# School Of Computer Application

***Bachelors of Computer Applications***

## Babu Banarasi Das University

## Lucknow

**Academic Session** 2024 - 2025

**Name :** Harshit Tripathi

**Section :** BCADS23

**Roll No:** University :- 1230258192
Class :- 38

**Assignment:** Data Science
(Netflix Analytics)

**Semester:** 4th

**Date:** 10/5/2025

**Submit To :** Mr.Ankit Soni Sir

1 Floor, H-Block, BBDU,BBD City, Faizabad Road, Lucknow (U. P.) INDIA 226028

**PHONE: HEAD:** 0522-3911127,3911321**Dept. Adm. & Exam Cell:** 0522-3911326**Dept. T&P Cell:** 0522-3911305; **E-Mail:** head.sca@gmail.com

W W W .

In [1]:
```
!pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\users\tripa\anaconda3\lib\site-
packages (0.12.2)
Requirement already satisfied: numpy!=1.24.0,>=1.17 in c:\users\tripa\anacon
da3\lib\site-packages (from seaborn) (1.24.3)
Requirement already satisfied: pandas>=0.25 in c:\users\tripa\anaconda3\lib
\site-packages (from seaborn) (2.0.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.1 in c:\users\tripa\ana
conda3\lib\site-packages (from seaborn) (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\tripa\anaconda3
\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\tripa\anaconda3\lib
\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\tripa\anaconda3
\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\tripa\anaconda3
\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\tripa\anaconda3\l
ib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\tripa\anaconda3\lib
\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (10.0.1)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\tripa\anaco
nda3\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\tripa\anacon
da3\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\tripa\anaconda3\lib
\site-packages (from pandas>=0.25->seaborn) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\tripa\anaconda3\li
b\site-packages (from pandas>=0.25->seaborn) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\tripa\anaconda3\lib\site
-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.1->seaborn) (1.1
6.0)
```

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
df = pd.read_csv(r"C:\Users\tripa\OneDrive\Desktop\Netflix_dataset.csv")
df.head()
```

Out[3]:

| | show_id | type | title | director | cast | country | date_added | release_year | rating | d |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | s1 | Movie | Dick Johnson Is Dead | Kirsten Johnson | NaN | United States | September 25, 2021 | 2020 | PG-13 | |
| **1** | s2 | TV Show | Blood & Water | NaN | Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban... | South Africa | September 24, 2021 | 2021 | TV-MA | S |
| **2** | s3 | TV Show | Ganglands | Julien Leclercq | Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi... | NaN | September 24, 2021 | 2021 | TV-MA | |
| **3** | s4 | TV Show | Jailbirds New Orleans | NaN | NaN | NaN | September 24, 2021 | 2021 | TV-MA | |
| **4** | s5 | TV Show | Kota Factory | NaN | Mayur More, Jitendra Kumar, Ranjan Raj, Alam K... | India | September 24, 2021 | 2021 | TV-MA | S |

In [4]:
```
# to get a summary of the dataset including null counts and data types .
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

In [5]:
```
#Statistical summary.
df.describe()
```

Out[5]:

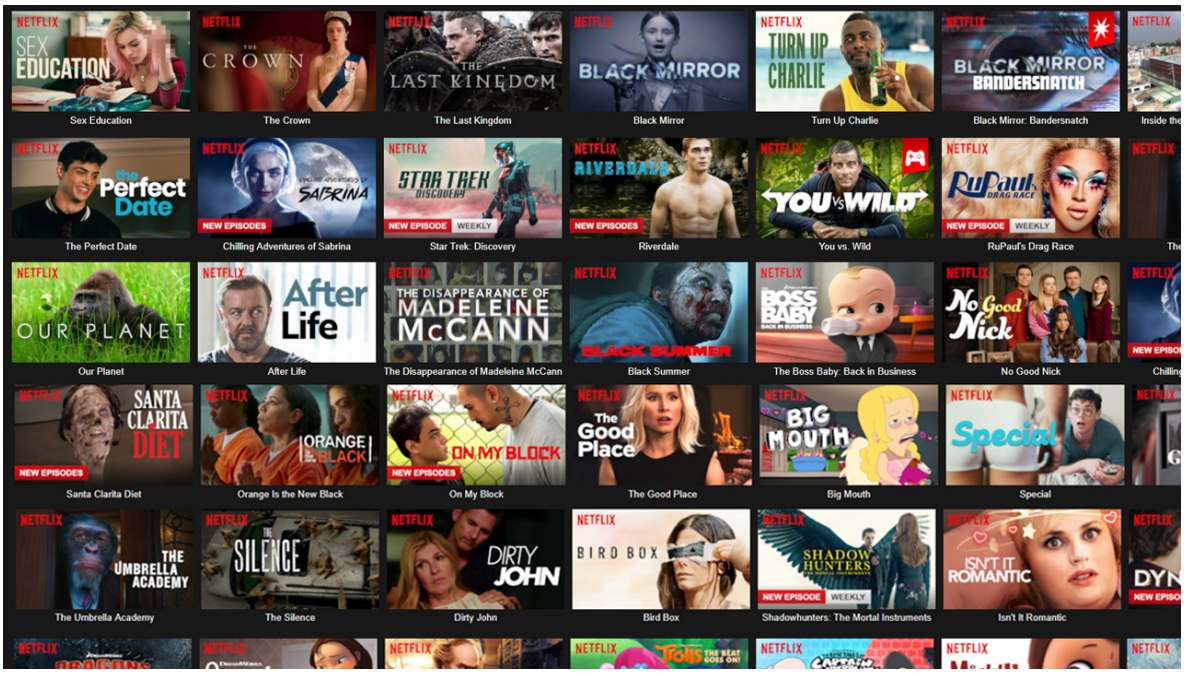
| | release_year |
|---|---|
| **count** | 8807.000000 |
| **mean** | 2014.180198 |
| **std** | 8.819312 |
| **min** | 1925.000000 |
| **25%** | 2013.000000 |
| **50%** | 2017.000000 |
| **75%** | 2019.000000 |
| **max** | 2021.000000 |

In [6]:
```
#generate column names from the datatset .
df.columns
```

Out[6]:
```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_adde
d',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

In [7]:
```
#provide unique values per column .
df.nunique()
```

Out[7]:
```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```




