Assignment Report
Student Name: Navneet P
Student ID: AF0411619

Problem Statement:

You are given a dataset which contains information about famous Hollywood movies. It includes attributes such as imdb ratings, director name, gross collection etc. Your task is to:

1. Download the dataset and remove any rows containing null data
2. Remove the special character present at the end of each movie title in the dataset.
3. Create a bar plot listing the movies made by the director James Cameron with proper labelling sorted by their IMDB ratings.
4. Extract the movie and the director's name with the following filters: *genre [Action], Language [English] ,Year [011-2015] ,rating [>7]* and create an excel file displaying the result.

You can use:

1. Numpy
2. Matplotlib
3. Pandas
4. PowerBI(optional)

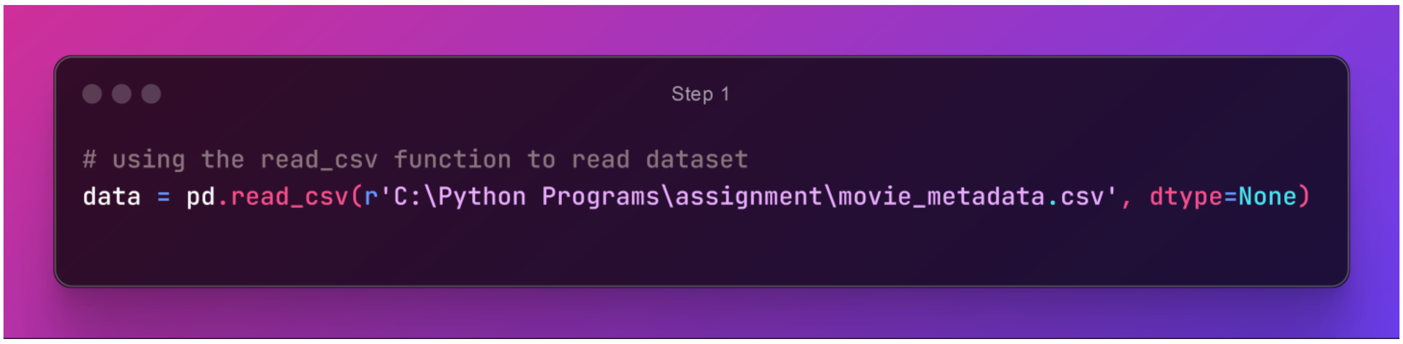
Expected Output:

1. A bar plot listing the movies made by James Cameron sorted by their IMDB ratings.
2. An excel file containing the filtered data according to the specified filters.

Solution:

Step 1: install pandas using pip install pandas. Once installed, read the dataset using the read_csv() method.

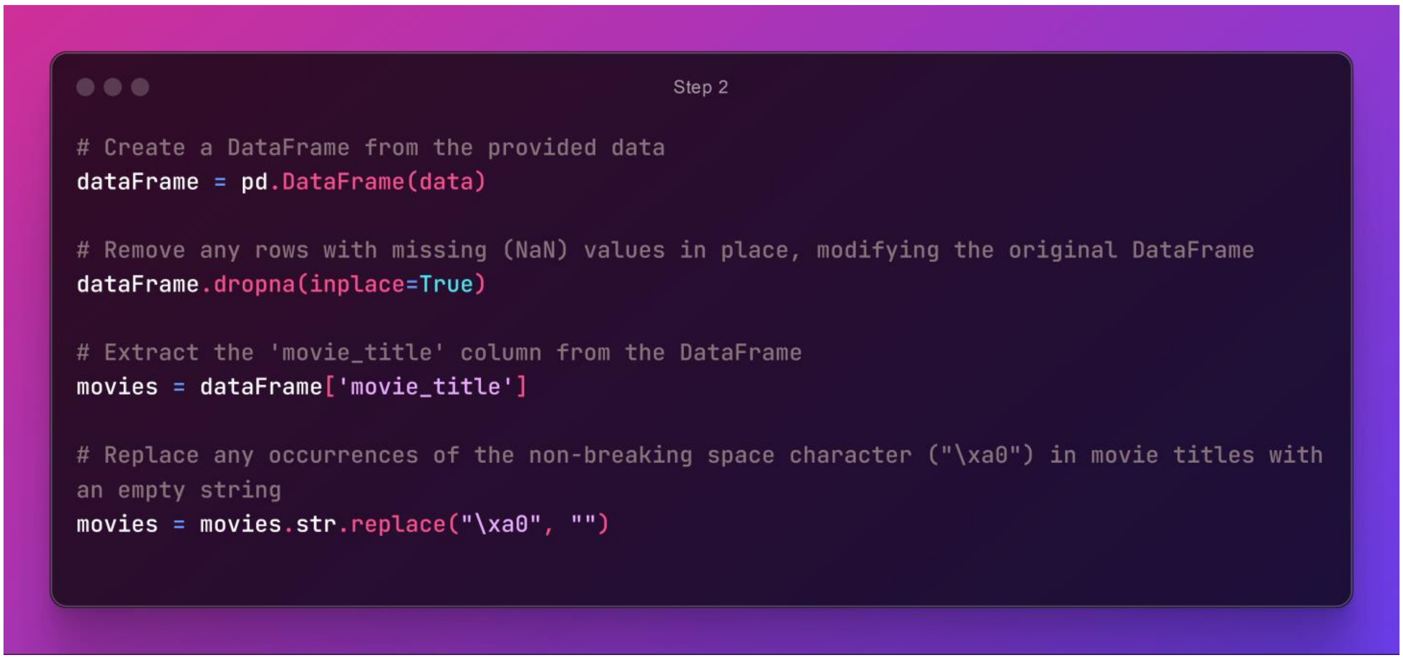
Code:



```
# using the read_csv function to read dataset
data = pd.read_csv(r'C:\Python Programs\assignment\movie_metadata.csv', dtype=None)
```

