Step 1: Basic Descriptive Statistics

In [2]:

```
import pandas as pd

# Load the dataset
file_path = "C:\\Users\\91778\Downloads\\disney_plus_titles (1).csv"  # Up
disney_data = pd.read_csv(file_path)

# Display basic information and statistics
print(disney_data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1368 entries, 0 to 1367
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       1368 non-null   object
 1   type          1368 non-null   object
 2   title         1368 non-null   object
 3   director      928 non-null    object
 4   cast          1194 non-null   object
 5   country       1193 non-null   object
 6   date_added    1365 non-null   object
 7   release_year  1368 non-null   int64 
 8   rating        1366 non-null   object
 9   duration      1368 non-null   object
 10  listed_in     1368 non-null   object
 11  description   1368 non-null   object
dtypes: int64(1), object(11)
memory usage: 128.4+ KB
None
```

In [3]:

```
print(disney_data.describe(include='all'))
```

```
       show_id   type           title     director             cast  \
count     1368   1368            1368          928             1194
unique    1368      2            1368          578             1132
top         s1  Movie  A Spark Story  Jack Hannah  Winston Hibler
freq         1    991               1           17               10
mean       NaN    NaN             NaN          NaN              NaN
std        NaN    NaN             NaN          NaN              NaN
min        NaN    NaN             NaN          NaN              NaN
25%        NaN    NaN             NaN          NaN              NaN
50%        NaN    NaN             NaN          NaN              NaN
75%        NaN    NaN             NaN          NaN              NaN
max        NaN    NaN             NaN          NaN              NaN

              country         date_added  release_year rating  duration
\
count            1193               1365   1368.000000   1366      1368
unique             87                150           NaN      9       156
top     United States  November 12, 2019           NaN   TV-G  1 Season
freq              976                723           NaN    307       204
mean              NaN                NaN   2002.348684    NaN       NaN
std               NaN                NaN     22.127559    NaN       NaN
min               NaN                NaN   1928.000000    NaN       NaN
25%               NaN                NaN   1998.000000    NaN       NaN
50%               NaN                NaN   2011.000000    NaN       NaN
75%               NaN                NaN   2018.000000    NaN       NaN
max               NaN                NaN   2021.000000    NaN       NaN

                       listed_in  \
count                       1368
unique                       317
top     Animation, Comedy, Family
freq                         120
mean                         NaN
std                          NaN
min                          NaN
25%                          NaN
50%                          NaN
75%                          NaN
max                          NaN

                                                description
count                                                  1368
unique                                                 1366
top     Miguel journeys to the magical land of his anc...
freq                                                      2
mean                                                    NaN
std                                                     NaN
min                                                     NaN
25%                                                     NaN
50%                                                     NaN
75%                                                     NaN
max                                                     NaN
```