In [8]:
```
#Checking for missing data values in dataset .
df.isnull().sum()
```

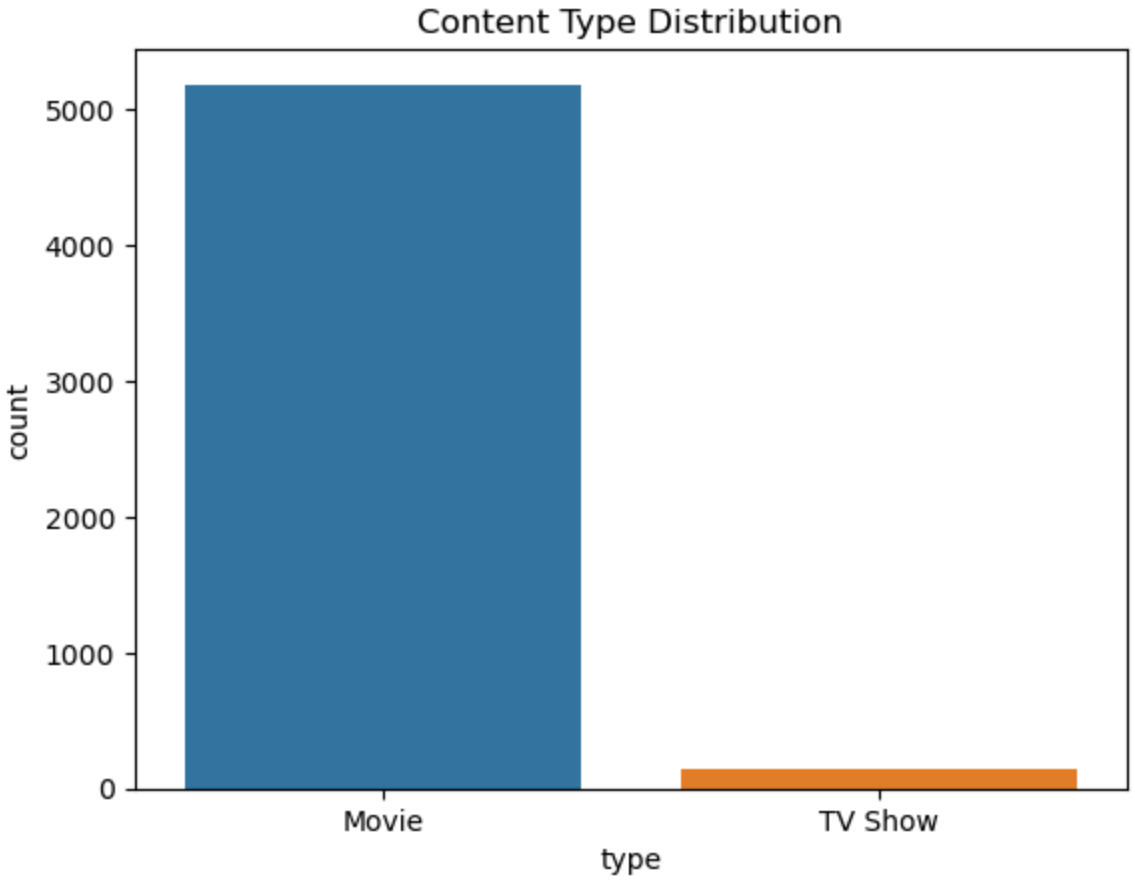
Out[8]:
```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
dtype: int64
```

In [9]:
```
#Handling the missing values .
df.dropna(subset=['director'],inplace=True)
df.dropna(subset=['cast'],inplace=True)
df.dropna(subset=['country'],inplace=True)
df.dropna(subset=['date_added'],inplace=True)
df.dropna(subset=['rating'],inplace=True)
df.dropna(subset=['duration'],inplace=True)
```

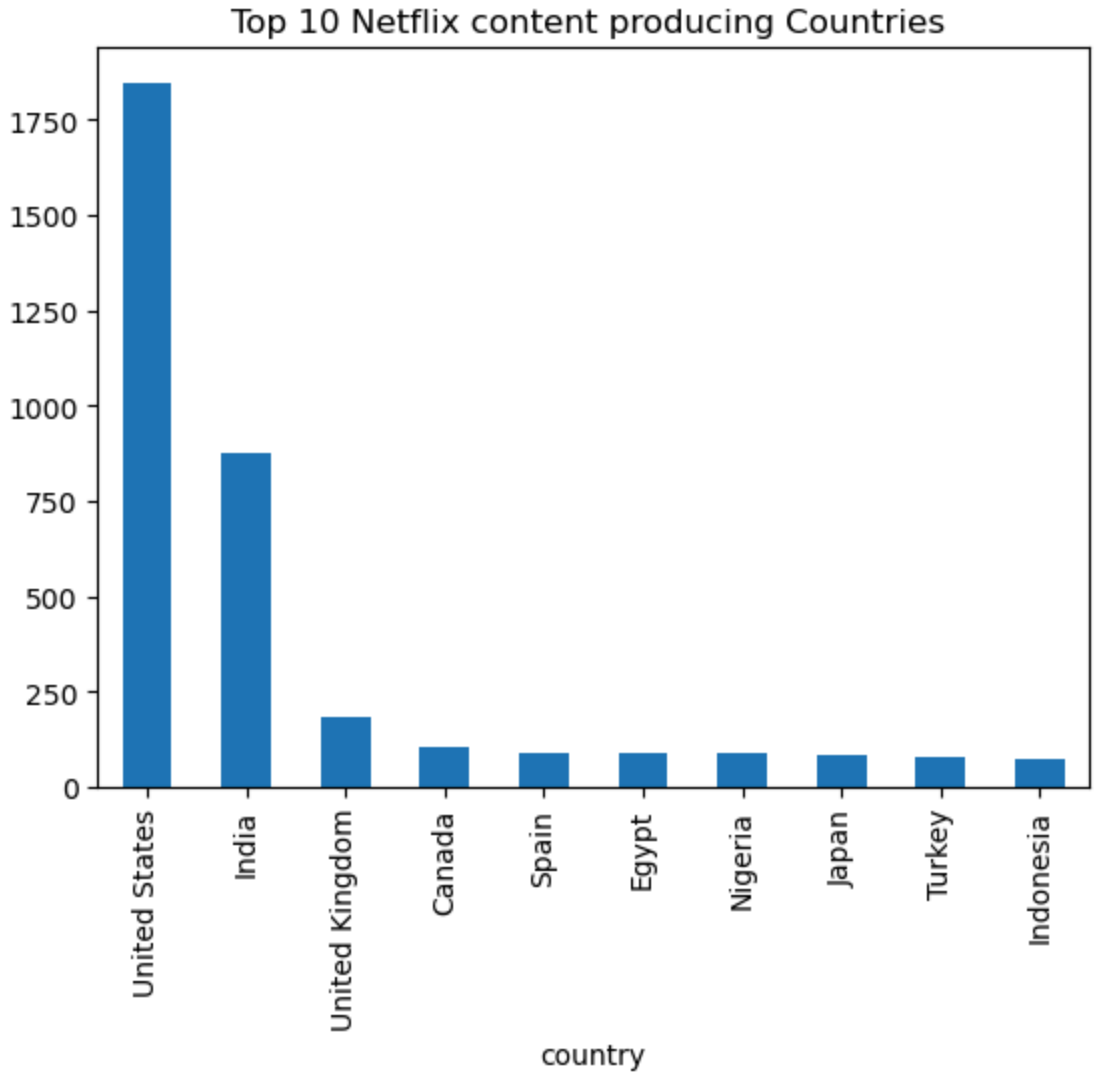
In [10]:
```
df.isnull().sum()
```

Out[10]:
```
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
description     0
dtype: int64
```