Step 2: In the movie title column, remove the unnecessary character. Also, remove the rows containing null values(NaN). [Dropna]

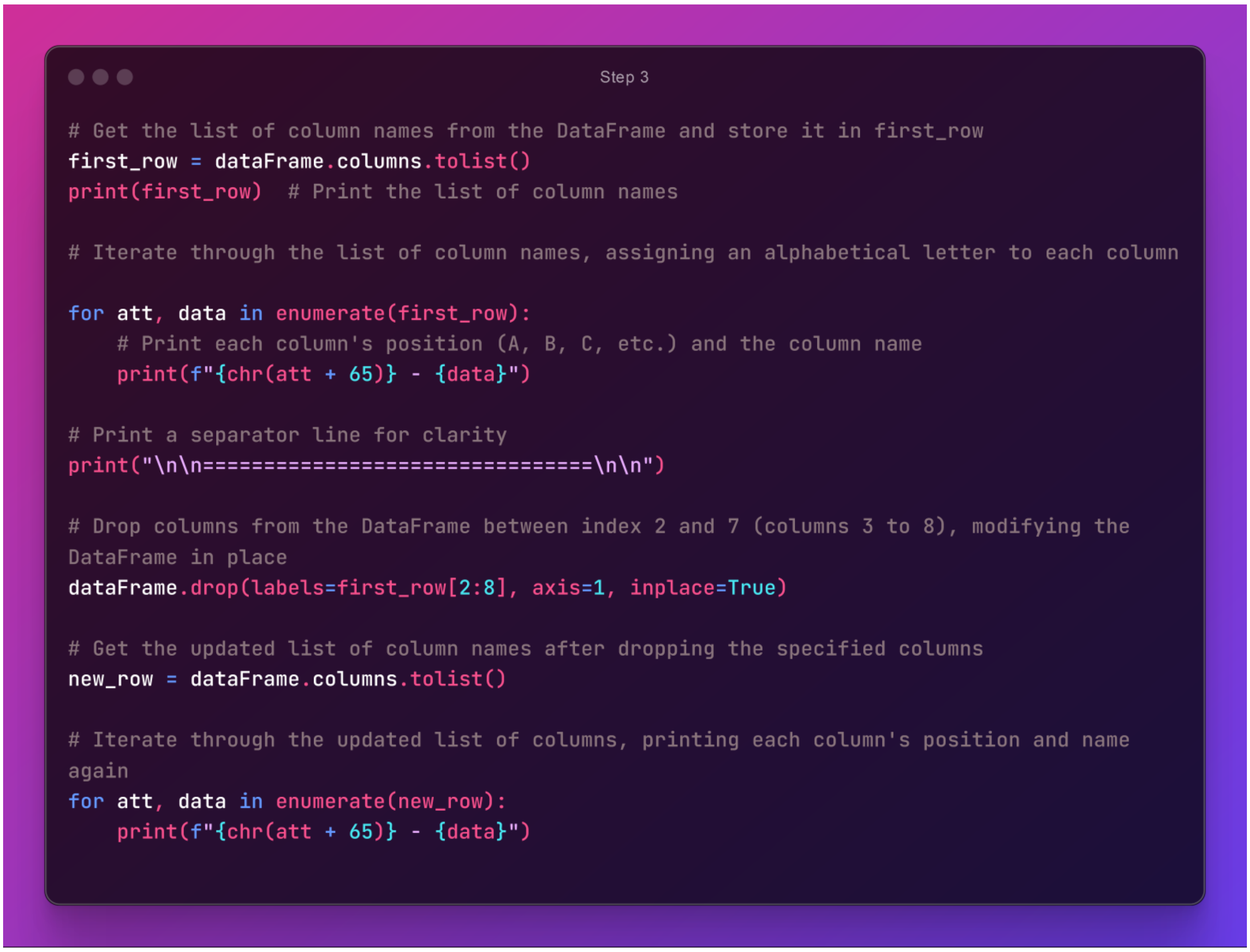
After analysis, it turns out the character present at the end of each Movie title is “\xa0”.



```
# Create a DataFrame from the provided data
dataFrame = pd.DataFrame(data)

# Remove any rows with missing (NaN) values in place, modifying the original DataFrame
dataFrame.dropna(inplace=True)

# Extract the 'movie_title' column from the DataFrame
movies = dataFrame['movie_title']

# Replace any occurrences of the non-breaking space character ("\xa0") in movie titles with
an empty string
movies = movies.str.replace("\xa0", "")
```

Step 3: Remove columns C – H

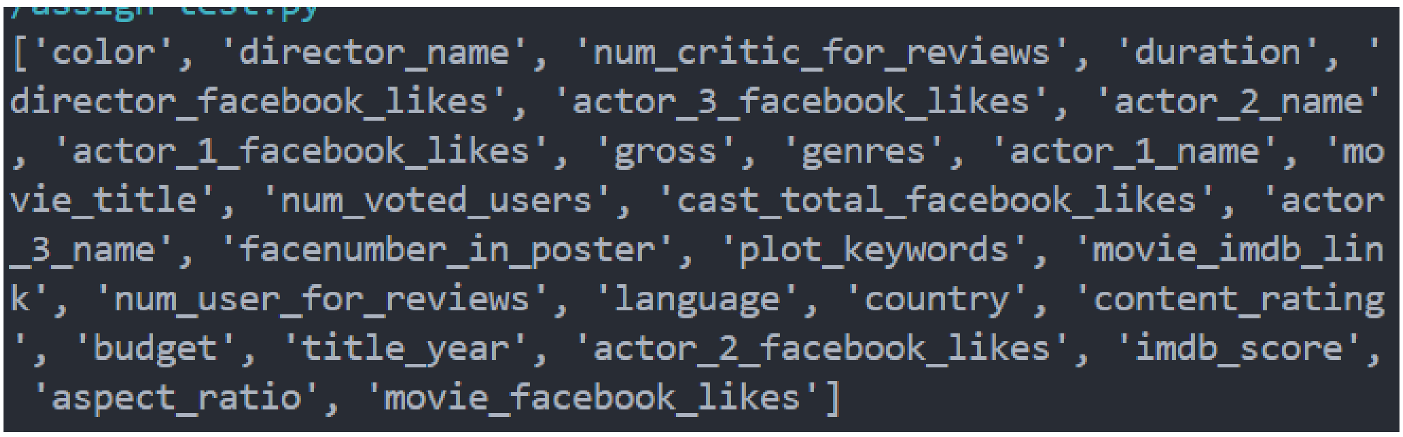
Since, we do not want to go back to the excel file again and again to look for the corresponding attributes to be removed, we will simply obtain the attribute columns using pandas. [Character Indexing] [Drop Columns]