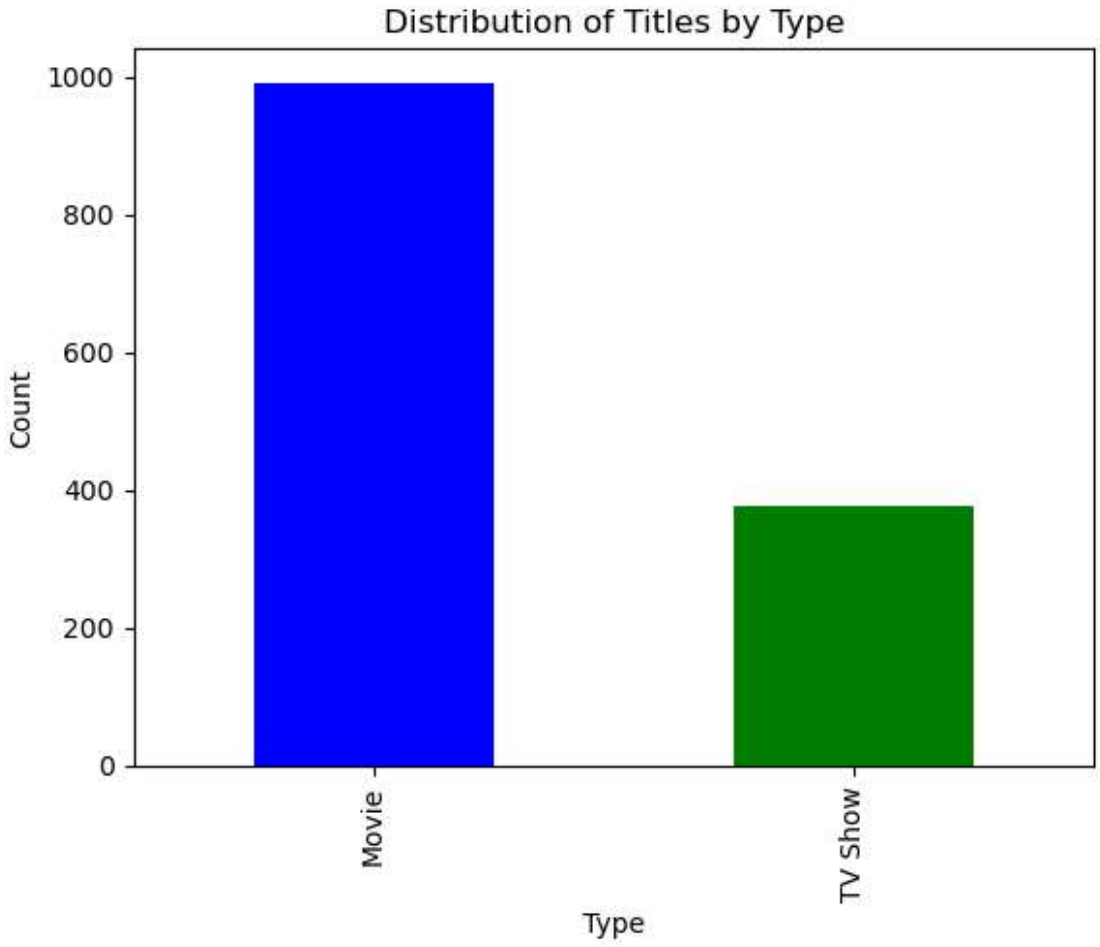
In [4]:
```
print(disney_data.head())
```

```
  show_id     type                              title  \
0      s1    Movie                   A Spark Story
1      s2    Movie                  Spooky Buddies
2      s3    Movie          The Fault in Our Stars
3      s4  TV Show                 Dog: Impossible
4      s5  TV Show  Spidey And His Amazing Friends

                     director  \
0  Jason Sterman, Leanne Dare
1                Robert Vince
2                  Josh Boone
3                         NaN
4                         NaN

                                                cast                coun
try  \
0                      Apthon Corbin, Louis Gonzales                    
NaN
1  Tucker Albrizzi, Diedrich Bader, Ameko Eks Mas...  United States, Can
ada
2  Shailene Woodley, Ansel Elgort, Laura Dern, Sa...          United Sta
tes
3                                   Matt Beisner          United Sta
tes
4  Benjamin Valic, Lily Sanfelippo, Jakari Fraser...          United Sta
tes

           date_added  release_year rating   duration  \
0  September 24, 2021          2021  TV-PG     88 min
1  September 24, 2021          2011      G     93 min
2  September 24, 2021          2014  PG-13    127 min
3  September 22, 2021          2019  TV-PG  2 Seasons
4  September 22, 2021          2021   TV-Y   1 Season

                             listed_in  \
0                          Documentary
1                Comedy, Fantasy, Kids
2        Coming of Age, Drama, Romance
3  Animals & Nature, Docuseries, Family
4    Action-Adventure, Animation, Kids

                                         description
0  Two Pixar filmmakers strive to bring their uni...
1  The puppies go on a spooky adventure through a...
2  Hazel and Gus share a love that sweeps them on...
3  Matt Beisner uses unique approaches to modifyi...
4  Spidey teams up with pals to become The Spidey...
```