In [11]:
```
#distribution of content type throughout the dataset(e.g movies vs tv shows).
sns.countplot(data=df, x='type')
plt.title("Content Type Distribution")
plt.show()
```



In [12]:
```
#top genres producing or top content producing netflix content .
df['country'].value_counts().head(10).plot(kind='bar')
plt.title("Top 10 Netflix content producing Countries ")
plt.show()
```

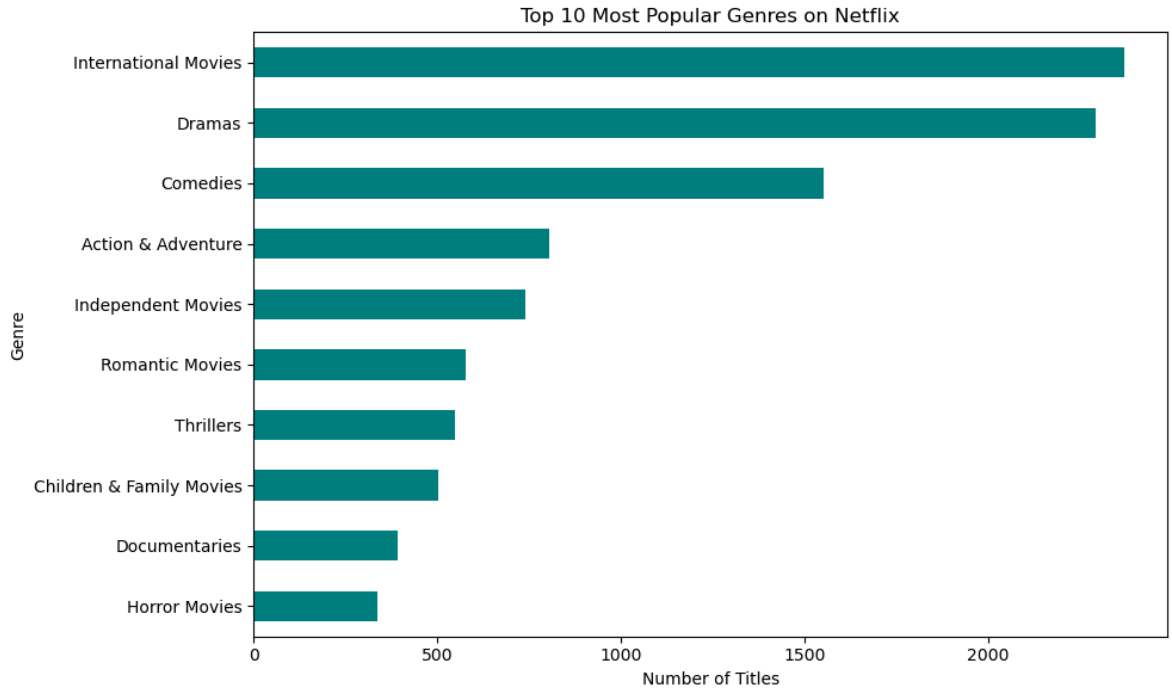



In [13]:
```
from collections import Counter
```

In [14]:
```
#Combining all genres into one list .
genres = df['listed_in'].dropna()
all_genres = ', '.join(genres).split(', ')
genre_counts = Counter(all_genres)
```

In [15]:
```
#with DataFrame get top 10 famous genres .
top_genres = pd.DataFrame(genre_counts.items(),columns = ['Genre', 'Count']).s
```

In [16]:
```
# Plot
top_genres.plot(kind='barh', x='Genre', y='Count', figsize=(10,6), color='teal
plt.title('Top 10 Most Popular Genres on Netflix')
plt.xlabel('Number of Titles')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```





# Yearly trend of adding new content of Netflix or New releases each year.

In [17]:
```
# Convert 'date_added' to datetime.
df['date_added'] = pd.to_datetime(df['date_added'], errors = 'coerce')
```

In [18]:
```
# Extract year from the 'date_added' column
df['year_added'] = df['date_added'].dt.year
```

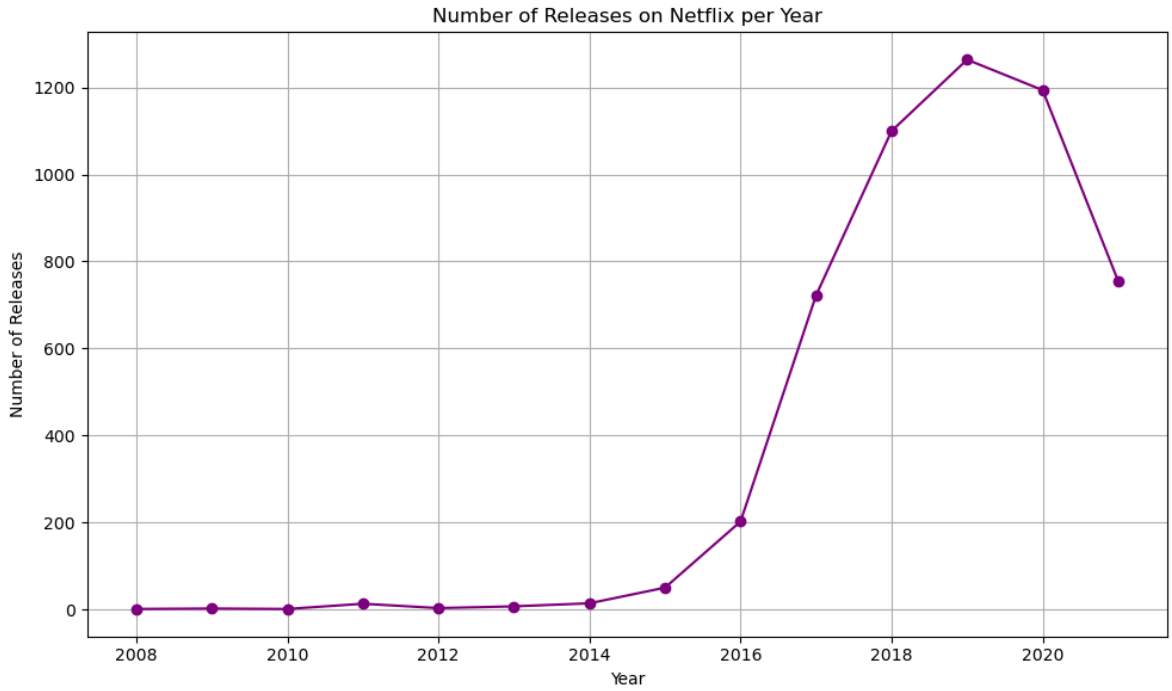
In [19]:
```
# Count the number of releases per year
release_counts = df['year_added'].value_counts().sort_index()
```

In [20]:
```
# finding the year with the highest number of releases .
highest_year = release_counts.idxmax()
highest_count = release_counts.max()
```

In [21]:
```
print(f"The year with the highest number of releases is {highest_year} with {h
```