Step 3

```python
# Get the list of column names from the DataFrame and store it in first_row
first_row = dataFrame.columns.tolist()
print(first_row)  # Print the list of column names

# Iterate through the list of column names, assigning an alphabetical letter to each column

for att, data in enumerate(first_row):
    # Print each column's position (A, B, C, etc.) and the column name
    print(f"{chr(att + 65)} - {data}")

# Print a separator line for clarity
print("\n\n================================\n\n")

# Drop columns from the DataFrame between index 2 and 7 (columns 3 to 8), modifying the
DataFrame in place
dataFrame.drop(labels=first_row[2:8], axis=1, inplace=True)

# Get the updated list of column names after dropping the specified columns
new_row = dataFrame.columns.tolist()

# Iterate through the updated list of columns, printing each column's position and name
again
for att, data in enumerate(new_row):
    print(f"{chr(att + 65)} - {data}")
```

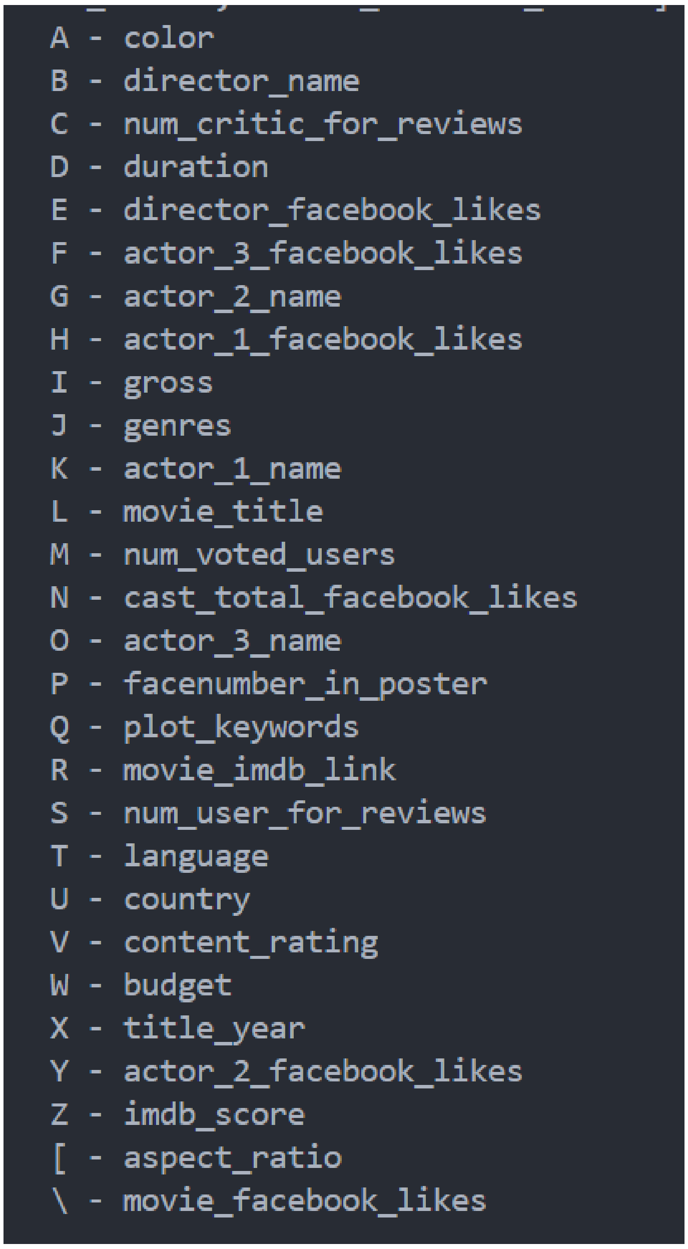
Output: List of all attributes



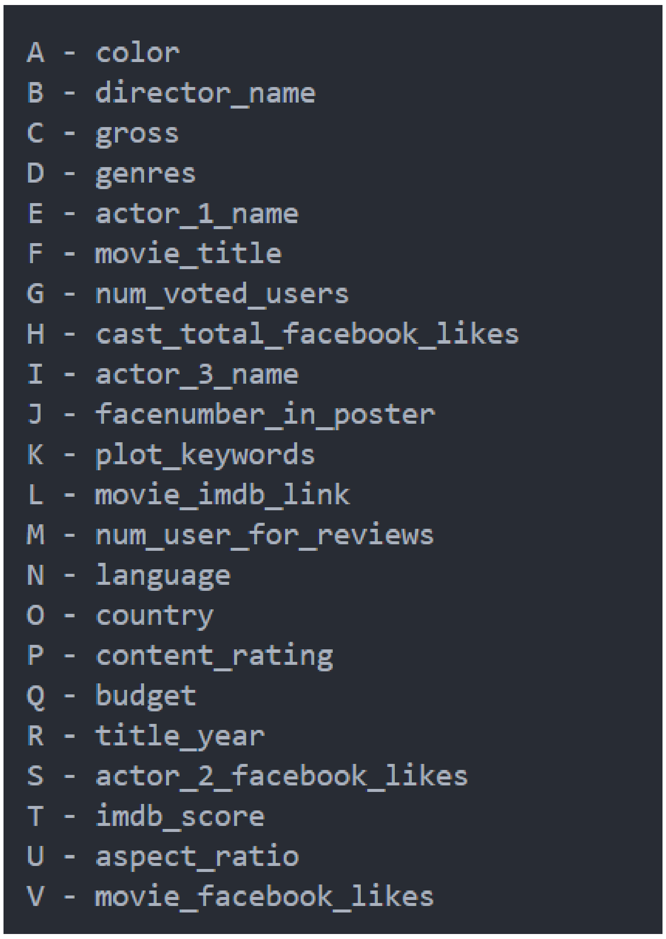
```
['color', 'director_name', 'num_critic_for_reviews', 'duration', '
director_facebook_likes', 'actor_3_facebook_likes', 'actor_2_name'
, 'actor_1_facebook_likes', 'gross', 'genres', 'actor_1_name', 'mo
vie_title', 'num_voted_users', 'cast_total_facebook_likes', 'actor
_3_name', 'facenumber_in_poster', 'plot_keywords', 'movie_imdb_lin
k', 'num_user_for_reviews', 'language', 'country', 'content_rating
', 'budget', 'title_year', 'actor_2_facebook_likes', 'imdb_score',
 'aspect_ratio', 'movie_facebook_likes']
```