Step 2: Data Visualization

Distribution of Titles by Type:

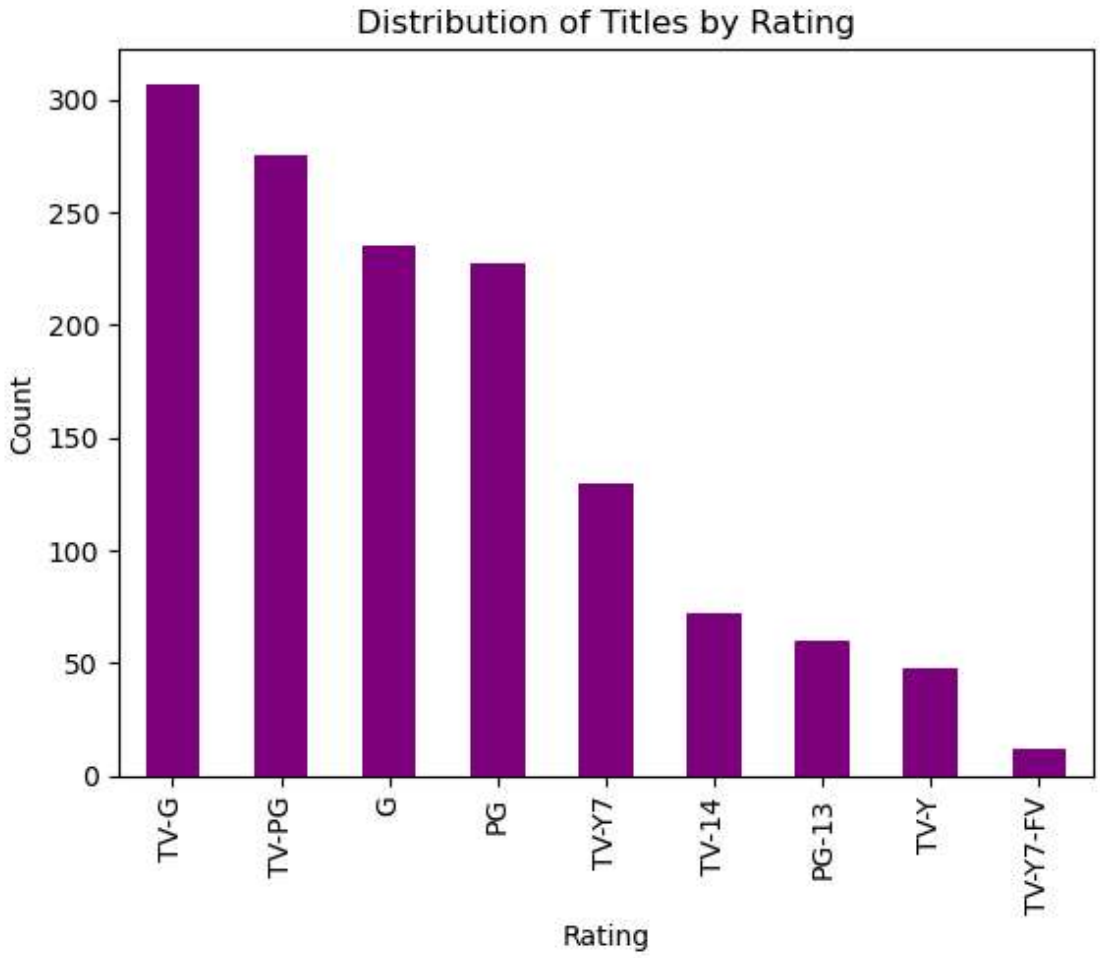
In [5]:
```
import matplotlib.pyplot as plt

# Plot the distribution of titles by type (Movie/TV Show)
disney_data['type'].value_counts().plot(kind='bar', color=['blue', 'green
plt.title('Distribution of Titles by Type')
plt.xlabel('Type')
plt.ylabel('Count')
plt.show()
```



