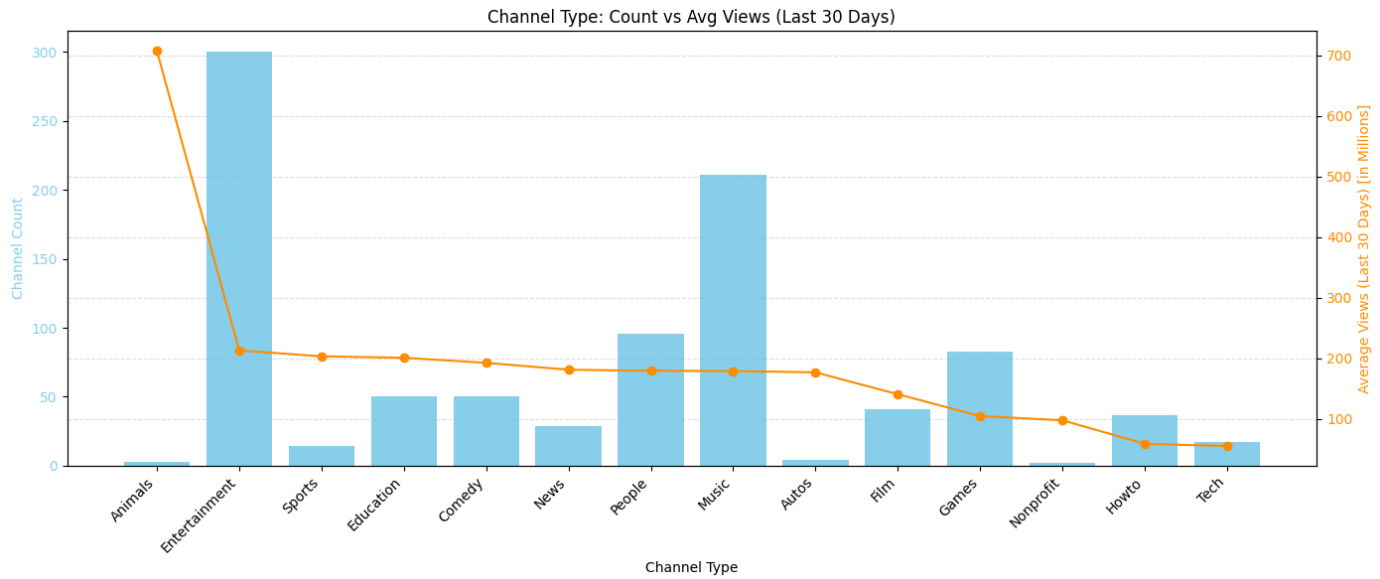
Output: Pearson correlation value is -0.06 which indicates no correlation.



Q18.

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

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df_clean = df[['channel_type', 'video_views_for_the_last_30_days']].dropna()
df_clean['video_views_for_the_last_30_days'] = pd.to_numeric(
    df_clean['video_views_for_the_last_30_days'], errors='coerce'
)
df_clean = df_clean.dropna()
df_clean = df_clean[df_clean['video_views_for_the_last_30_days'] > 0]

grouped = df_clean.groupby('channel_type').agg({
    'video_views_for_the_last_30_days': 'mean',
    'channel_type': 'count'
}).rename(columns={
    'video_views_for_the_last_30_days': 'Average Views (Last 30 Days)',
    'channel_type': 'Channel Count'
}).reset_index()
grouped = grouped.sort_values('Average Views (Last 30 Days)', ascending=False)
fig, ax1 = plt.subplots(figsize=(14, 6))

ax1.bar(grouped['channel_type'], grouped['Channel Count'], color='skyblue', label='Channel Count')
ax1.set_xlabel('Channel Type')
ax1.set_ylabel('Channel Count', color='skyblue')
ax1.tick_params(axis='y', labelcolor='skyblue')
ax1.set_xticklabels(grouped['channel_type'], rotation=45, ha='right')

ax2 = ax1.twinx()
ax2.plot(grouped['channel_type'], grouped['Average Views (Last 30 Days)'] / 1e6, color='darkorange', marker='o', label='Avg Views (in Millions)')
ax2.set_ylabel('Average Views (Last 30 Days) [in Millions]', color='darkorange')
ax2.tick_params(axis='y', labelcolor='darkorange')

plt.title('Channel Type: Count vs Avg Views (Last 30 Days)')
plt.grid(axis='y', linestyle='--', alpha=0.4)
fig.tight_layout()
plt.show()
```

Output:



Q19.

Output:

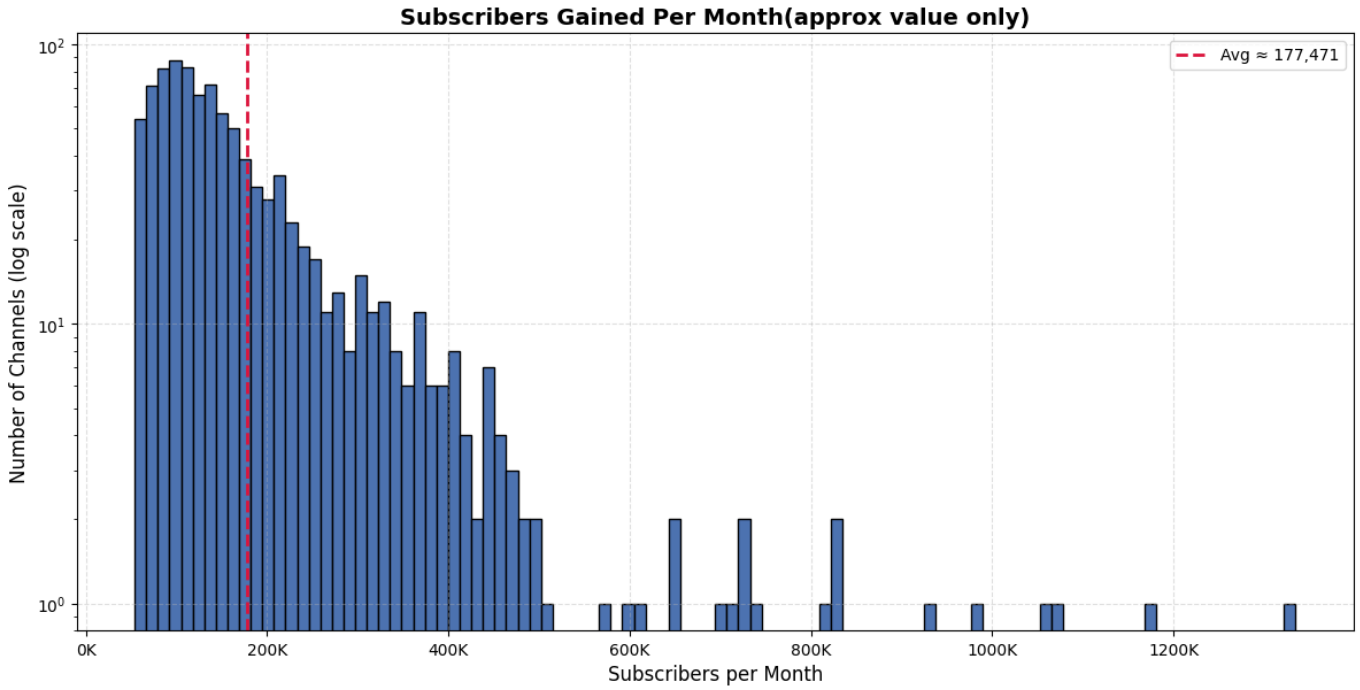
Based on the available dataset, there is no timestamped information about individual video uploads because only the total number of uploads per channel and their creation month are provided.

Therefore, we do not have sufficient data to accurately determine seasonal trends in upload behavior over time.

Q20.

```python
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter
from datetime import datetime

df = pd.read_csv("Global YouTube Statistics.csv", encoding='latin1')
df.columns = df.columns.str.strip()
df['subscribers'] = pd.to_numeric(df['subscribers'], errors='coerce')
df['created_year'] = pd.to_numeric(df['created_year'], errors='coerce')
df = df.dropna(subset=['subscribers', 'created_year'])
df = df[(df['created_year'] >= 2006) & (df['subscribers'] > 0)]

current_year = datetime.now().year
df['years_since_creation'] = current_year - df['created_year']
df = df[df['years_since_creation'] > 0]
df['months_since_creation'] = df['years_since_creation'] * 12
df['subs_per_month'] = df['subscribers'] / df['months_since_creation']
overall_avg = df['subs_per_month'].mean()

plt.figure(figsize=(12, 6))
plt.hist(df['subs_per_month'], bins=100, color='#4C72B0', edgecolor='black', log=True)
plt.gca().xaxis.set_major_formatter(FuncFormatter(lambda x, _: f'{int(x/1000)}K'))
plt.axvline(overall_avg, color='crimson', linestyle='--', linewidth=2, label=f'Avg ≈ {overall_avg:,.0f}')

plt.title('Subscribers Gained Per Month(approx value only)', fontsize=14, fontweight='bold')
plt.xlabel('Subscribers per Month', fontsize=12)
plt.ylabel('Number of Channels (log scale)', fontsize=12)
plt.legend()
plt.grid(True, linestyle='--', alpha=0.4)
plt.tight_layout()
plt.show()
```

Output:


Subscribers Gained Per Month(approx value only)
Avg ≈ 177,471
Number of Channels (log scale)
10^2
10^1
10^0
0K
200K
400K
600K
800K
1000K
1200K
Subscribers per Month