Before columns were removed

After Columns are removed



```
A - color
B - director_name
C - num_critic_for_reviews
D - duration
E - director_facebook_likes
F - actor_3_facebook_likes
G - actor_2_name
H - actor_1_facebook_likes
I - gross
J - genres
K - actor_1_name
L - movie_title
M - num_voted_users
N - cast_total_facebook_likes
O - actor_3_name
P - facenumber_in_poster
Q - plot_keywords
R - movie_imdb_link
S - num_user_for_reviews
T - language
U - country
V - content_rating
W - budget
X - title_year
Y - actor_2_facebook_likes
Z - imdb_score
[ - aspect_ratio
\ - movie_facebook_likes
```

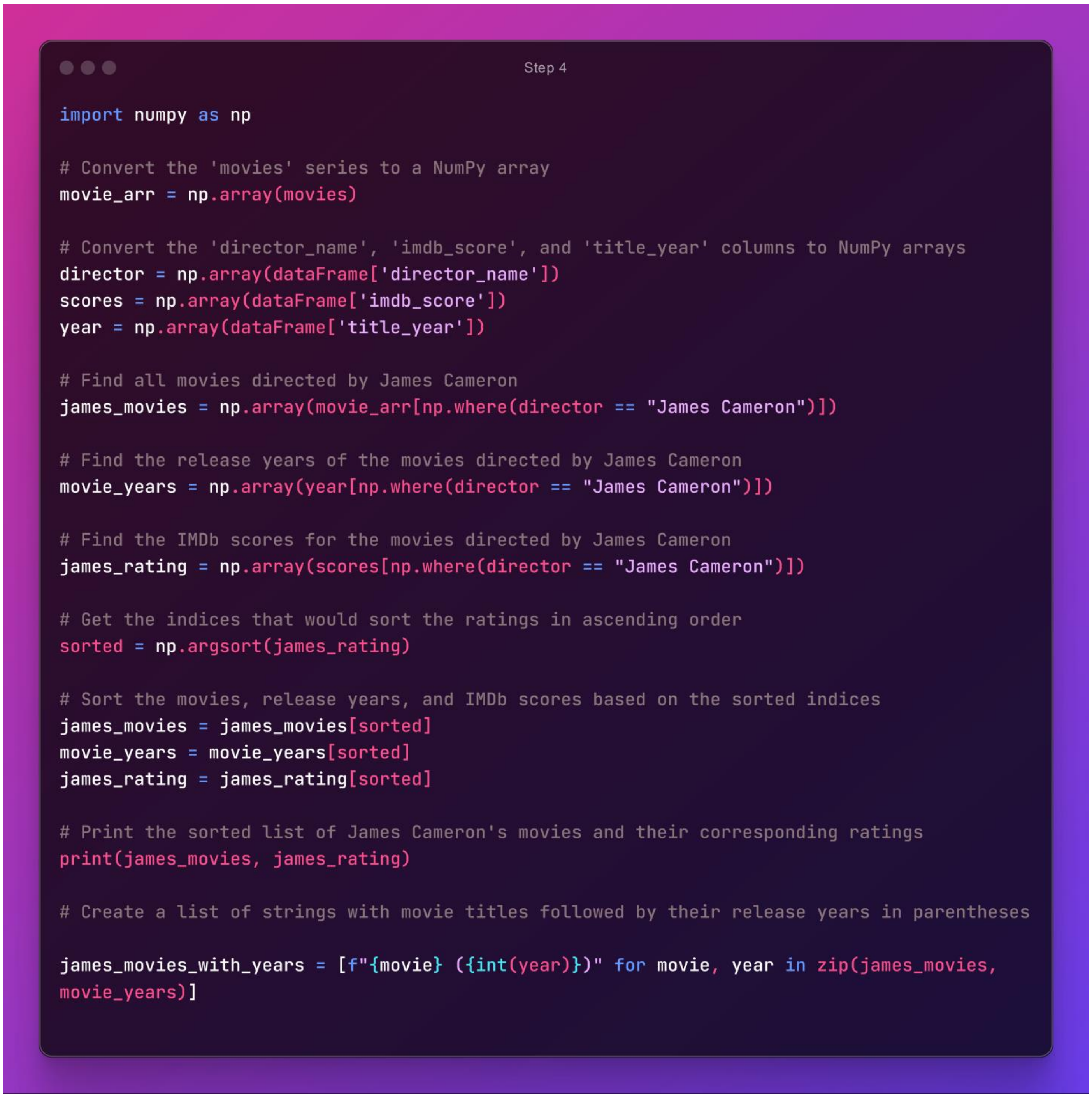


```
A - color
B - director_name
C - gross
D - genres
E - actor_1_name
F - movie_title
G - num_voted_users
H - cast_total_facebook_likes
I - actor_3_name
J - facenumber_in_poster
K - plot_keywords
L - movie_imdb_link
M - num_user_for_reviews
N - language
O - country
P - content_rating
Q - budget
R - title_year
S - actor_2_facebook_likes
T - imdb_score
U - aspect_ratio
V - movie_facebook_likes
```

With this, the columns from C-H have been dropped.