```
The year with the highest number of releases is 2019.0 with 1264 titles.
```

In [22]:
```
# Plot the release counts per year
release_counts.plot(kind='line', figsize=(10,6), color='purple', marker='o')
plt.title('Number of Releases on Netflix per Year')
plt.xlabel('Year')
plt.ylabel('Number of Releases')
plt.grid(True)
plt.tight_layout()
plt.show()
```




# Most frequent ratings on Netflix.

In [23]:
```
#first we drop missing rating values.
df = df.dropna(subset=['rating'])
```

In [24]:
```
#Count each rating
rating_counts = df['rating'].value_counts()
```

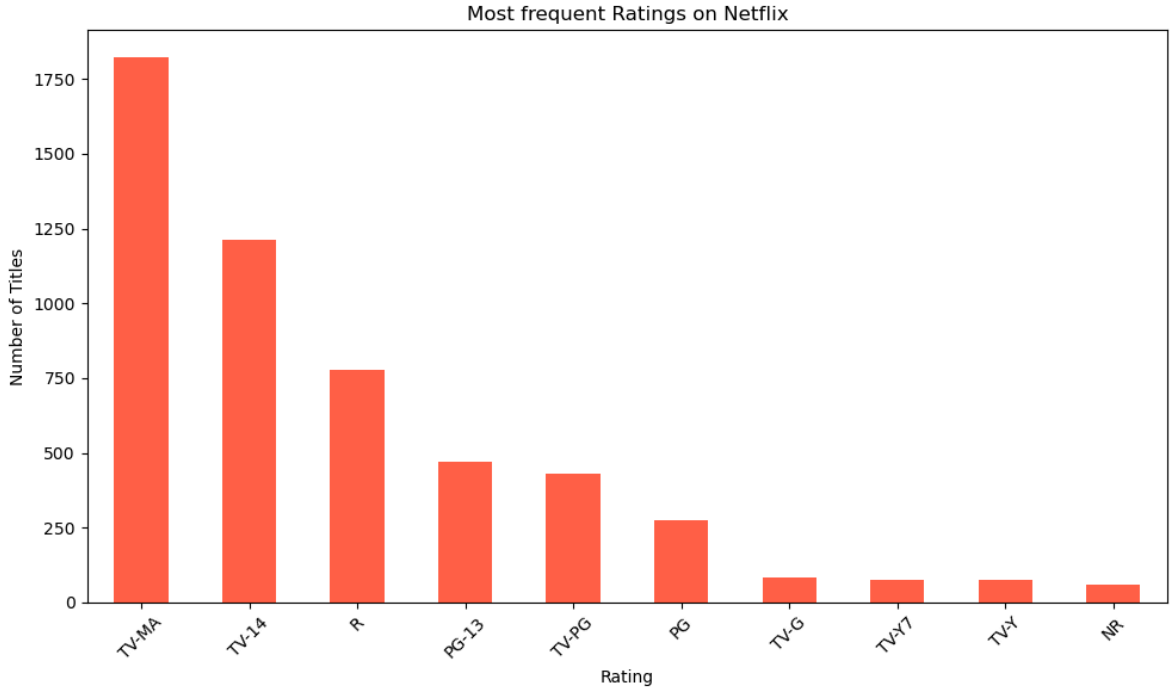
In [25]:
```
#Display the top 10 ratings .
print("Top Ratings on Netflix:")
print(rating_counts.head(0))
```

```
Top Ratings on Netflix:
Series([], Name: count, dtype: int64)
```

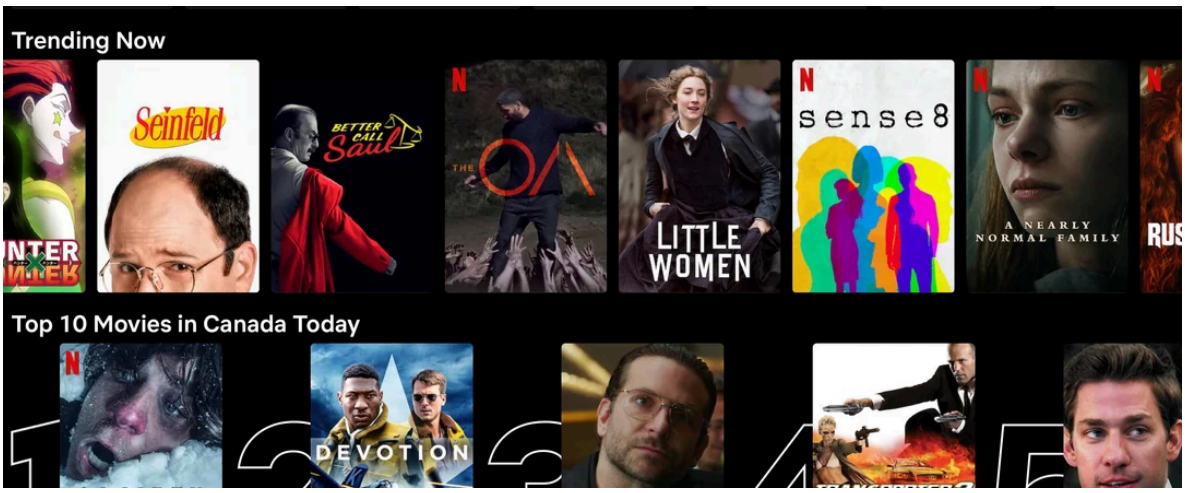
In [26]:
```
#Printing the most common ratings .
print("Top ratings on Netflix:")
print(rating_counts.head(10))
```

```
Top ratings on Netflix:
rating
TV-MA    1822
TV-14    1214
R         778
PG-13     470
TV-PG     431
PG        275
TV-G       84
TV-Y7      76
TV-Y       76
NR         58
Name: count, dtype: int64
```

In [27]:
```
#plotting the top 10 ratings on Netflix.
plt.figure(figsize=(10,6))
rating_counts.head(10).plot(kind='bar', color = 'tomato')
plt.title('Most frequent Ratings on Netflix')
plt.xlabel('Rating')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




# Genres dominating US v/s Other countries .



In [28]:
```
# Drop missing values
df = df.dropna(subset=['listed_in', 'country'])
```

In [29]:
```
# Helper function to count genres
def get_top_genres(country_filter, top_n=10):
    filtered = df[df['country'].str.contains(country_filter, na=False)]
    genre_list = ', '.join(filtered['listed_in']).split(', ')
    genre_counts = Counter(genre_list)
    return pd.DataFrame(genre_counts.items(), columns=['Genre', 'Count']).sor
```

In [30]:
```
# Top genres in the US
us_genres = get_top_genres('United States')
```