Distribution of Titles by Rating

In [6]:

```
# Plot the distribution of titles by rating
disney_data['rating'].value_counts().plot(kind='bar', color='purple')
plt.title('Distribution of Titles by Rating')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.show()
```

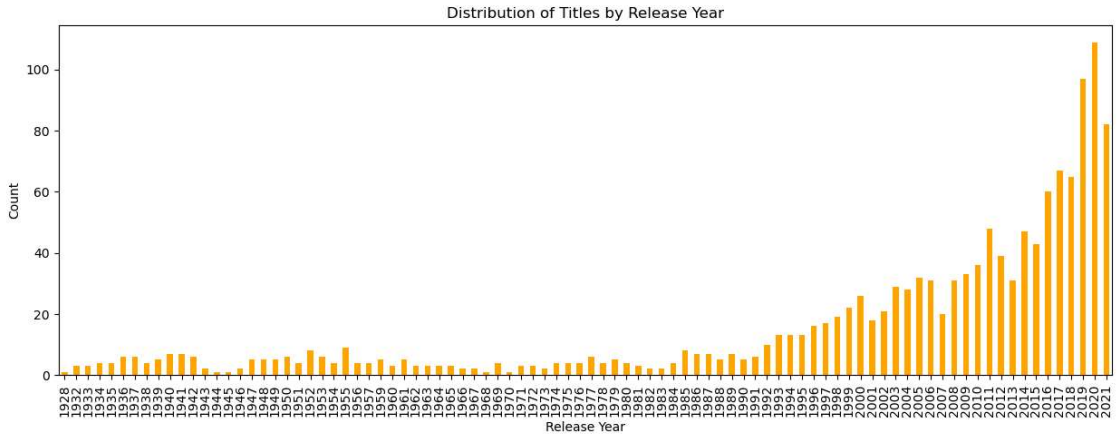


In [ ]:

```
Distribution of Titles by Release Year
```

In [7]:

```
# Plot the distribution of titles by release year
disney_data['release_year'].value_counts().sort_index().plot(kind='bar', f
plt.title('Distribution of Titles by Release Year')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.show()
```