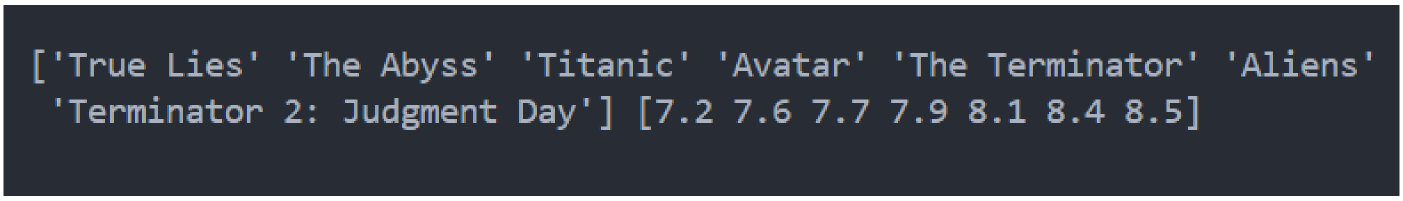
Step 4: Filter director_name column with “James Cameron” and construct a bar plot where all his movies are sorted according to IMDB ratings. [Zip Func]

Creating the required arrays first…



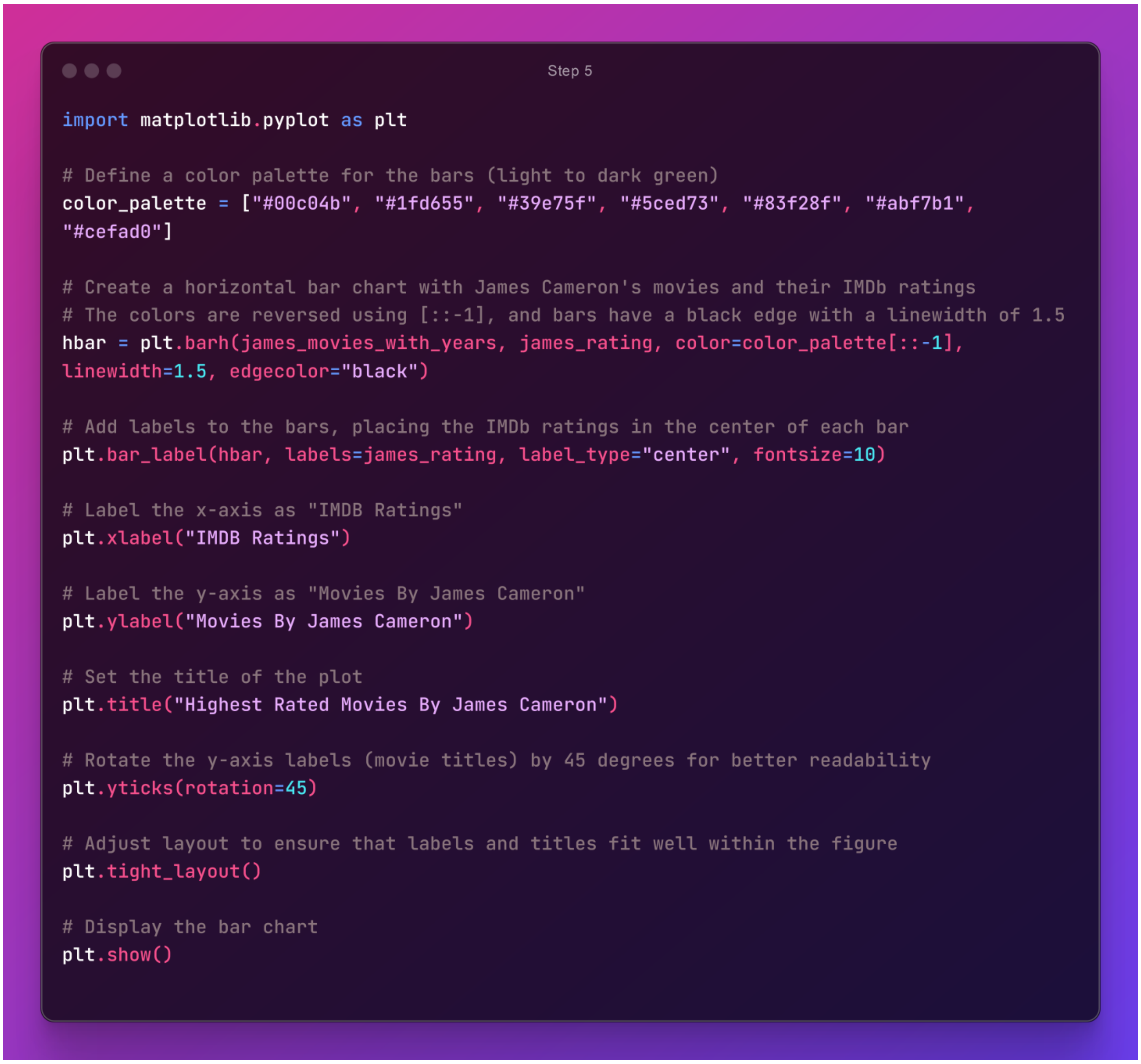
```python
import numpy as np

# Convert the 'movies' series to a NumPy array
movie_arr = np.array(movies)

# Convert the 'director_name', 'imdb_score', and 'title_year' columns to NumPy arrays
director = np.array(dataFrame['director_name'])
scores = np.array(dataFrame['imdb_score'])
year = np.array(dataFrame['title_year'])

# Find all movies directed by James Cameron
james_movies = np.array(movie_arr[np.where(director == "James Cameron")])

# Find the release years of the movies directed by James Cameron
movie_years = np.array(year[np.where(director == "James Cameron")])

# Find the IMDb scores for the movies directed by James Cameron
james_rating = np.array(scores[np.where(director == "James Cameron")])

# Get the indices that would sort the ratings in ascending order
sorted = np.argsort(james_rating)

# Sort the movies, release years, and IMDb scores based on the sorted indices
james_movies = james_movies[sorted]
movie_years = movie_years[sorted]
james_rating = james_rating[sorted]

# Print the sorted list of James Cameron's movies and their corresponding ratings
print(james_movies, james_rating)

# Create a list of strings with movie titles followed by their release years in parentheses

james_movies_with_years = [f"{movie} ({int(year)})" for movie, year in zip(james_movies,
movie_years)]
```