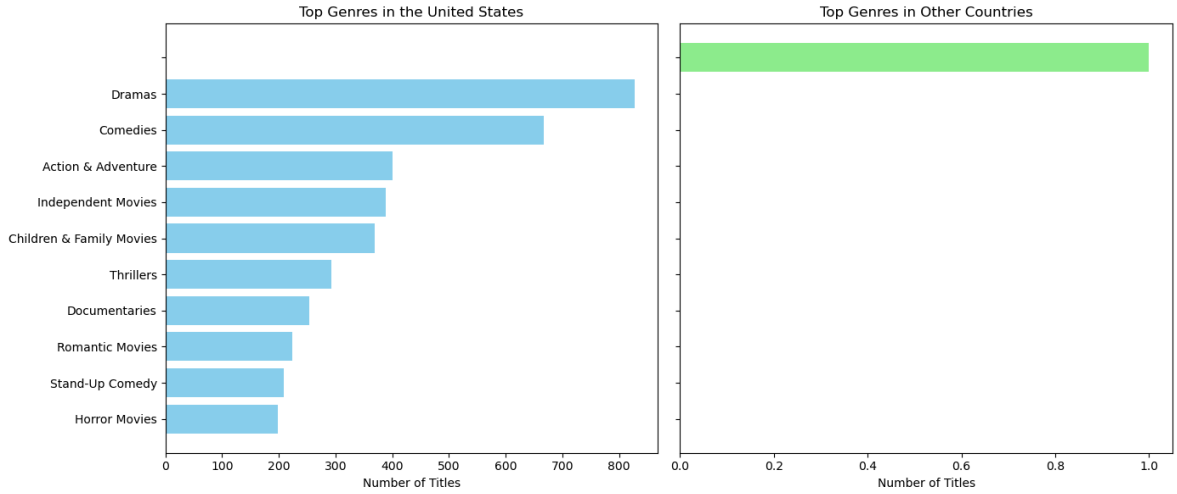
In [31]:
```
# Top genres in other countries (excluding US)
non_us_genres = get_top_genres('^(?!.United States).$', top_n=10)
```

In [32]:
```
# Plotting
fig, axes = plt.subplots(1, 2, figsize=(14, 6), sharey=True)

axes[0].barh(us_genres['Genre'][::-1], us_genres['Count'][::-1], color='skyblu
axes[0].set_title('Top Genres in the United States')
axes[0].set_xlabel('Number of Titles')

axes[1].barh(non_us_genres['Genre'][::-1], non_us_genres['Count'][::-1], colo
axes[1].set_title('Top Genres in Other Countries')
axes[1].set_xlabel('Number of Titles')

plt.tight_layout()
plt.show()
```



# Most popular genres in the last 3 years .

In [33]:
```
# Drop rows with missing genre
df = df.dropna(subset=['listed_in'])
```

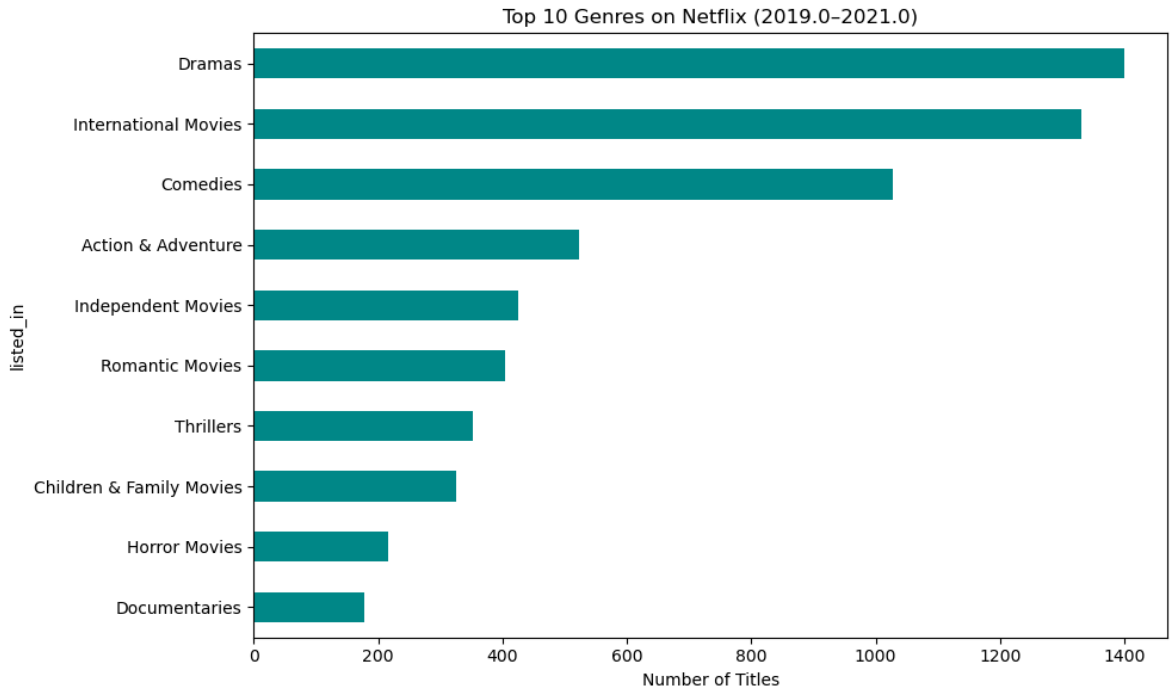
In [34]:
```
# Get max year in dataset (in case data is old)
max_year = df['date_added'].dt.year.max()
```

In [35]:
```
# Filter for the last 3 years from available data
recent_df = df[df['date_added'].dt.year >= (max_year - 2)]
```

In [36]:
```
# If there's data, proceed
if not recent_df.empty:
    # Explode genres into separate rows
    genres = recent_df['listed_in'].str.split(', ').explode()
```