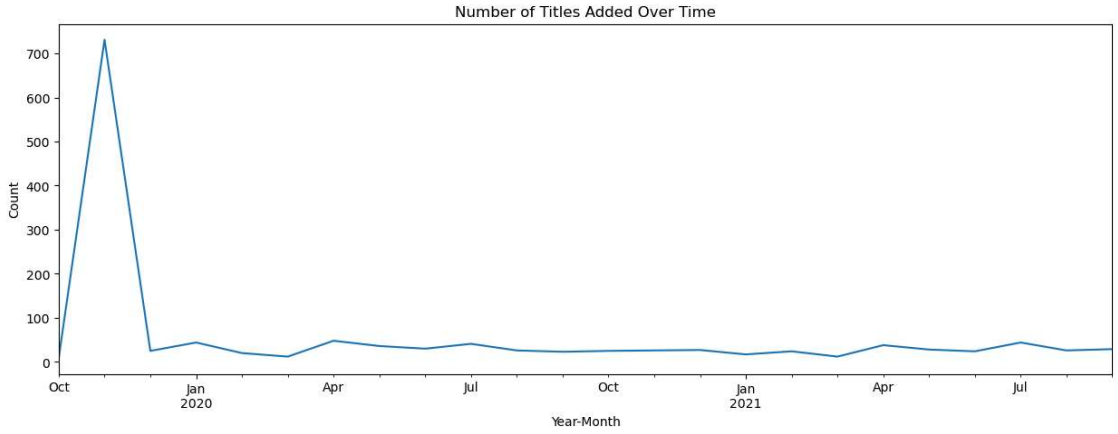


Step 3: Detailed Analysis of Key Features

```
Titles Added Over Time
```

In [8]:

```
# Ensure 'date_added' is in datetime format
disney_data['date_added'] = pd.to_datetime(disney_data['date_added'], erro

# Extract year and month for aggregation
disney_data['year_month_added'] = disney_data['date_added'].dt.to_period(

# Aggregate by month/year
titles_per_month = disney_data.groupby('year_month_added').size()

# Plot the number of titles added over time
titles_per_month.plot(kind='line', figsize=(15, 5))
plt.title('Number of Titles Added Over Time')
plt.xlabel('Year-Month')
plt.ylabel('Count')
plt.show()
```



Step 4: Time Series Analysis with ARIMA

In [11]:

```
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
import numpy as np
# Ensure 'date_added' is in datetime format
disney_data['date_added'] = pd.to_datetime(disney_data['date_added'], erro

# Extract year and month for aggregation
disney_data['year_month_added'] = disney_data['date_added'].dt.to_period(

# Aggregate by month/year
titles_per_month = disney_data.groupby('year_month_added').size()

# Convert to DataFrame for better handling
titles_per_month_df = titles_per_month.to_frame(name='titles_added').reset

# Ensure the index is in datetime format
titles_per_month_df['year_month_added'] = titles_per_month_df['year_month_
titles_per_month_df['year_month_added'] = pd.to_datetime(titles_per_month_
# Split the data into training and test sets
train_size = int(len(titles_per_month_df) * 0.8)
train, test = titles_per_month_df['titles_added'][:train_size], titles_per
```

In [12]:

```
arima_model(train, order):

arima_model = ARIMA(train, order=order)
arima_fit = arima_model.fit()
return arima_fit
pt np.linalg.LinAlgError:
print("LinAlgError encountered, adjusting model parameters.")
return None
pt Exception as e:
print(f"An error occurred: {e}")
return None

IMA model on the training data
(5, 1, 0)
t = fit_arima_model(train, order)

if the model was fitted successfully
_fit:
recast the test set
cast = arima_fit.forecast(steps=len(test))

lculate the mean squared error
= mean_squared_error(test, forecast)
 = np.sqrt(mse)

ot the actual vs forecasted values
figure(figsize=(12, 6))
plot(train.index, train, label='Training Data')
plot(test.index, test, label='Actual Data')
plot(test.index, forecast, label='Forecasted Data', color='red')
title('ARIMA Model Forecast')
xlabel('Date')
ylabel('Number of Titles Added')
legend()
grid(True)
show()

t(f"Root Mean Squared Error: {rmse}")

recast future values (e.g., next 12 months)
re_forecast = arima_fit.forecast(steps=12)

ot the forecasted future values
figure(figsize=(12, 6))
plot(titles_per_month_df['year_month_added'], titles_per_month_df['titles_
plot(pd.date_range(start=titles_per_month_df['year_month_added'].iloc[-1],
title('Future Forecast with ARIMA Model')
xlabel('Date')
ylabel('Number of Titles Added')
legend()
grid(True)
show()

t("ARIMA model fitting was unsuccessful.")
```