Output: Movies array and the ratings array



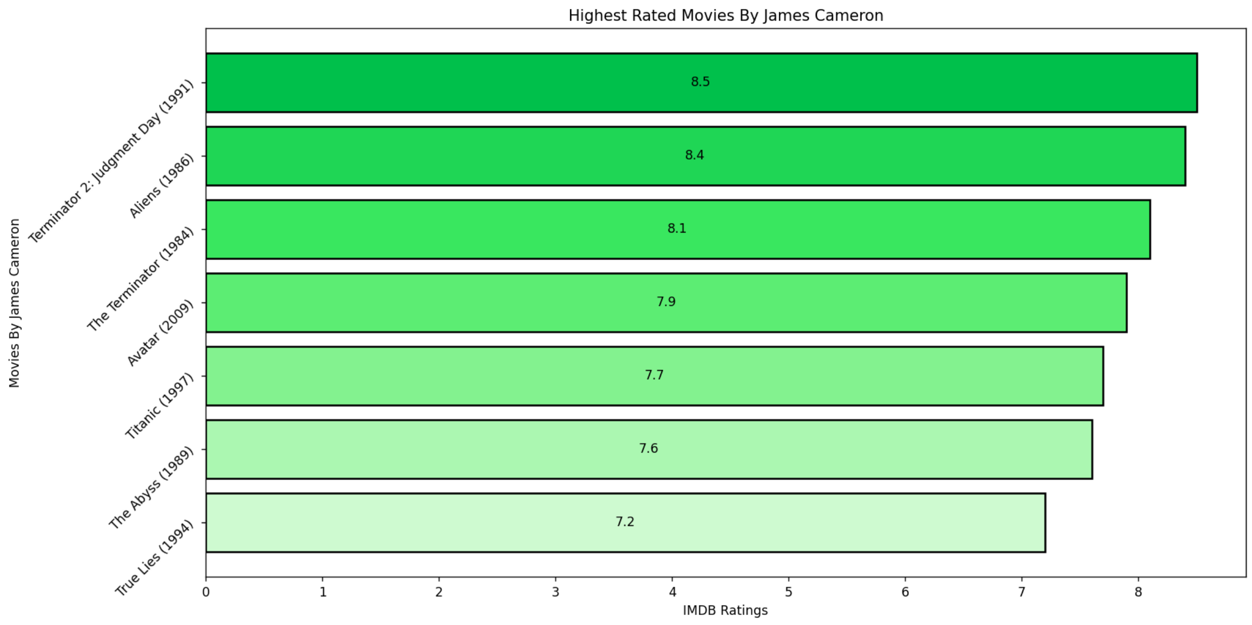
```
['True Lies' 'The Abyss' 'Titanic' 'Avatar' 'The Terminator' 'Aliens'
 'Terminator 2: Judgment Day'] [7.2 7.6 7.7 7.9 8.1 8.4 8.5]
```

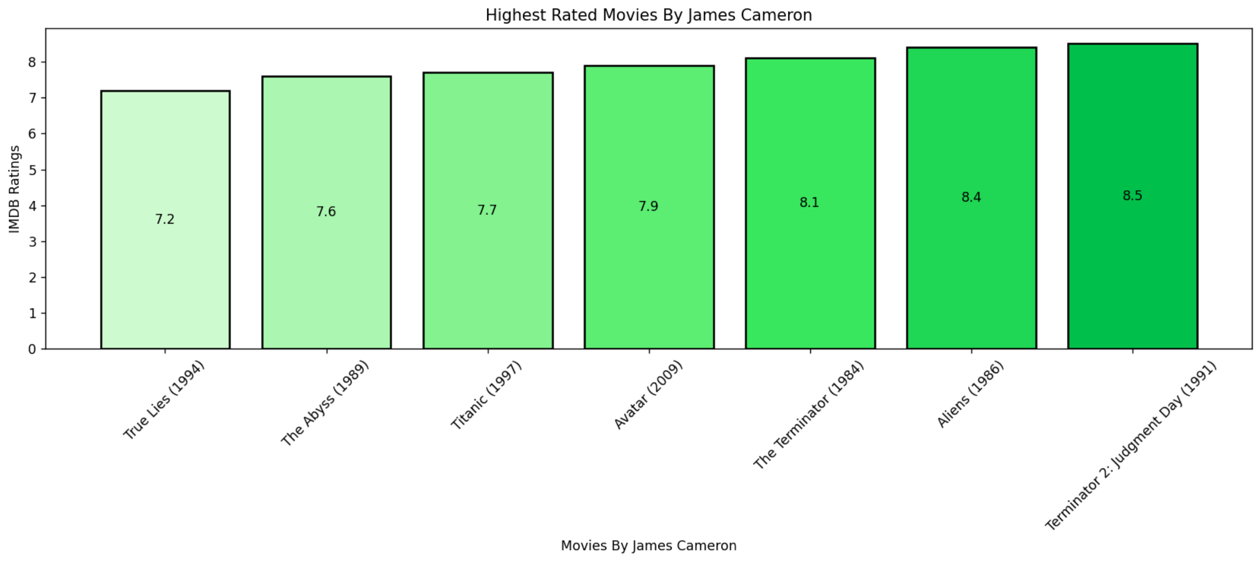
Step 5: Construct a bar plot using this data.