In [37]:
```
# Count genres
genre_counts = genres.value_counts().head(10)
```

In [38]:
```
# Plot
plt.figure(figsize=(10,6))
genre_counts.plot(kind='barh', color='darkcyan')
plt.title(f"Top 10 Genres on Netflix ({max_year - 2}-{max_year})")
plt.xlabel('Number of Titles')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```




# Top 10 directors with the most Netflix content

.

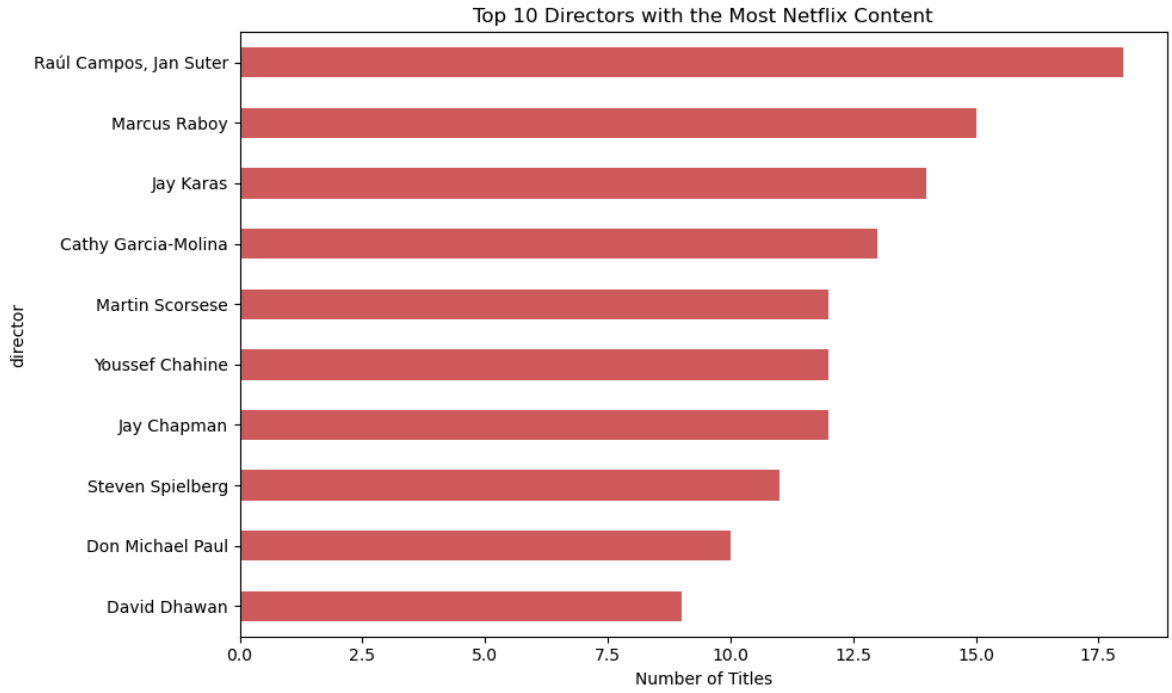
In [39]:
```
# Drop rows with missing director names
df = df.dropna(subset=['director'])
```

In [40]:
```
# Count occurrences of each director
director_counts = df['director'].value_counts().head(10)
```

In [41]:
```
# Print top directors
print("Top 10 Directors with Most Content on Netflix:")
print(director_counts)
```

```
Top 10 Directors with Most Content on Netflix:
director
Raúl Campos, Jan Suter    18
Marcus Raboy              15
Jay Karas                 14
Cathy Garcia-Molina       13
Martin Scorsese           12
Youssef Chahine           12
Jay Chapman               12
Steven Spielberg          11
Don Michael Paul          10
David Dhawan               9
Name: count, dtype: int64
```

In [42]:
```
# Plot
plt.figure(figsize=(10,6))
director_counts.plot(kind='barh', color='indianred')
plt.title('Top 10 Directors with the Most Netflix Content')
plt.xlabel('Number of Titles')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```




# Actors appearing most frequently on shows.

In [43]:
```
# Filter only TV Shows
tv_shows = df[df['type'] == 'TV Show']
```

In [44]:
```
# Drop missing cast data
tv_shows = tv_shows.dropna(subset=['cast'])
```

In [45]:
```
# Split and explode the cast list
actors = tv_shows['cast'].str.split(', ').explode()
```

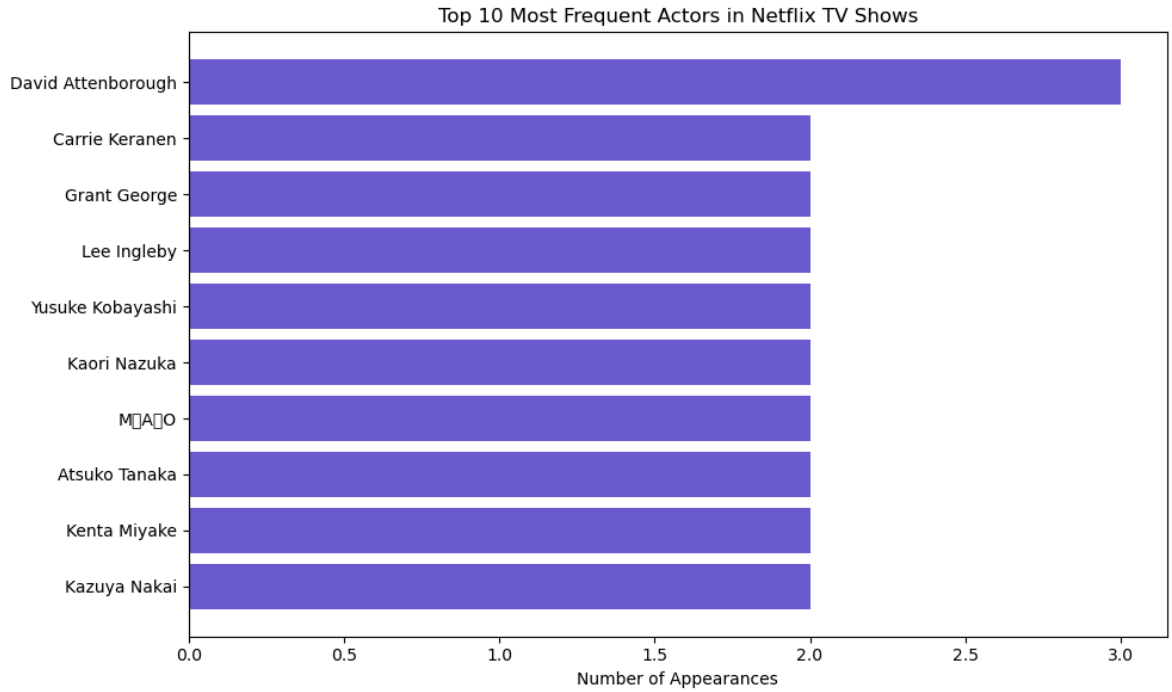
In [46]:
```
# Count actor appearances
actor_counts = Counter(actors)
top_actors = pd.DataFrame(actor_counts.items(), columns=['Actor', 'Count']).sc
```

In [47]:
```
# Print results
print("Top 10 Most Frequent Actors in Netflix TV Shows:")
print(top_actors)
```

```
Top 10 Most Frequent Actors in Netflix TV Shows:
                  Actor  Count
1181  David Attenborough      3
662       Carrie Keranen      2
673         Grant George      2
926          Lee Ingleby      2
964     Yusuke Kobayashi      2
967         Kaori Nazuka      2
968                M・A・O      2
972        Atsuko Tanaka      2
979         Kenta Miyake      2
981         Kazuya Nakai      2
```