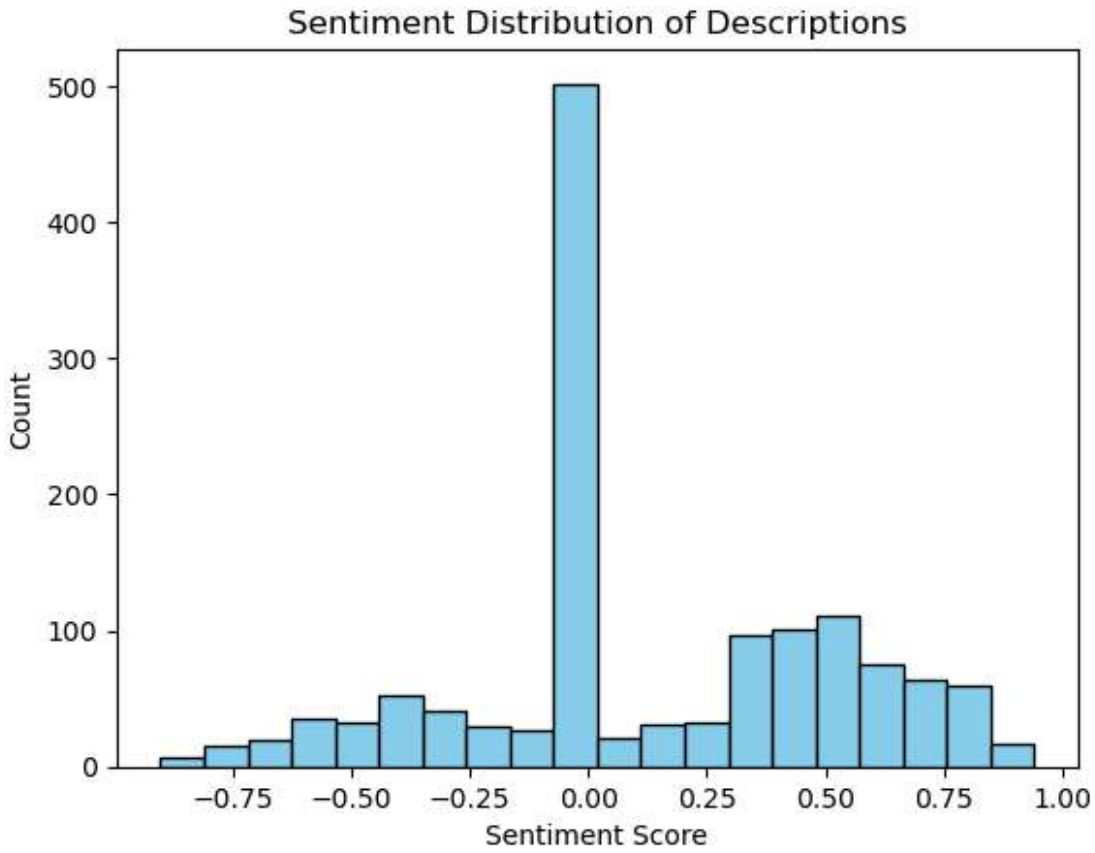
```
LinAlgError encountered, adjusting model parameters.
ARIMA model fitting was unsuccessful.
```

Step 5: Sentiment Analysis on Descriptions

In [13]:

```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import nltk

# Download VADER lexicon
nltk.download('vader_lexicon')

# Initialize VADER sentiment analyzer
sid = SentimentIntensityAnalyzer()

# Apply sentiment analysis on the descriptions
disney_data['description'] = disney_data['description'].astype(str)  # Ens
disney_data['sentiment'] = disney_data['description'].apply(lambda x: sid.

# Display sentiment analysis results
print(disney_data[['title', 'description', 'sentiment']].head())

# Plot sentiment distribution
disney_data['sentiment'].plot(kind='hist', bins=20, color='skyblue', edgec
plt.title('Sentiment Distribution of Descriptions')
plt.xlabel('Sentiment Score')
plt.ylabel('Count')
plt.show()
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\91778\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!

                          title  \
0                   A Spark Story
1                  Spooky Buddies
2          The Fault in Our Stars
3                 Dog: Impossible
4  Spidey And His Amazing Friends

                                         description  sentiment
0  Two Pixar filmmakers strive to bring their uni...     0.2263
1  The puppies go on a spooky adventure through a...    -0.2023
2  Hazel and Gus share a love that sweeps them on...     0.7506
3  Matt Beisner uses unique approaches to modifyi...     0.0000
4  Spidey teams up with pals to become The Spidey...     0.0000
```