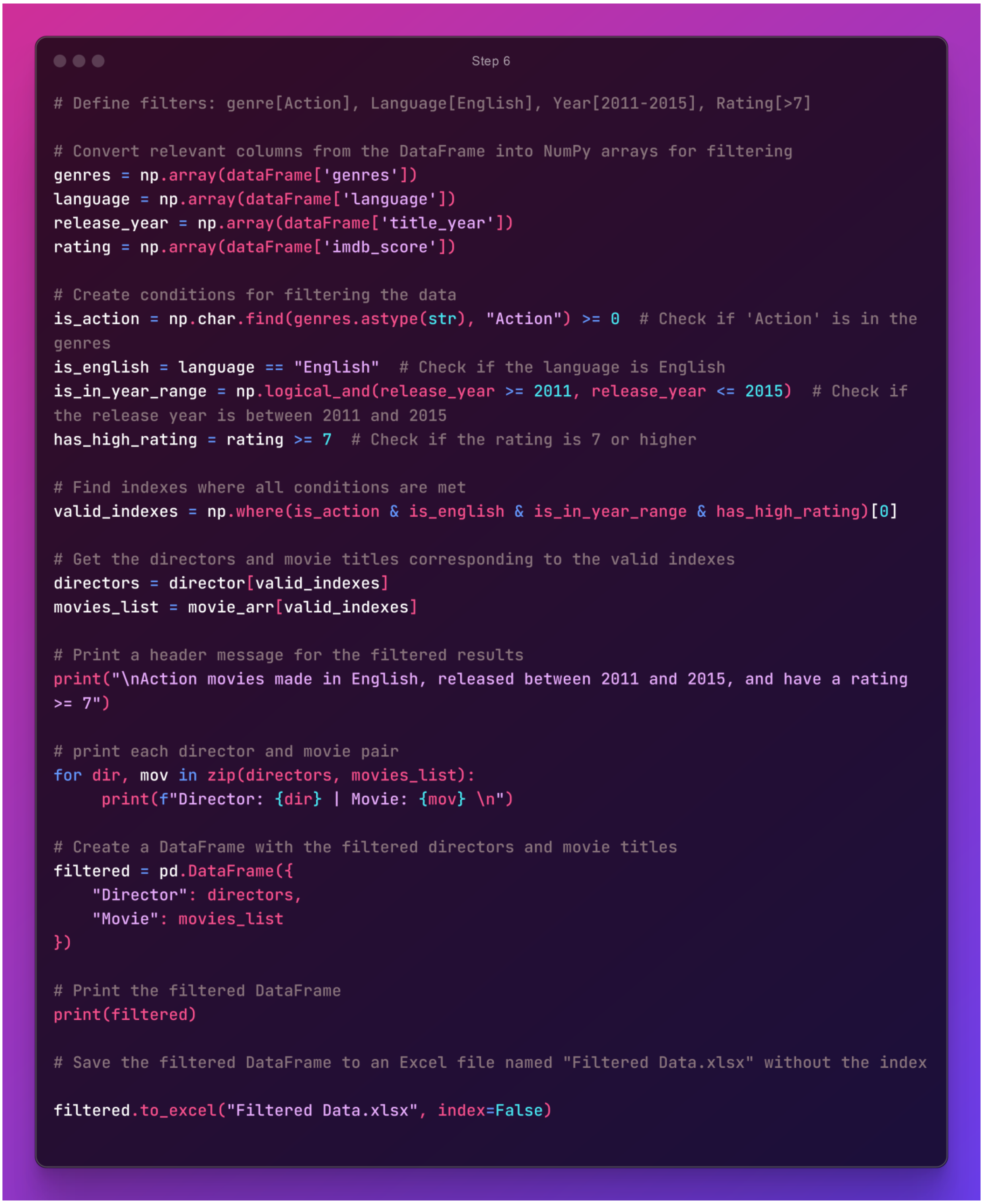
```python
import matplotlib.pyplot as plt

# Define a color palette for the bars (light to dark green)
color_palette = ["#00c04b", "#1fd655", "#39e75f", "#5ced73", "#83f28f", "#abf7b1",
"#cefad0"]

# Create a horizontal bar chart with James Cameron's movies and their IMDb ratings
# The colors are reversed using [::-1], and bars have a black edge with a linewidth of 1.5
hbar = plt.barh(james_movies_with_years, james_rating, color=color_palette[::-1],
linewidth=1.5, edgecolor="black")

# Add labels to the bars, placing the IMDb ratings in the center of each bar
plt.bar_label(hbar, labels=james_rating, label_type="center", fontsize=10)

# Label the x-axis as "IMDB Ratings"
plt.xlabel("IMDB Ratings")

# Label the y-axis as "Movies By James Cameron"
plt.ylabel("Movies By James Cameron")

# Set the title of the plot
plt.title("Highest Rated Movies By James Cameron")

# Rotate the y-axis labels (movie titles) by 45 degrees for better readability
plt.yticks(rotation=45)

# Adjust layout to ensure that labels and titles fit well within the figure
plt.tight_layout()

# Display the bar chart
plt.show()
```

Output:


Highest Rated Movies By James Cameron
Movies By James Cameron
IMDB Ratings
Terminator 2: Judgment Day (1991)
Aliens (1986)
The Terminator (1984)
Avatar (2009)
Titanic (1997)
The Abyss (1989)
True Lies (1994)
8.5
8.4
8.1
7.9
7.7
7.6
7.2
0
1
2
3
4
5
6
7
8


Highest Rated Movies By James Cameron
IMDB Ratings
Movies By James Cameron
True Lies (1994)
The Abyss (1989)
Titanic (1997)
Avatar (2009)
The Terminator (1984)
Aliens (1986)
Terminator 2: Judgment Day (1991)
7.2
7.6
7.7
7.9
8.1
8.4
8.5
0
1
2
3
4
5
6
7
8

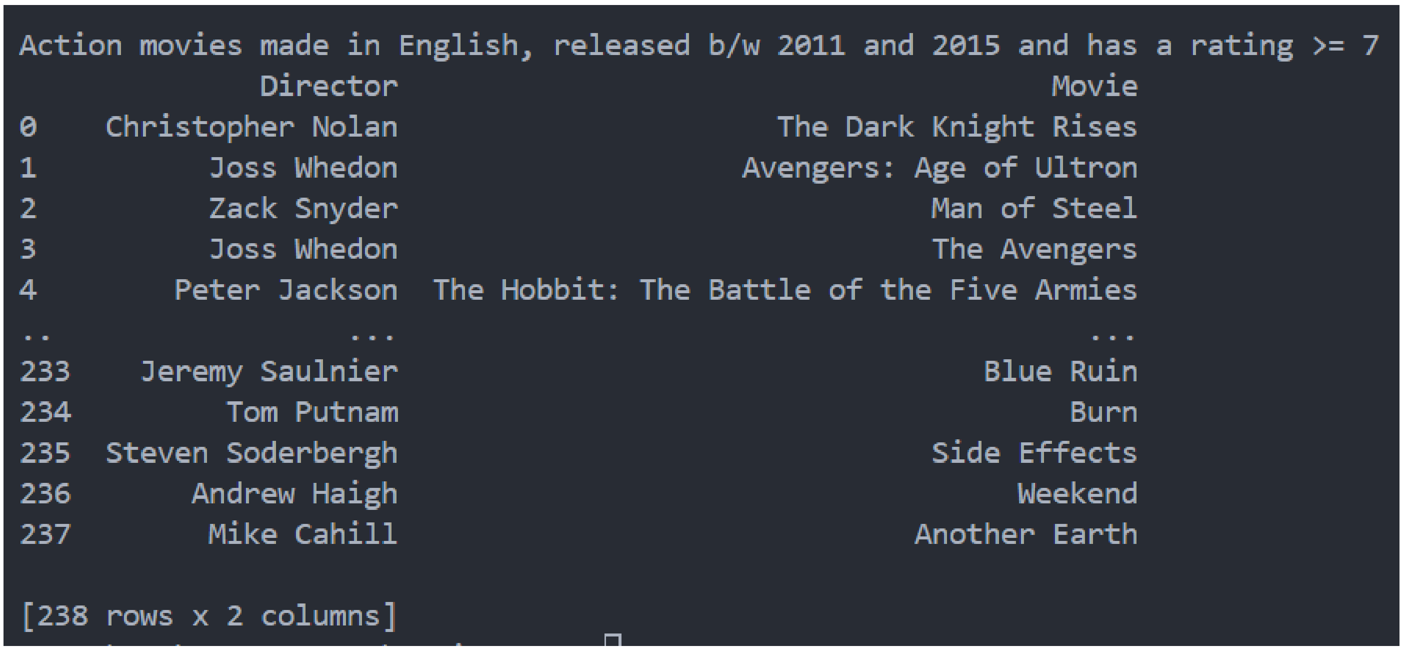
Step 6: Obtain the filtered data according to the requirements and exporting it into an excel file.



Step 6

```python
# Define filters: genre[Action], Language[English], Year[2011-2015], Rating[>7]

# Convert relevant columns from the DataFrame into NumPy arrays for filtering
genres = np.array(dataFrame['genres'])
language = np.array(dataFrame['language'])
release_year = np.array(dataFrame['title_year'])
rating = np.array(dataFrame['imdb_score'])

# Create conditions for filtering the data
is_action = np.char.find(genres.astype(str), "Action") >= 0  # Check if 'Action' is in the genres
is_english = language == "English"  # Check if the language is English
is_in_year_range = np.logical_and(release_year >= 2011, release_year <= 2015)  # Check if the release year is between 2011 and 2015
has_high_rating = rating >= 7  # Check if the rating is 7 or higher

# Find indexes where all conditions are met
valid_indexes = np.where(is_action & is_english & is_in_year_range & has_high_rating)[0]

# Get the directors and movie titles corresponding to the valid indexes
directors = director[valid_indexes]
movies_list = movie_arr[valid_indexes]

# Print a header message for the filtered results
print("\nAction movies made in English, released between 2011 and 2015, and have a rating >= 7")

# print each director and movie pair
for dir, mov in zip(directors, movies_list):
     print(f"Director: {dir} | Movie: {mov} \n")

# Create a DataFrame with the filtered directors and movie titles
filtered = pd.DataFrame({
    "Director": directors,
    "Movie": movies_list
})

# Print the filtered DataFrame
print(filtered)

# Save the filtered DataFrame to an Excel file named "Filtered Data.xlsx" without the index

filtered.to_excel("Filtered Data.xlsx", index=False)
```

Output:



```
Action movies made in English, released b/w 2011 and 2015 and has a rating >= 7
              Director                                      Movie
0    Christopher Nolan                      The Dark Knight Rises
1          Joss Whedon                    Avengers: Age of Ultron
2          Zack Snyder                               Man of Steel
3          Joss Whedon                               The Avengers
4        Peter Jackson  The Hobbit: The Battle of the Five Armies
..                 ...                                        ...
233    Jeremy Saulnier                                  Blue Ruin
234         Tom Putnam                                       Burn
235  Steven Soderbergh                               Side Effects
236       Andrew Haigh                                    Weekend
237        Mike Cahill                              Another Earth

[238 rows x 2 columns]
```

Showcase of Filtered Data.xlsx => Link