In [48]:
```
# Plot
plt.figure(figsize=(10,6))
plt.barh(top_actors['Actor'][::-1], top_actors['Count'][::-1], color='slateblu
plt.title('Top 10 Most Frequent Actors in Netflix TV Shows')
plt.xlabel('Number of Appearances')
plt.tight_layout()
plt.show()
```

```
C:\Users\tripa\AppData\Local\Temp\ipykernel_40676\2285827395.py:6: UserWarni
ng: Glyph 12539 (\N{KATAKANA MIDDLE DOT}) missing from current font.
  plt.tight_layout()
C:\Users\tripa\anaconda3\Lib\site-packages\IPython\core\pylabtools.py:152: U
serWarning: Glyph 12539 (\N{KATAKANA MIDDLE DOT}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
```




# Average duration of shows on this platform.

In [49]:
```
# Filter only Movies
movies = df[df['type'] == 'Movie']
```

In [50]:
```
# Drop rows with missing duration
movies = movies.dropna(subset=['duration'])
```

In [51]:
```
# Extract number of minutes (e.g., from "100 min" → 100)
movies['minutes'] = movies['duration'].str.extract('(\d+)').astype(float)
```

In [52]:
```
# Calculate average duration
average_duration = movies['minutes'].mean()

print(f"Average duration of Movies on Netflix: {average_duration:.2f} minutes"
```

```
Average duration of Movies on Netflix: 102.70 minutes
```

# Most common number of seasons for TV shows on this platform.


In [53]:
```
# Filter only TV Shows
tv_shows = df[df['type'] == 'TV Show']
```

In [54]:
```
# Drop rows with missing duration
tv_shows = tv_shows.dropna(subset=['duration'])
```

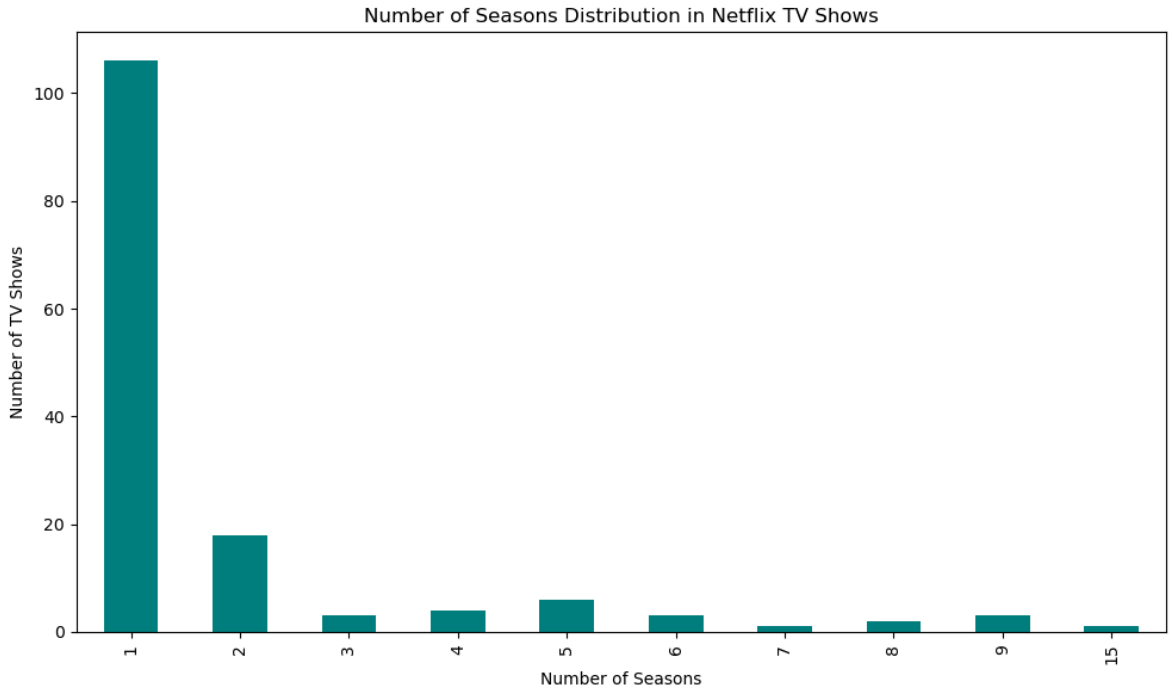
In [55]:
```
# Extract the number of seasons
tv_shows['seasons'] = tv_shows['duration'].str.extract('(\d+)').astype(int)
```

In [56]:
```
# Count frequency of each number of seasons
season_counts = tv_shows['seasons'].value_counts().sort_index()
```

In [57]:
```
# Print the most common number of seasons
most_common = season_counts.idxmax()
print(f"The most common number of seasons is: {most_common}")
```

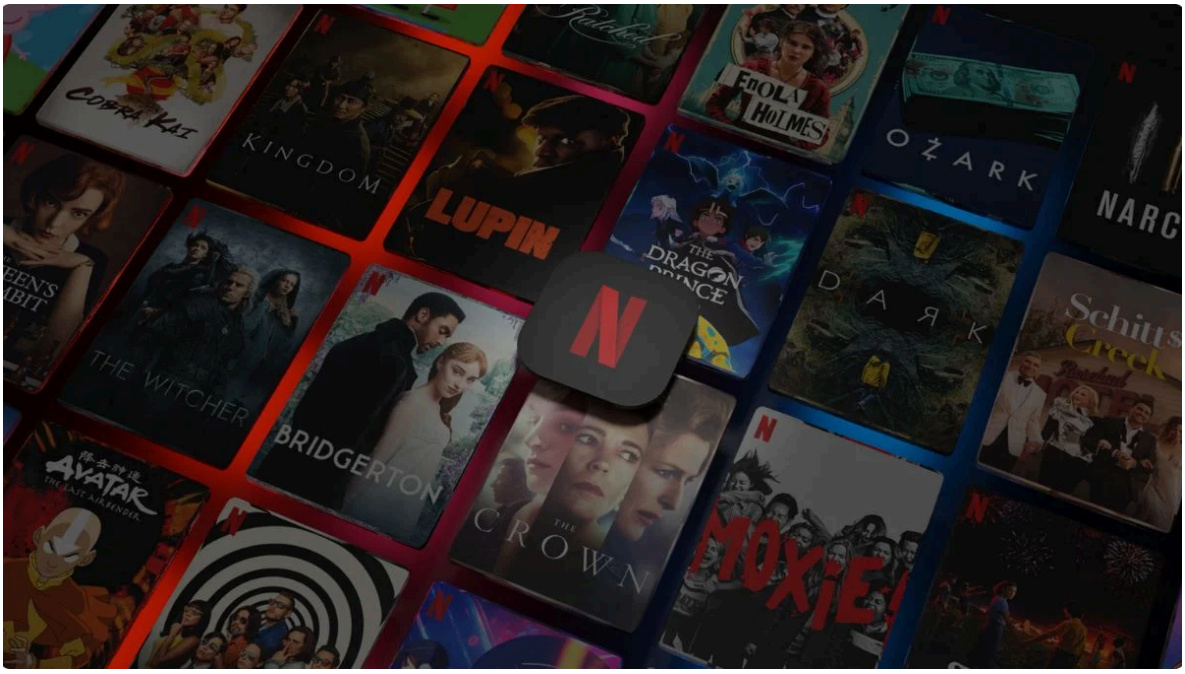
```
The most common number of seasons is: 1
```

In [58]:
```
# Plot
plt.figure(figsize=(10,6))
season_counts.plot(kind='bar', color='teal')
plt.title('Number of Seasons Distribution in Netflix TV Shows')
plt.xlabel('Number of Seasons')
plt.ylabel('Number of TV Shows')
plt.tight_layout()
plt.show()
```




# Months in which Netflix uploads most contents

.




In [59]:
```
# Convert 'date_added' to datetime
df['date_added'] = pd.to_datetime(df['date_added'], errors='coerce')
```

In [60]:
```
# Drop rows with missing dates
df = df.dropna(subset=['date_added'])
```

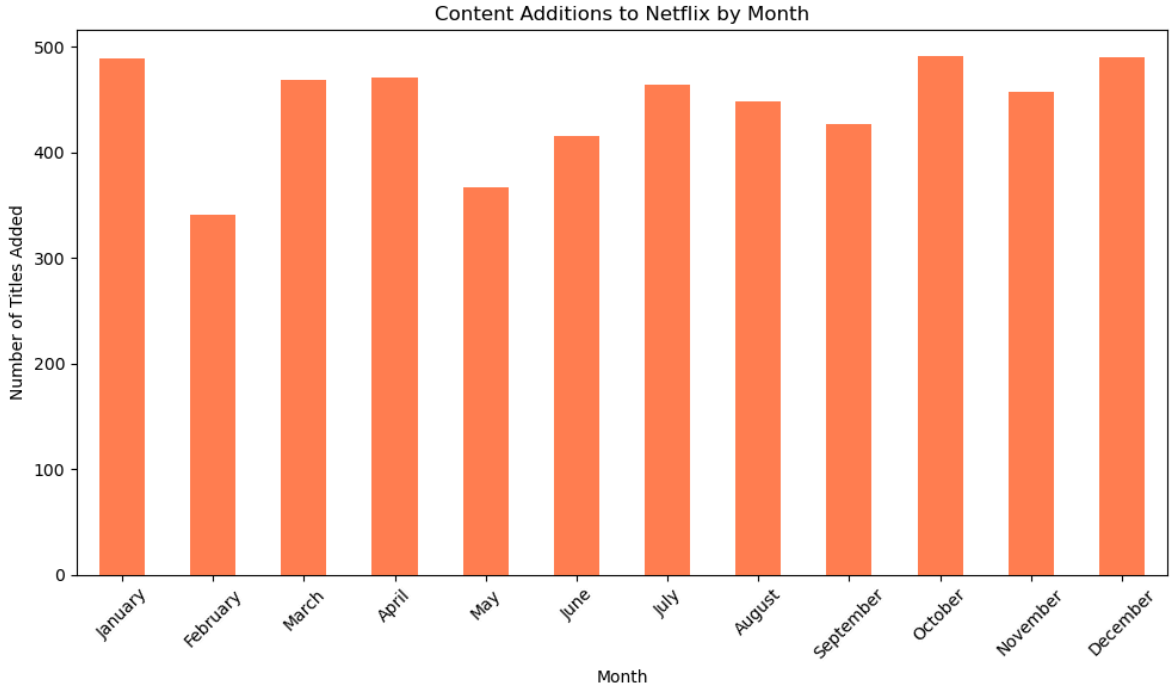
In [61]:
```
# Extract month name from 'date_added'
df['month_added'] = df['date_added'].dt.month_name()
```

In [62]:
```
# Count content added by month
month_counts = df['month_added'].value_counts().reindex([
    'January', 'February', 'March', 'April', 'May', 'June',
    'July', 'August', 'September', 'October', 'November', 'December'
])
```

In [63]:
```
# Print the month with the highest additions
most_common_month = month_counts.idxmax()
print(f"Month with most content added: {most_common_month}")
```

```
Month with most content added: October
```

In [64]:
```
# Plot
plt.figure(figsize=(10,6))
month_counts.plot(kind='bar', color='coral')
plt.title('Content Additions to Netflix by Month')
plt.xlabel('Month')
plt.ylabel('Number of Titles Added')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




# Countries that produce most content in each genre .


In [65]:
```
 #Drop missing values
df = df.dropna(subset=['country', 'listed_in'])
```

In [66]:
```
# Split and explode countries and genres
df['country'] = df['country'].str.split(', ')
df['listed_in'] = df['listed_in'].str.split(', ')
df = df.explode('country')
df = df.explode('listed_in')
```

In [67]:
```
# Clean column names
df['listed_in'] = df['listed_in'].str.strip()
df['country'] = df['country'].str.strip()
```

In [68]:
```
# Group by genre and country, count titles
genre_country_counts = df.groupby(['listed_in', 'country']).size().reset_index
```

In [69]:
```
# For each genre, get the country with the highest count
top_country_per_genre = genre_country_counts.sort_values(['listed_in', 'count'
                                                          .drop_duplicates('listed_in')
```

In [70]:
```
# Display results
print("Country with the most content for each genre:")
print(top_country_per_genre.sort_values(by='count', ascending=False).reset_inc
```

```
Country with the most content for each genre:
                      listed_in         country  count
0          International Movies           India    840
1                        Dramas   United States    827
2                       Comedies   United States    667
3            Action & Adventure   United States    401
4            Independent Movies   United States    389
5      Children & Family Movies   United States    369
6                      Thrillers   United States    292
7                  Documentaries   United States    254
8                Romantic Movies   United States    224
9                Stand-Up Comedy   United States    209
10                 Horror Movies   United States    199
11             Sci-Fi & Fantasy   United States    176
12             Music & Musicals   United States    128
13                Sports Movies   United States     91
14               Classic Movies   United States     74
15               Anime Features           Japan     60
16                  Cult Movies   United States     51
17                LGBTQ Movies   United States     49
18         Faith & Spirituality   United States     40
19             British TV Shows  United Kingdom     19
20               Crime TV Shows   United States     10
21                       Movies   United States     10
22       International TV Shows     South Korea     10
23              Korean TV Shows     South Korea      9
24                  TV Comedies   United States      9
25        TV Action & Adventure   United States      9
26                 Anime Series           Japan      8
27                    TV Dramas   United States      8
28  Stand-Up Comedy & Talk Shows   United States      7
29                   Docuseries   United States      6
30    Spanish-Language TV Shows           Spain      5
31             Romantic TV Shows     South Korea      5
32                     Kids' TV   United States      5
33          TV Sci-Fi & Fantasy   United States      3
34            Classic & Cult TV  United Kingdom      2
35                    TV Horror   United States      2
36                TV Mysteries          Taiwan      2
37                    TV Shows           India      2
38                  Reality TV          Brazil      1
39                TV Thrillers          Taiwan      1
40                Teen TV Shows          Canada      1
```


THANKYOU