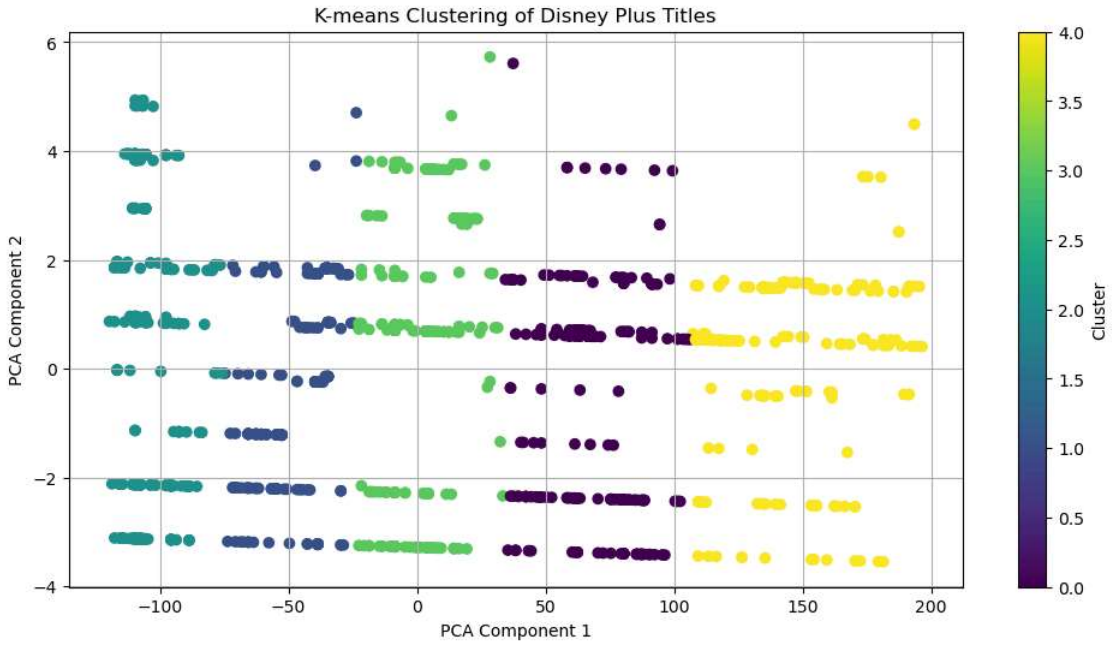


Step 5: Clustering Analysis for Segmentation

In [14]:

```
from sklearn.preprocessing import LabelEncoder
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA

# Prepare data for clustering
# Convert categorical variables to numerical
label_encoders = {}
categorical_columns = ['type', 'rating', 'listed_in']

for column in categorical_columns:
    le = LabelEncoder()
    disney_data[column] = le.fit_transform(disney_data[column].astype(str)
    label_encoders[column] = le

# Select features for clustering
features = ['type', 'rating', 'listed_in']
X = disney_data[features].dropna()

# Apply K-means clustering
kmeans = KMeans(n_clusters=5, random_state=42)
disney_data['cluster'] = kmeans.fit_predict(X)

# Reduce dimensionality for visualization
pca = PCA(n_components=2)
components = pca.fit_transform(X)

# Plot the clusters
plt.figure(figsize=(12, 6))
plt.scatter(components[:, 0], components[:, 1], c=disney_data['cluster'],
plt.title('K-means Clustering of Disney Plus Titles')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.colorbar(label='Cluster')
plt.grid(True)
plt.show()
```

```
C:\Users\91778\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to
'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warn
ing
  warnings.warn(
C:\Users\91778\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:13
82: UserWarning: KMeans is known to have a memory leak on Windows with M
KL, when there are less chunks than available threads. You can avoid it
by setting the environment variable OMP_NUM_THREADS=6.
  warnings.warn(
```


K-means Clustering of Disney Plus Titles
PCA Component 2
PCA Component 1
Cluster
6
4
2
0
−2
−4
−100
−50
0
50
100
150
200
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0

In [ ]:

In [ ]: