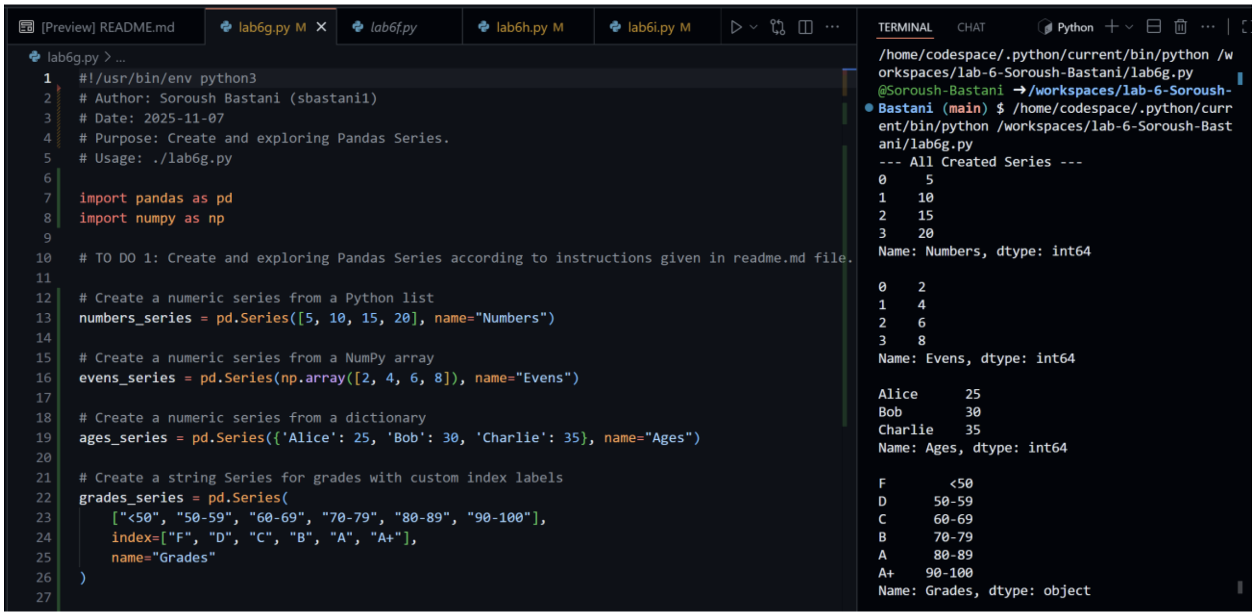
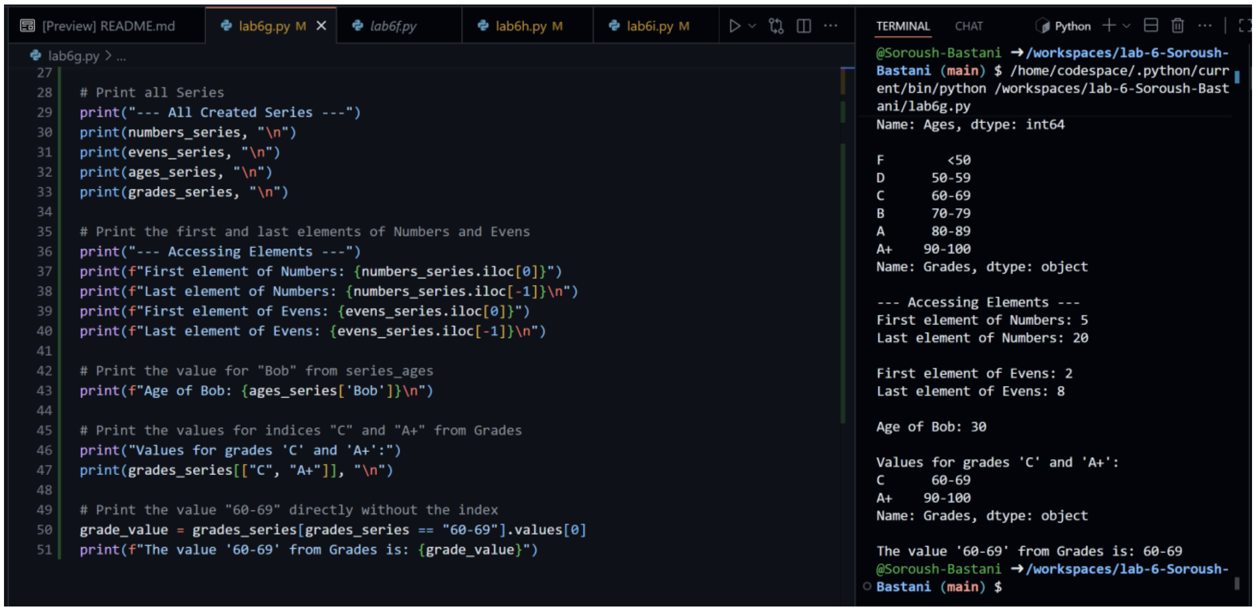
Lab 6 - Investigation 2

Lab6g



```python
#!/usr/bin/env python3
# Author: Soroush Bastani (sbastani1)
# Date: 2025-11-07
# Purpose: Create and exploring Pandas Series.
# Usage: ./lab6g.py

import pandas as pd
import numpy as np

# TO DO 1: Create and exploring Pandas Series according to instructions given in readme.md file.

# Create a numeric series from a Python list
numbers_series = pd.Series([5, 10, 15, 20], name="Numbers")

# Create a numeric series from a NumPy array
evens_series = pd.Series(np.array([2, 4, 6, 8]), name="Evens")

# Create a numeric series from a dictionary
ages_series = pd.Series({'Alice': 25, 'Bob': 30, 'Charlie': 35}, name="Ages")

# Create a string Series for grades with custom index labels
grades_series = pd.Series(
    ["<50", "50-59", "60-69", "70-79", "80-89", "90-100"],
    index=["F", "D", "C", "B", "A", "A+"],
    name="Grades"
)
```

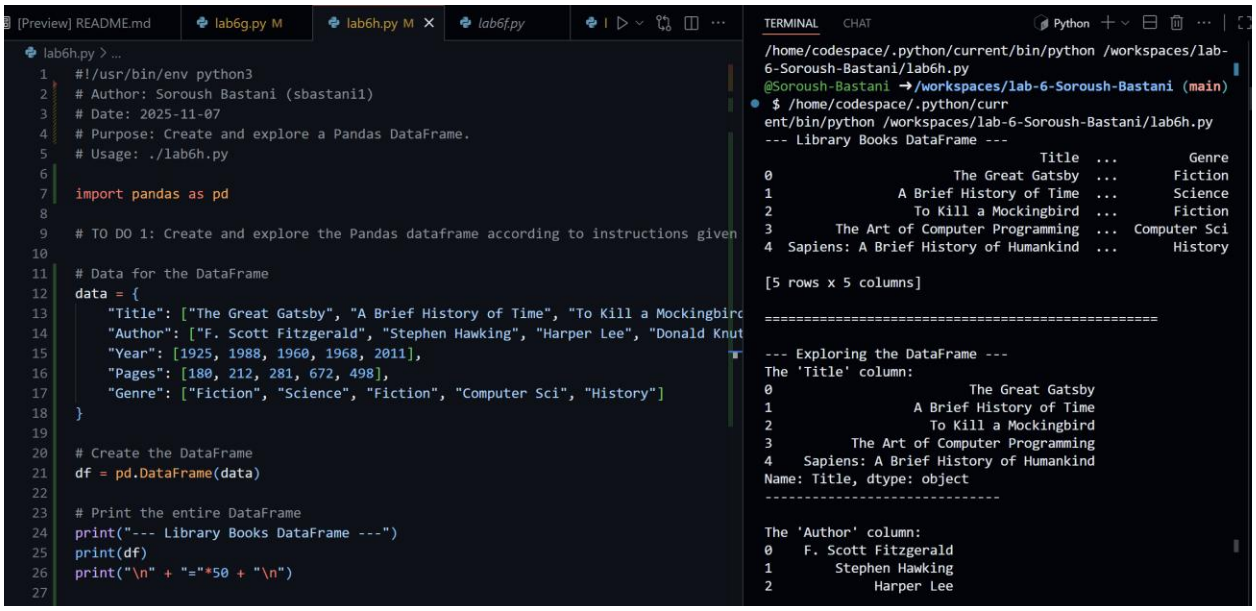
```
/home/codespace/.python/current/bin/python /workspaces/lab-6-Soroush-Bastani/lab6g.py
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main) $ /home/codespace/.python/current/bin/python /workspaces/lab-6-Soroush-Bastani/lab6g.py
--- All Created Series ---
0     5
1    10
2    15
3    20
Name: Numbers, dtype: int64

0    2
1    4
2    6
3    8
Name: Evens, dtype: int64

Alice      25
Bob        30
Charlie    35
Name: Ages, dtype: int64

F        <50
D      50-59
C      60-69
B      70-79
A      80-89
A+    90-100
Name: Grades, dtype: object
```



```python

# Print all Series
print("--- All Created Series ---")
print(numbers_series, "\n")
print(evens_series, "\n")
print(ages_series, "\n")
print(grades_series, "\n")

# Print the first and last elements of Numbers and Evens
print("--- Accessing Elements ---")
print(f"First element of Numbers: {numbers_series.iloc[0]}")
print(f"Last element of Numbers: {numbers_series.iloc[-1]}\n")
print(f"First element of Evens: {evens_series.iloc[0]}")
print(f"Last element of Evens: {evens_series.iloc[-1]}\n")

# Print the value for "Bob" from series_ages
print(f"Age of Bob: {ages_series['Bob']}\n")

# Print the values for indices "C" and "A+" from Grades
print("Values for grades 'C' and 'A+':")
print(grades_series[["C", "A+"]], "\n")

# Print the value "60-69" directly without the index
grade_value = grades_series[grades_series == "60-69"].values[0]
print(f"The value '60-69' from Grades is: {grade_value}")
```

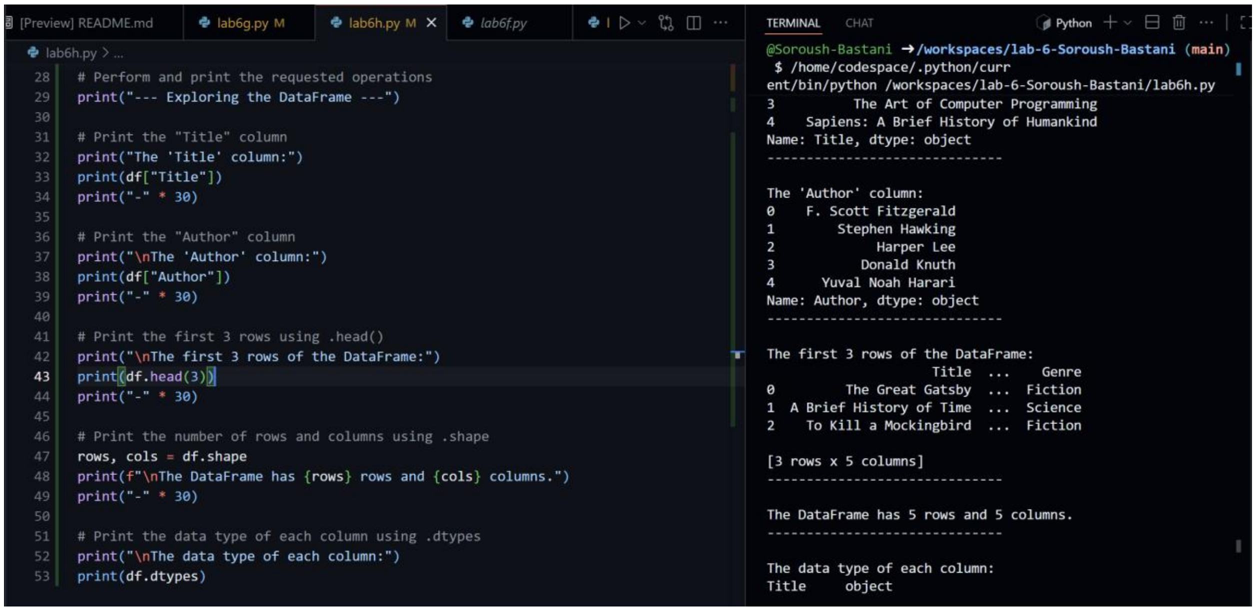
```
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main) $ /home/codespace/.python/current/bin/python /workspaces/lab-6-Soroush-Bastani/lab6g.py
Name: Ages, dtype: int64

F        <50
D      50-59
C      60-69
B      70-79
A      80-89
A+    90-100
Name: Grades, dtype: object

--- Accessing Elements ---
First element of Numbers: 5
Last element of Numbers: 20

First element of Evens: 2
Last element of Evens: 8

Age of Bob: 30

Values for grades 'C' and 'A+':
C      60-69
A+    90-100
Name: Grades, dtype: object

The value '60-69' from Grades is: 60-69
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main) $
```

Lab6h



```python
#!/usr/bin/env python3
# Author: Soroush Bastani (sbastani1)
# Date: 2025-11-07
# Purpose: Create and explore a Pandas DataFrame.
# Usage: ./lab6h.py

import pandas as pd

# TO DO 1: Create and explore the Pandas dataframe according to instructions given

# Data for the DataFrame
data = {
    "Title": ["The Great Gatsby", "A Brief History of Time", "To Kill a Mockingbirc
    "Author": ["F. Scott Fitzgerald", "Stephen Hawking", "Harper Lee", "Donald Knut
    "Year": [1925, 1988, 1960, 1968, 2011],
    "Pages": [180, 212, 281, 672, 498],
    "Genre": ["Fiction", "Science", "Fiction", "Computer Sci", "History"]
}

# Create the DataFrame
df = pd.DataFrame(data)

# Print the entire DataFrame
print("--- Library Books DataFrame ---")
print(df)
print("\n" + "="*50 + "\n")

```

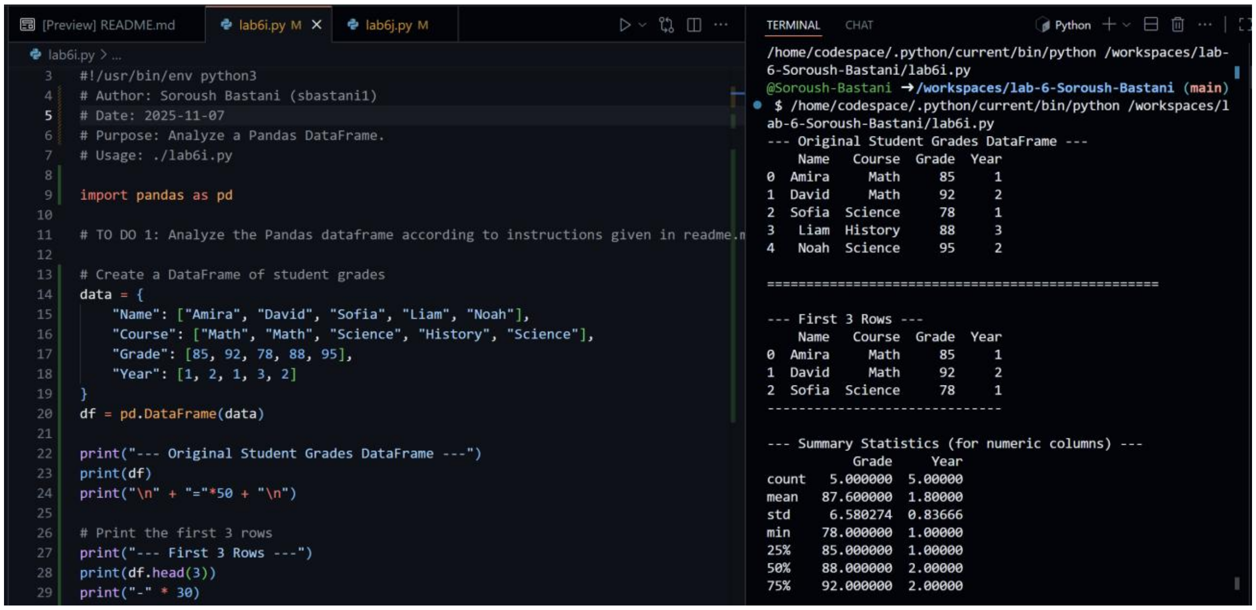
```
/home/codespace/.python/current/bin/python /workspaces/lab-
6-Soroush-Bastani/lab6h.py
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main)
 $ /home/codespace/.python/curr
ent/bin/python /workspaces/lab-6-Soroush-Bastani/lab6h.py
--- Library Books DataFrame ---
                               Title  ...         Genre
0                   The Great Gatsby  ...       Fiction
1            A Brief History of Time  ...       Science
2              To Kill a Mockingbird  ...       Fiction
3    The Art of Computer Programming  ...  Computer Sci
4  Sapiens: A Brief History of Humankind  ...       History

[5 rows x 5 columns]

==================================================

--- Exploring the DataFrame ---
The 'Title' column:
0                          The Great Gatsby
1                   A Brief History of Time
2                     To Kill a Mockingbird
3           The Art of Computer Programming
4     Sapiens: A Brief History of Humankind
Name: Title, dtype: object
------------------------------

The 'Author' column:
0    F. Scott Fitzgerald
1        Stephen Hawking
2             Harper Lee
```



```python
# Perform and print the requested operations
print("--- Exploring the DataFrame ---")

# Print the "Title" column
print("The 'Title' column:")
print(df["Title"])
print("-" * 30)

# Print the "Author" column
print("\nThe 'Author' column:")
print(df["Author"])
print("-" * 30)

# Print the first 3 rows using .head()
print("\nThe first 3 rows of the DataFrame:")
print(df.head(3))
print("-" * 30)

# Print the number of rows and columns using .shape
rows, cols = df.shape
print(f"\nThe DataFrame has {rows} rows and {cols} columns.")
print("-" * 30)

# Print the data type of each column using .dtypes
print("\nThe data type of each column:")
print(df.dtypes)
```

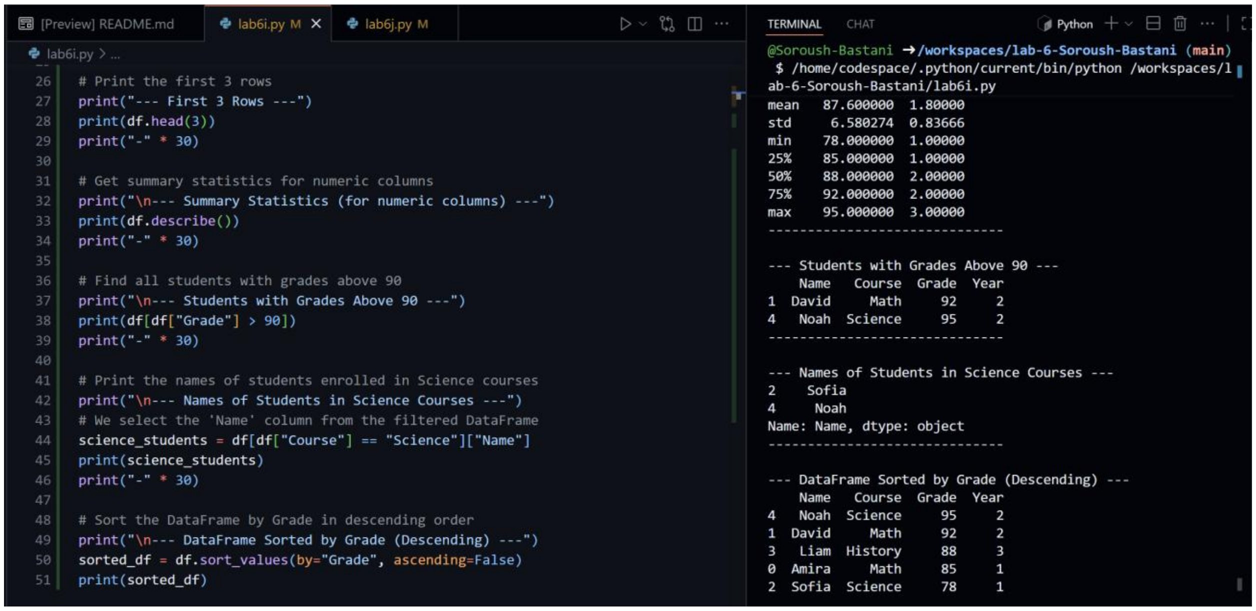
```
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main)
 $ /home/codespace/.python/curr
ent/bin/python /workspaces/lab-6-Soroush-Bastani/lab6h.py
3           The Art of Computer Programming
4     Sapiens: A Brief History of Humankind
Name: Title, dtype: object
------------------------------

The 'Author' column:
0    F. Scott Fitzgerald
1        Stephen Hawking
2             Harper Lee
3           Donald Knuth
4      Yuval Noah Harari
Name: Author, dtype: object
------------------------------

The first 3 rows of the DataFrame:
                     Title  ...    Genre
0         The Great Gatsby  ...  Fiction
1  A Brief History of Time  ...  Science
2    To Kill a Mockingbird  ...  Fiction

[3 rows x 5 columns]
------------------------------

The DataFrame has 5 rows and 5 columns.
------------------------------

The data type of each column:
Title      object
```

Lab6i



```python
#!/usr/bin/env python3
# Author: Soroush Bastani (sbastani1)
# Date: 2025-11-07
# Purpose: Analyze a Pandas DataFrame.
# Usage: ./lab6i.py

import pandas as pd

# TO DO 1: Analyze the Pandas dataframe according to instructions given in readme.m

# Create a DataFrame of student grades
data = {
    "Name": ["Amira", "David", "Sofia", "Liam", "Noah"],
    "Course": ["Math", "Math", "Science", "History", "Science"],
    "Grade": [85, 92, 78, 88, 95],
    "Year": [1, 2, 1, 3, 2]
}
df = pd.DataFrame(data)

print("--- Original Student Grades DataFrame ---")
print(df)
print("\n" + "="*50 + "\n")

# Print the first 3 rows
print("--- First 3 Rows ---")
print(df.head(3))
print("-" * 30)
```

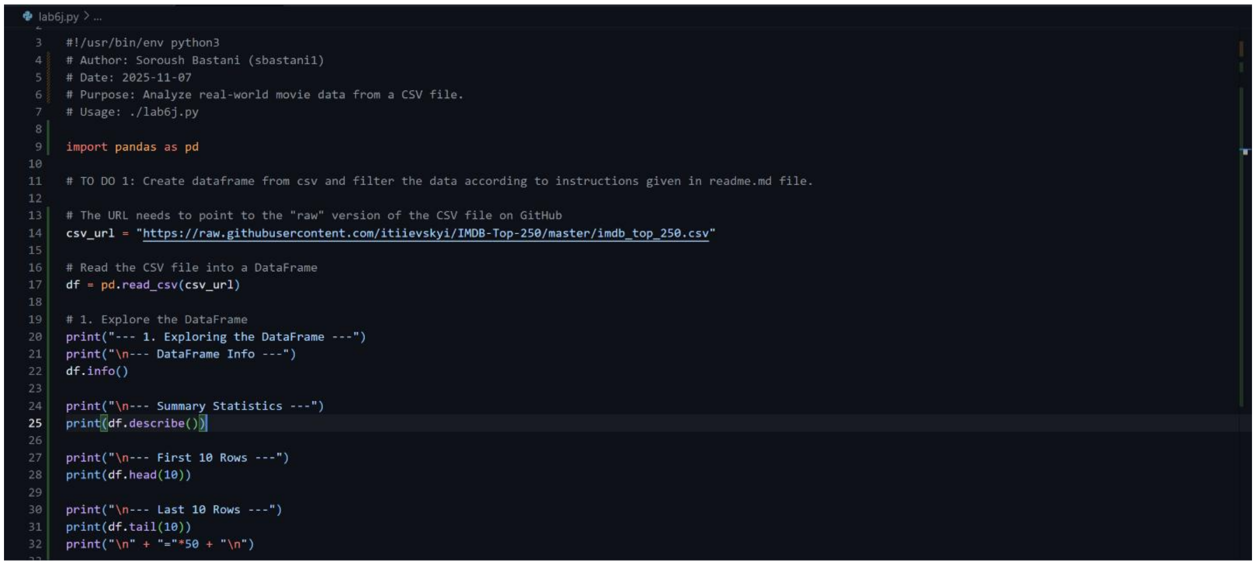
```
/home/codespace/.python/current/bin/python /workspaces/lab-
6-Soroush-Bastani/lab6i.py
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main)
$ /home/codespace/.python/current/bin/python /workspaces/l
ab-6-Soroush-Bastani/lab6i.py
--- Original Student Grades DataFrame ---
    Name   Course  Grade  Year
0  Amira     Math     85     1
1  David     Math     92     2
2  Sofia  Science     78     1
3   Liam  History     88     3
4   Noah  Science     95     2

==================================================

--- First 3 Rows ---
    Name   Course  Grade  Year
0  Amira     Math     85     1
1  David     Math     92     2
2  Sofia  Science     78     1
------------------------------

--- Summary Statistics (for numeric columns) ---
           Grade     Year
count   5.000000  5.00000
mean   87.600000  1.80000
std     6.580274  0.83666
min    78.000000  1.00000
25%    85.000000  1.00000
50%    88.000000  2.00000
75%    92.000000  2.00000
```



```python
# Print the first 3 rows
print("--- First 3 Rows ---")
print(df.head(3))
print("-" * 30)

# Get summary statistics for numeric columns
print("\n--- Summary Statistics (for numeric columns) ---")
print(df.describe())
print("-" * 30)

# Find all students with grades above 90
print("\n--- Students with Grades Above 90 ---")
print(df[df["Grade"] > 90])
print("-" * 30)

# Print the names of students enrolled in Science courses
print("\n--- Names of Students in Science Courses ---")
# We select the 'Name' column from the filtered DataFrame
science_students = df[df["Course"] == "Science"]["Name"]
print(science_students)
print("-" * 30)

# Sort the DataFrame by Grade in descending order
print("\n--- DataFrame Sorted by Grade (Descending) ---")
sorted_df = df.sort_values(by="Grade", ascending=False)
print(sorted_df)
```

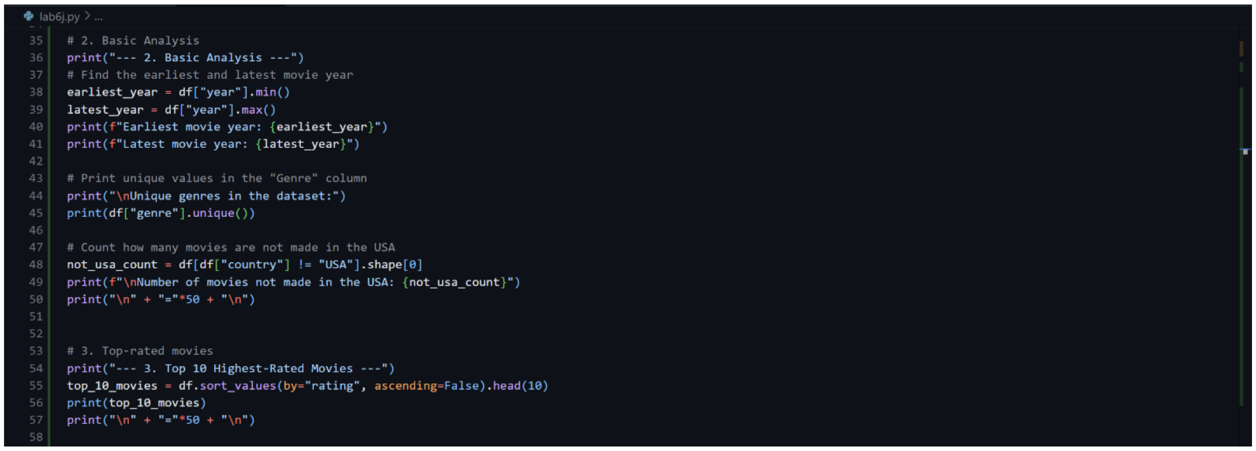
```
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main)
$ /home/codespace/.python/current/bin/python /workspaces/l
ab-6-Soroush-Bastani/lab6i.py
mean   87.600000  1.80000
std     6.580274  0.83666
min    78.000000  1.00000
25%    85.000000  1.00000
50%    88.000000  2.00000
75%    92.000000  2.00000
max    95.000000  3.00000
------------------------------

--- Students with Grades Above 90 ---
    Name   Course  Grade  Year
1  David     Math     92     2
4   Noah  Science     95     2
------------------------------

--- Names of Students in Science Courses ---
2    Sofia
4     Noah
Name: Name, dtype: object
------------------------------

--- DataFrame Sorted by Grade (Descending) ---
    Name   Course  Grade  Year
4   Noah  Science     95     2
1  David     Math     92     2
3   Liam  History     88     3
0  Amira     Math     85     1
2  Sofia  Science     78     1
```

Lab6j



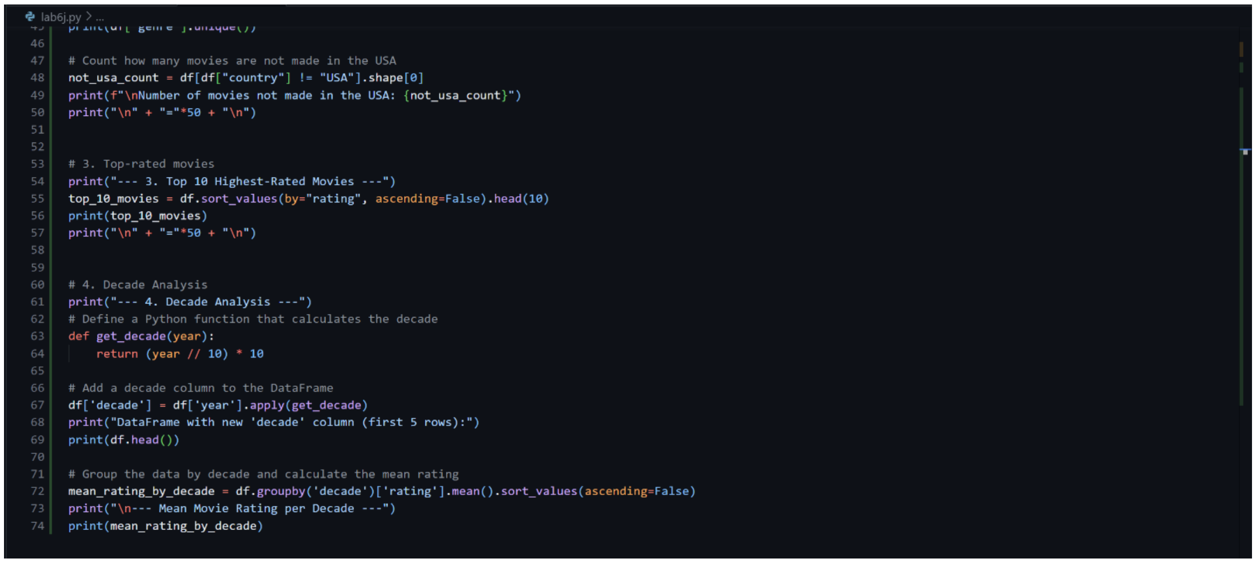
lab6j.py > ...

```python
#!/usr/bin/env python3
# Author: Soroush Bastani (sbastani1)
# Date: 2025-11-07
# Purpose: Analyze real-world movie data from a CSV file.
# Usage: ./lab6j.py

import pandas as pd

# TO DO 1: Create dataframe from csv and filter the data according to instructions given in readme.md file.

# The URL needs to point to the "raw" version of the CSV file on GitHub
csv_url = "https://raw.githubusercontent.com/itiievskyi/IMDB-Top-250/master/imdb_top_250.csv"

# Read the CSV file into a DataFrame
df = pd.read_csv(csv_url)

# 1. Explore the DataFrame
print("--- 1. Exploring the DataFrame ---")
print("\n--- DataFrame Info ---")
df.info()

print("\n--- Summary Statistics ---")
print(df.describe())

print("\n--- First 10 Rows ---")
print(df.head(10))

print("\n--- Last 10 Rows ---")
print(df.tail(10))
print("\n" + "="*50 + "\n")
```



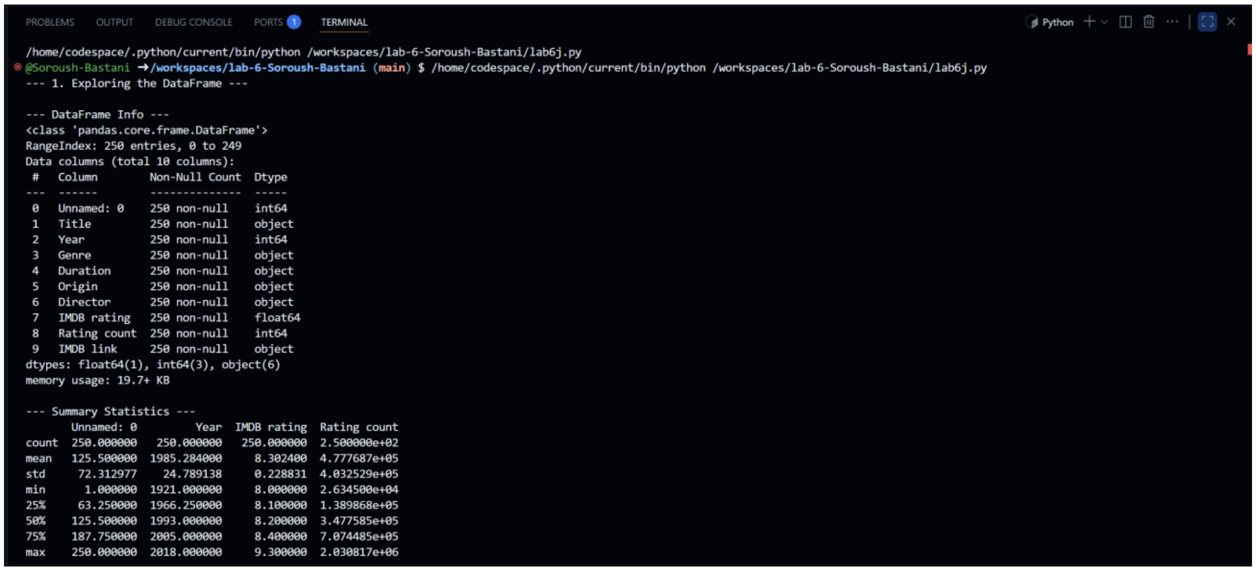
lab6j.py > ...

```python
# 2. Basic Analysis
print("--- 2. Basic Analysis ---")
# Find the earliest and latest movie year
earliest_year = df["year"].min()
latest_year = df["year"].max()
print(f"Earliest movie year: {earliest_year}")
print(f"Latest movie year: {latest_year}")

# Print unique values in the "Genre" column
print("\nUnique genres in the dataset:")
print(df["genre"].unique())

# Count how many movies are not made in the USA
not_usa_count = df[df["country"] != "USA"].shape[0]
print(f"\nNumber of movies not made in the USA: {not_usa_count}")
print("\n" + "="*50 + "\n")


# 3. Top-rated movies
print("--- 3. Top 10 Highest-Rated Movies ---")
top_10_movies = df.sort_values(by="rating", ascending=False).head(10)
print(top_10_movies)
print("\n" + "="*50 + "\n")

```

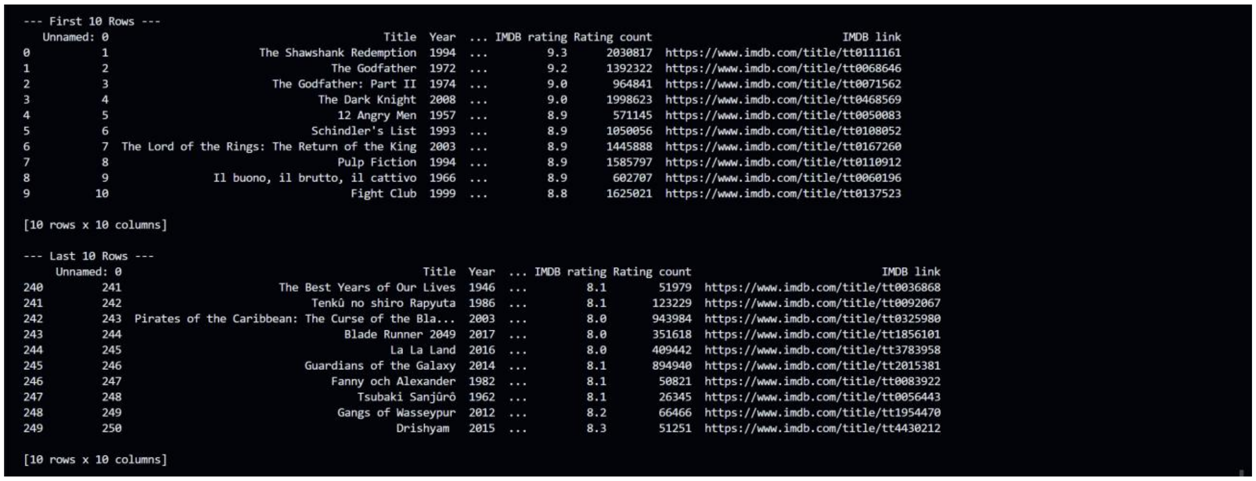


lab6j.py > ...

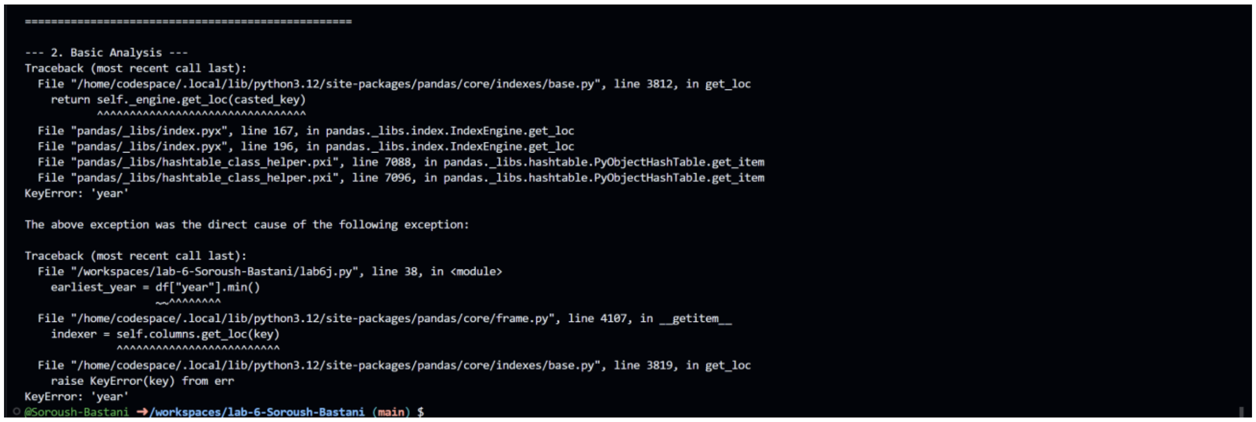
```python

# Count how many movies are not made in the USA
not_usa_count = df[df["country"] != "USA"].shape[0]
print(f"\nNumber of movies not made in the USA: {not_usa_count}")
print("\n" + "="*50 + "\n")


# 3. Top-rated movies
print("--- 3. Top 10 Highest-Rated Movies ---")
top_10_movies = df.sort_values(by="rating", ascending=False).head(10)
print(top_10_movies)
print("\n" + "="*50 + "\n")


# 4. Decade Analysis
print("--- 4. Decade Analysis ---")
# Define a Python function that calculates the decade
def get_decade(year):
    return (year // 10) * 10

# Add a decade column to the DataFrame
df['decade'] = df['year'].apply(get_decade)
print("DataFrame with new 'decade' column (first 5 rows):")
print(df.head())

# Group the data by decade and calculate the mean rating
mean_rating_by_decade = df.groupby('decade')['rating'].mean().sort_values(ascending=False)
print("\n--- Mean Movie Rating per Decade ---")
print(mean_rating_by_decade)
```



```
/home/codespace/.python/current/bin/python /workspaces/lab-6-Soroush-Bastani/lab6j.py
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main) $ /home/codespace/.python/current/bin/python /workspaces/lab-6-Soroush-Bastani/lab6j.py
--- 1. Exploring the DataFrame ---

--- DataFrame Info ---
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250 entries, 0 to 249
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Unnamed: 0    250 non-null    int64
 1   Title         250 non-null    object
 2   Year          250 non-null    int64
 3   Genre         250 non-null    object
 4   Duration      250 non-null    object
 5   Origin        250 non-null    object
 6   Director      250 non-null    object
 7   IMDB rating   250 non-null    float64
 8   Rating count  250 non-null    int64
 9   IMDB link     250 non-null    object
dtypes: float64(1), int64(3), object(6)
memory usage: 19.7+ KB

--- Summary Statistics ---
       Unnamed: 0         Year  IMDB rating  Rating count
count  250.000000   250.000000   250.000000  2.500000e+02
mean   125.500000  1985.284000     8.302400  4.777687e+05
std     72.312977    24.789138     0.228831  4.032529e+05
min      1.000000  1921.000000     8.000000  2.634500e+04
25%     63.250000  1966.250000     8.100000  1.389868e+05
50%    125.500000  1993.000000     8.200000  3.477585e+05
75%    187.750000  2005.000000     8.400000  7.074485e+05
max    250.000000  2018.000000     9.300000  2.030817e+06
```



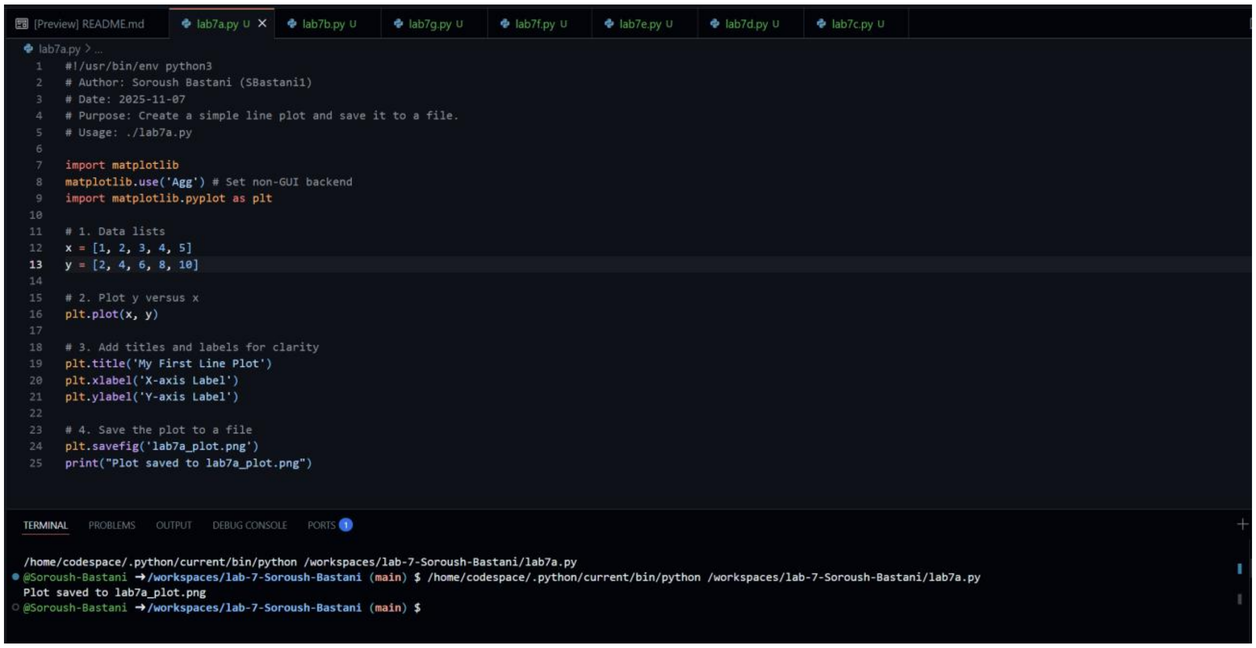
```
--- First 10 Rows ---
   Unnamed: 0                                          Title  Year  ... IMDB rating Rating count                              IMDB link
0           1                       The Shawshank Redemption  1994  ...         9.3      2030817  https://www.imdb.com/title/tt0111161
1           2                                  The Godfather  1972  ...         9.2      1392322  https://www.imdb.com/title/tt0068646
2           3                         The Godfather: Part II  1974  ...         9.0       964841  https://www.imdb.com/title/tt0071562
3           4                                The Dark Knight  2008  ...         9.0      1998623  https://www.imdb.com/title/tt0468569
4           5                                   12 Angry Men  1957  ...         8.9       571145  https://www.imdb.com/title/tt0050083
5           6                               Schindler's List  1993  ...         8.9      1050056  https://www.imdb.com/title/tt0108052
6           7  The Lord of the Rings: The Return of the King  2003  ...         8.9      1445888  https://www.imdb.com/title/tt0167260
7           8                                   Pulp Fiction  1994  ...         8.9      1585797  https://www.imdb.com/title/tt0110912
8           9                Il buono, il brutto, il cattivo  1966  ...         8.9       602707  https://www.imdb.com/title/tt0060196
9          10                                     Fight Club  1999  ...         8.8      1625021  https://www.imdb.com/title/tt0137523

[10 rows x 10 columns]

--- Last 10 Rows ---
     Unnamed: 0                                          Title  Year  ... IMDB rating Rating count                              IMDB link
240         241                    The Best Years of Our Lives  1946  ...         8.1        51979  https://www.imdb.com/title/tt0036868
241         242                        Tenkû no shiro Rapyuta  1986  ...         8.1       123229  https://www.imdb.com/title/tt0092067
242         243  Pirates of the Caribbean: The Curse of the Bla...  2003  ...         8.0       943984  https://www.imdb.com/title/tt0325980
243         244                              Blade Runner 2049  2017  ...         8.0       351618  https://www.imdb.com/title/tt1856101
244         245                                     La La Land  2016  ...         8.0       409442  https://www.imdb.com/title/tt3783958
245         246                        Guardians of the Galaxy  2014  ...         8.1       894940  https://www.imdb.com/title/tt2015381
246         247                          Fanny och Alexander  1982  ...         8.1        50821  https://www.imdb.com/title/tt0083922
247         248                              Tsubaki Sanjûrô  1962  ...         8.1        26345  https://www.imdb.com/title/tt0056443
248         249                             Gangs of Wasseypur  2012  ...         8.2        66466  https://www.imdb.com/title/tt1954470
249         250                                      Drishyam  2015  ...         8.3        51251  https://www.imdb.com/title/tt4430212

[10 rows x 10 columns]
```



```
==================================================

--- 2. Basic Analysis ---
Traceback (most recent call last):
  File "/home/codespace/.local/lib/python3.12/site-packages/pandas/core/indexes/base.py", line 3812, in get_loc
    return self._engine.get_loc(casted_key)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "pandas/_libs/index.pyx", line 167, in pandas._libs.index.IndexEngine.get_loc
  File "pandas/_libs/index.pyx", line 196, in pandas._libs.index.IndexEngine.get_loc
  File "pandas/_libs/hashtable_class_helper.pxi", line 7088, in pandas._libs.hashtable.PyObjectHashTable.get_item
  File "pandas/_libs/hashtable_class_helper.pxi", line 7096, in pandas._libs.hashtable.PyObjectHashTable.get_item
KeyError: 'year'

The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "/workspaces/lab-6-Soroush-Bastani/lab6j.py", line 38, in <module>
    earliest_year = df["year"].min()
                    ~~^^^^^^^^
  File "/home/codespace/.local/lib/python3.12/site-packages/pandas/core/frame.py", line 4107, in __getitem__
    indexer = self.columns.get_loc(key)
              ^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/codespace/.local/lib/python3.12/site-packages/pandas/core/indexes/base.py", line 3819, in get_loc
    raise KeyError(key) from err
KeyError: 'year'
@Soroush-Bastani ➜ /workspaces/lab-6-Soroush-Bastani (main) $
```

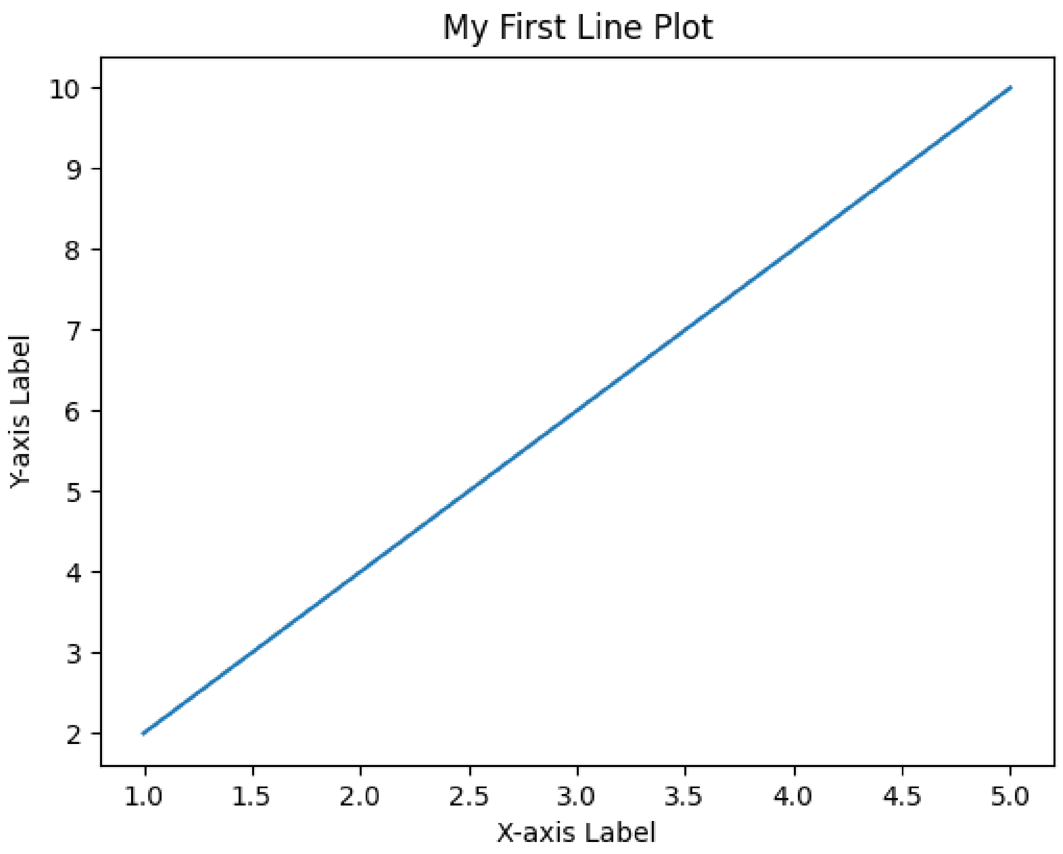
Lab7

Lab7a

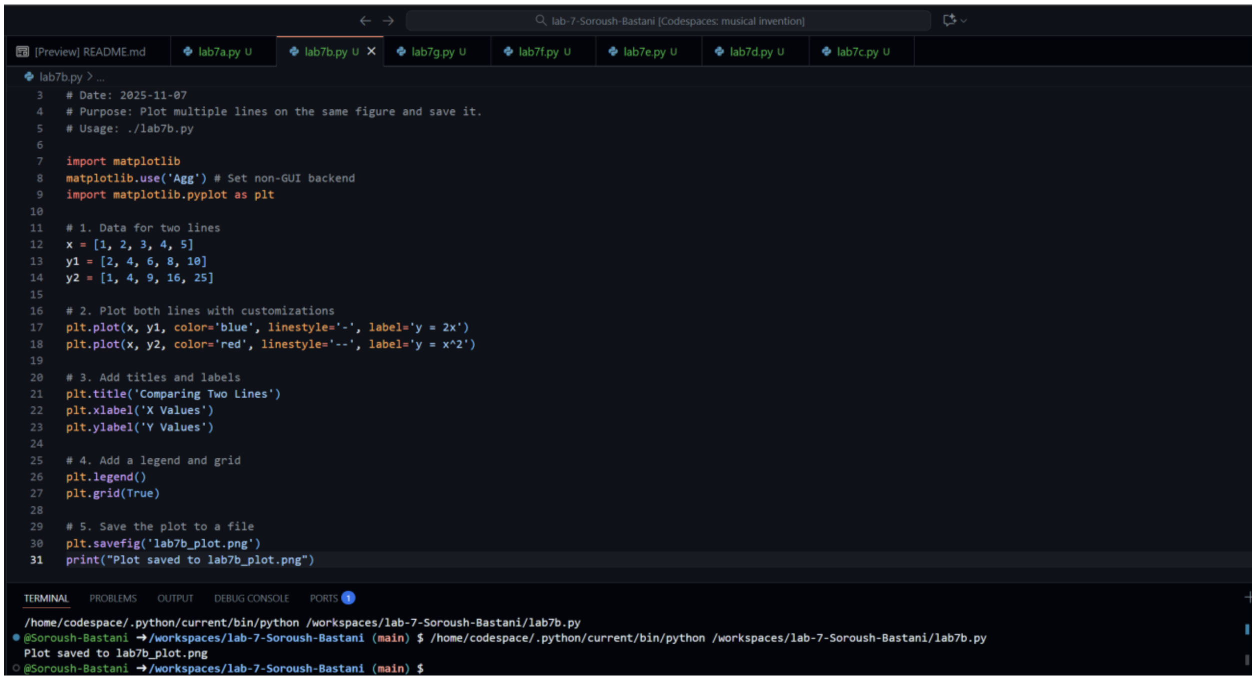


```python
#!/usr/bin/env python3
# Author: Soroush Bastani (SBastani1)
# Date: 2025-11-07
# Purpose: Create a simple line plot and save it to a file.
# Usage: ./lab7a.py

import matplotlib
matplotlib.use('Agg') # Set non-GUI backend
import matplotlib.pyplot as plt

# 1. Data lists
x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]

# 2. Plot y versus x
plt.plot(x, y)

# 3. Add titles and labels for clarity
plt.title('My First Line Plot')
plt.xlabel('X-axis Label')
plt.ylabel('Y-axis Label')

# 4. Save the plot to a file
plt.savefig('lab7a_plot.png')
print("Plot saved to lab7a_plot.png")
```

```
/home/codespace/.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/lab7a.py
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main) $ /home/codespace/.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/lab7a.py
Plot saved to lab7a_plot.png
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main) $
```

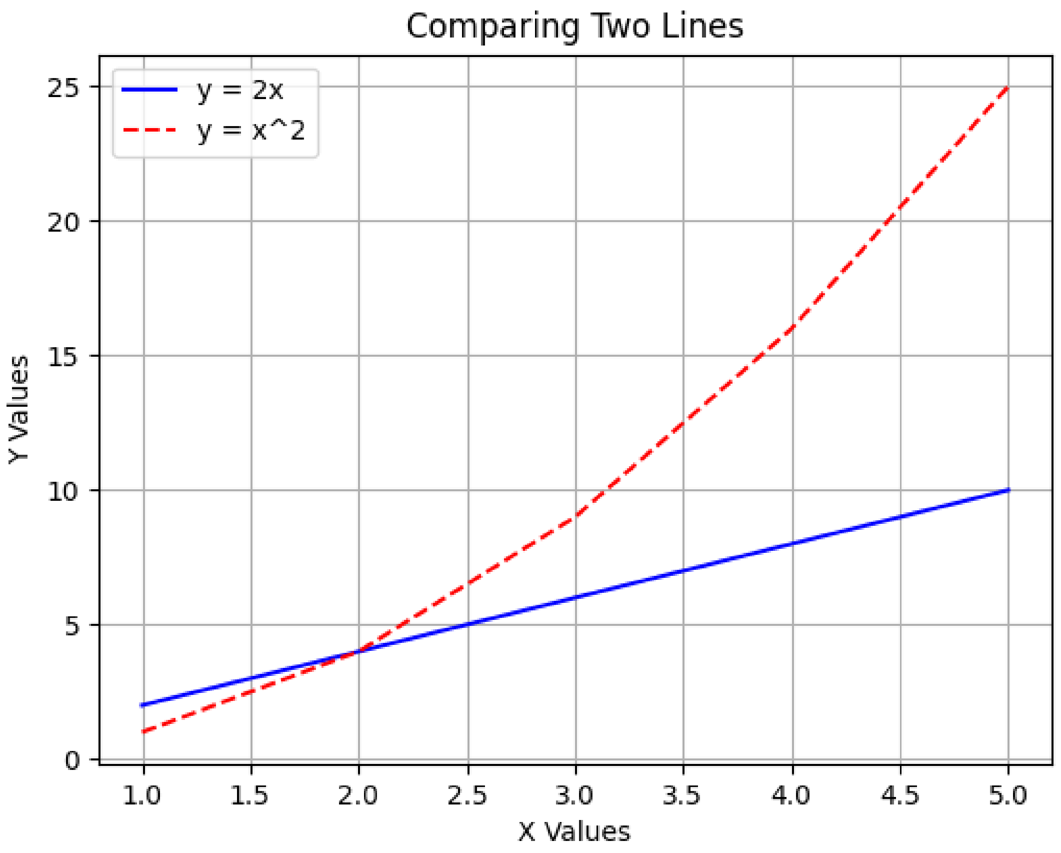


Lab7b

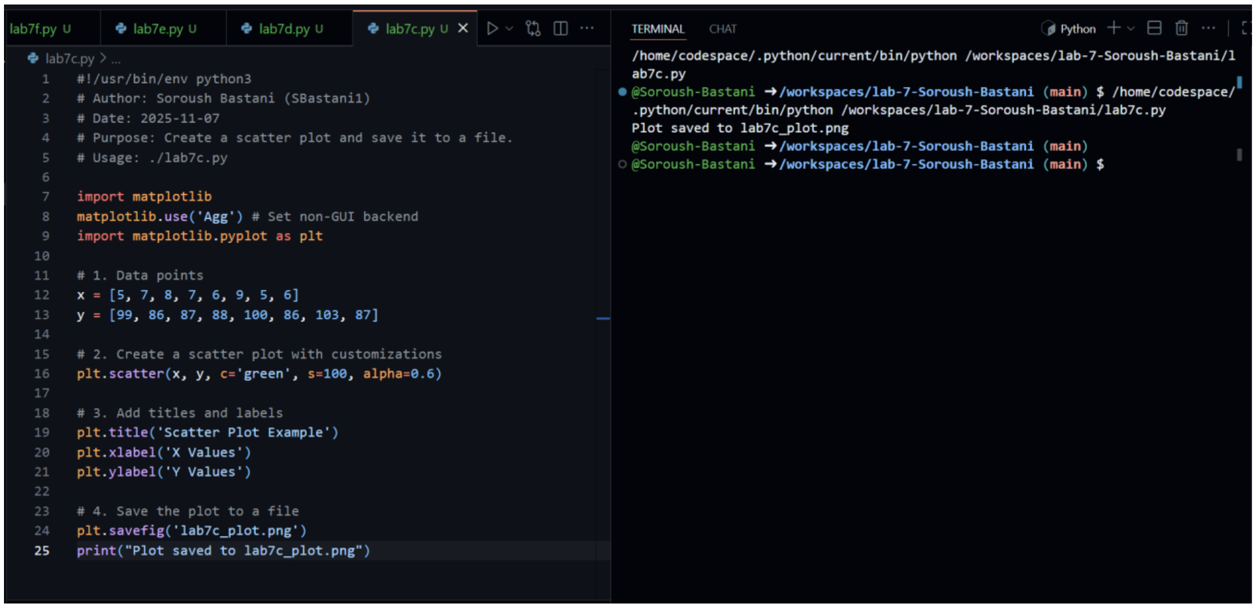


```python
# Date: 2025-11-07
# Purpose: Plot multiple lines on the same figure and save it.
# Usage: ./lab7b.py

import matplotlib
matplotlib.use('Agg') # Set non-GUI backend
import matplotlib.pyplot as plt

# 1. Data for two lines
x = [1, 2, 3, 4, 5]
y1 = [2, 4, 6, 8, 10]
y2 = [1, 4, 9, 16, 25]

# 2. Plot both lines with customizations
plt.plot(x, y1, color='blue', linestyle='-', label='y = 2x')
plt.plot(x, y2, color='red', linestyle='--', label='y = x^2')

# 3. Add titles and labels
plt.title('Comparing Two Lines')
plt.xlabel('X Values')
plt.ylabel('Y Values')

# 4. Add a legend and grid
plt.legend()
plt.grid(True)

# 5. Save the plot to a file
plt.savefig('lab7b_plot.png')
print("Plot saved to lab7b_plot.png")
```

```
/home/codespace/.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/lab7b.py
● @Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main) $ /home/codespace/.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/lab7b.py
Plot saved to lab7b_plot.png
○ @Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main) $
```

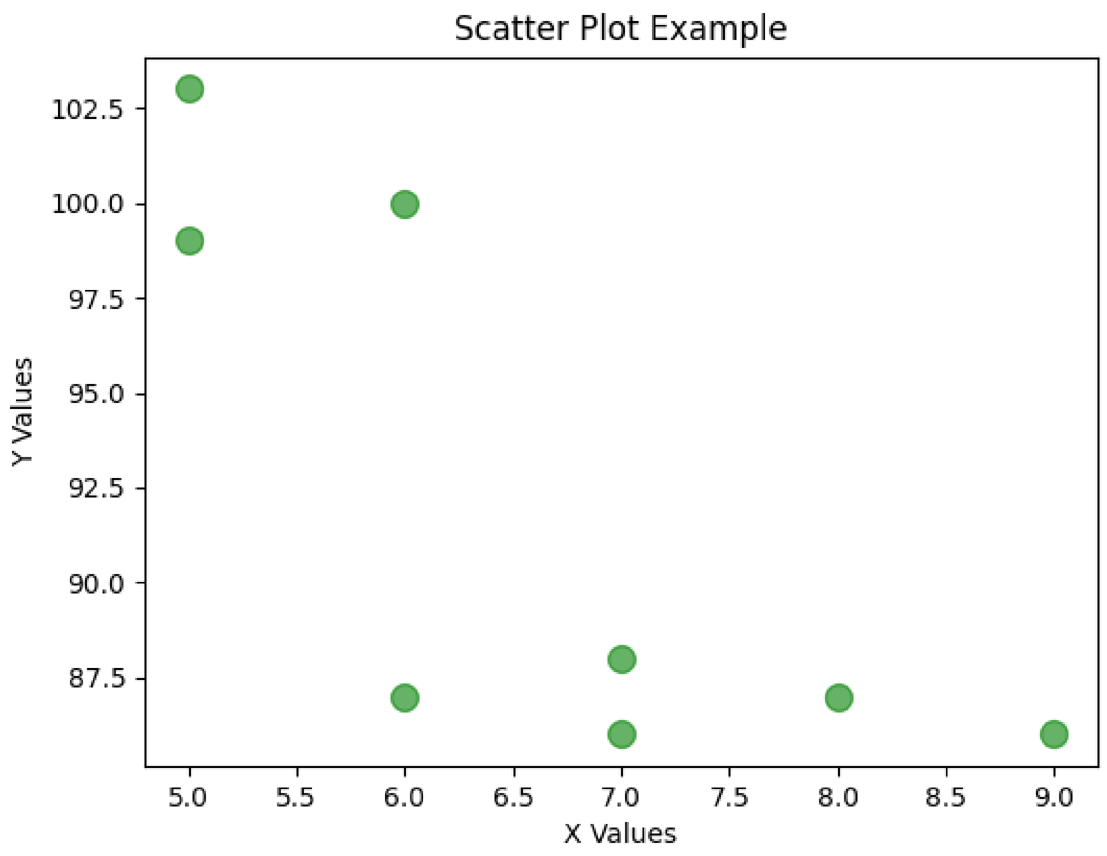


Lab7c

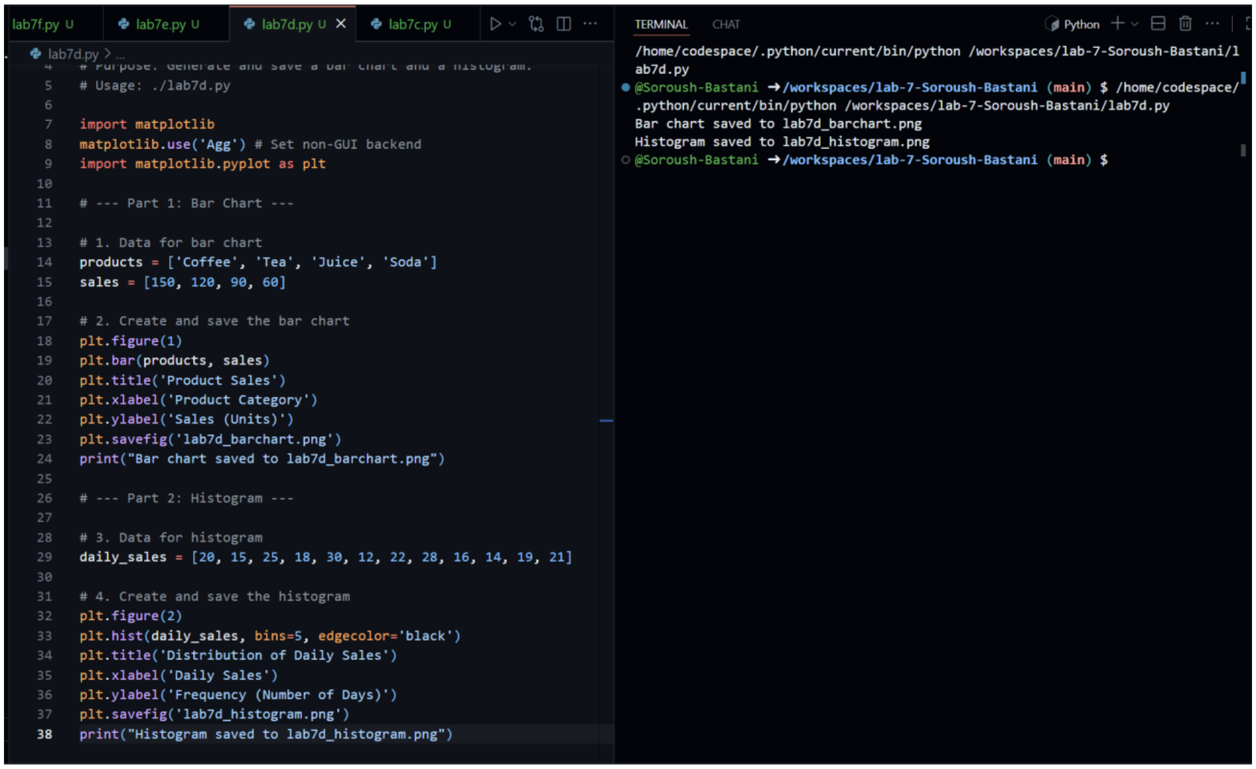


```python
#!/usr/bin/env python3
# Author: Soroush Bastani (SBastani1)
# Date: 2025-11-07
# Purpose: Create a scatter plot and save it to a file.
# Usage: ./lab7c.py

import matplotlib
matplotlib.use('Agg') # Set non-GUI backend
import matplotlib.pyplot as plt

# 1. Data points
x = [5, 7, 8, 7, 6, 9, 5, 6]
y = [99, 86, 87, 88, 100, 86, 103, 87]

# 2. Create a scatter plot with customizations
plt.scatter(x, y, c='green', s=100, alpha=0.6)

# 3. Add titles and labels
plt.title('Scatter Plot Example')
plt.xlabel('X Values')
plt.ylabel('Y Values')

# 4. Save the plot to a file
plt.savefig('lab7c_plot.png')
print("Plot saved to lab7c_plot.png")
```

```
/home/codespace/.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/l
ab7c.py
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main) $ /home/codespace/
.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/lab7c.py
Plot saved to lab7c_plot.png
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main)
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main) $
```

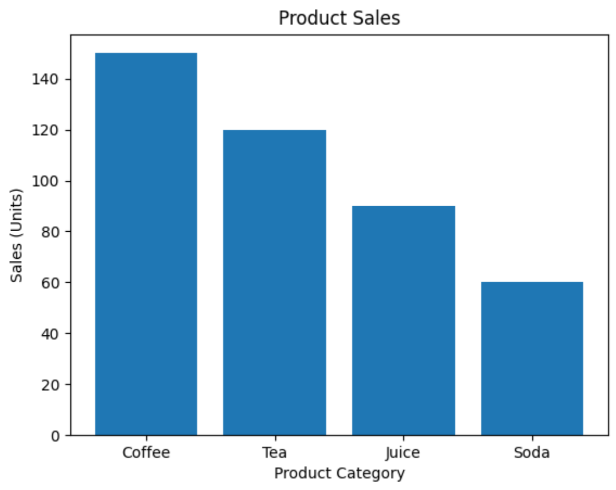


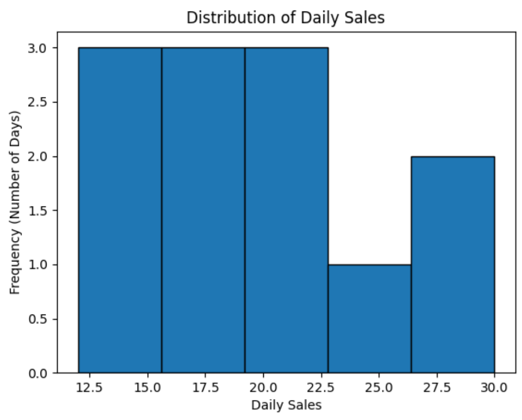
Lab7d



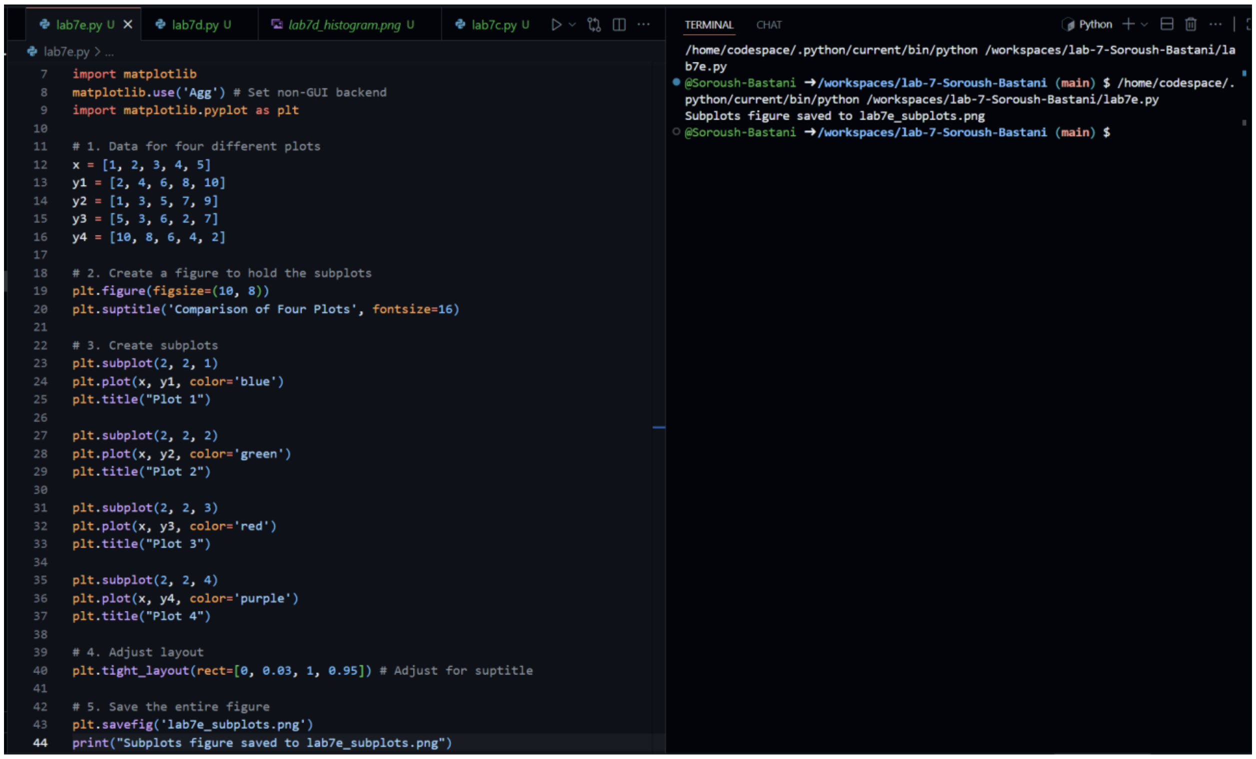
```python
# Purpose: Generate and save a bar chart and a histogram.
# Usage: ./lab7d.py

import matplotlib
matplotlib.use('Agg') # Set non-GUI backend
import matplotlib.pyplot as plt

# --- Part 1: Bar Chart ---

# 1. Data for bar chart
products = ['Coffee', 'Tea', 'Juice', 'Soda']
sales = [150, 120, 90, 60]

# 2. Create and save the bar chart
plt.figure(1)
plt.bar(products, sales)
plt.title('Product Sales')
plt.xlabel('Product Category')
plt.ylabel('Sales (Units)')
plt.savefig('lab7d_barchart.png')
print("Bar chart saved to lab7d_barchart.png")

# --- Part 2: Histogram ---

# 3. Data for histogram
daily_sales = [20, 15, 25, 18, 30, 12, 22, 28, 16, 14, 19, 21]

# 4. Create and save the histogram
plt.figure(2)
plt.hist(daily_sales, bins=5, edgecolor='black')
plt.title('Distribution of Daily Sales')
plt.xlabel('Daily Sales')
plt.ylabel('Frequency (Number of Days)')
plt.savefig('lab7d_histogram.png')
print("Histogram saved to lab7d_histogram.png")
```

```
/home/codespace/.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/l
ab7d.py
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main) $ /home/codespace/
.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/lab7d.py
Bar chart saved to lab7d_barchart.png
Histogram saved to lab7d_histogram.png
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main) $
```



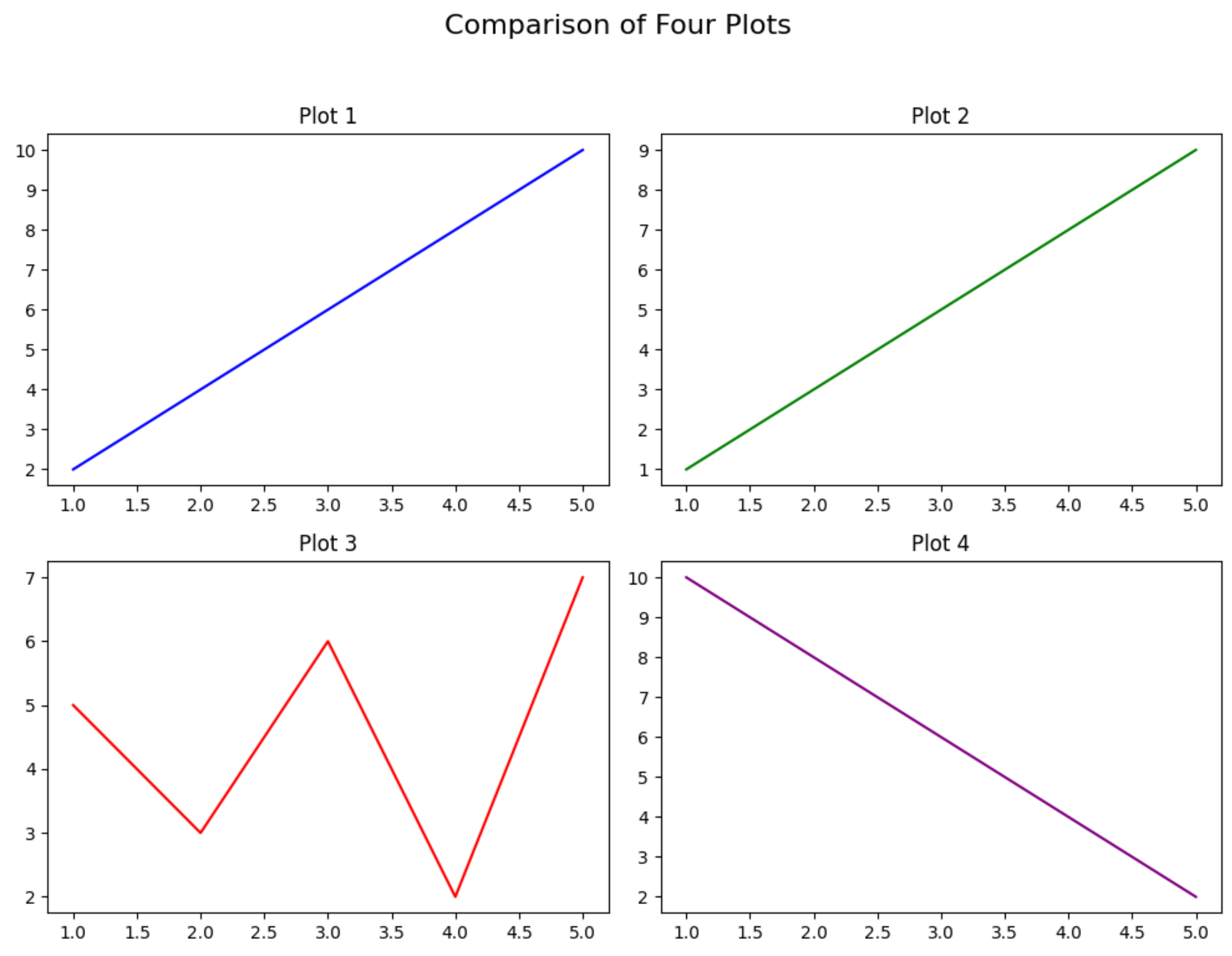


Lab7e

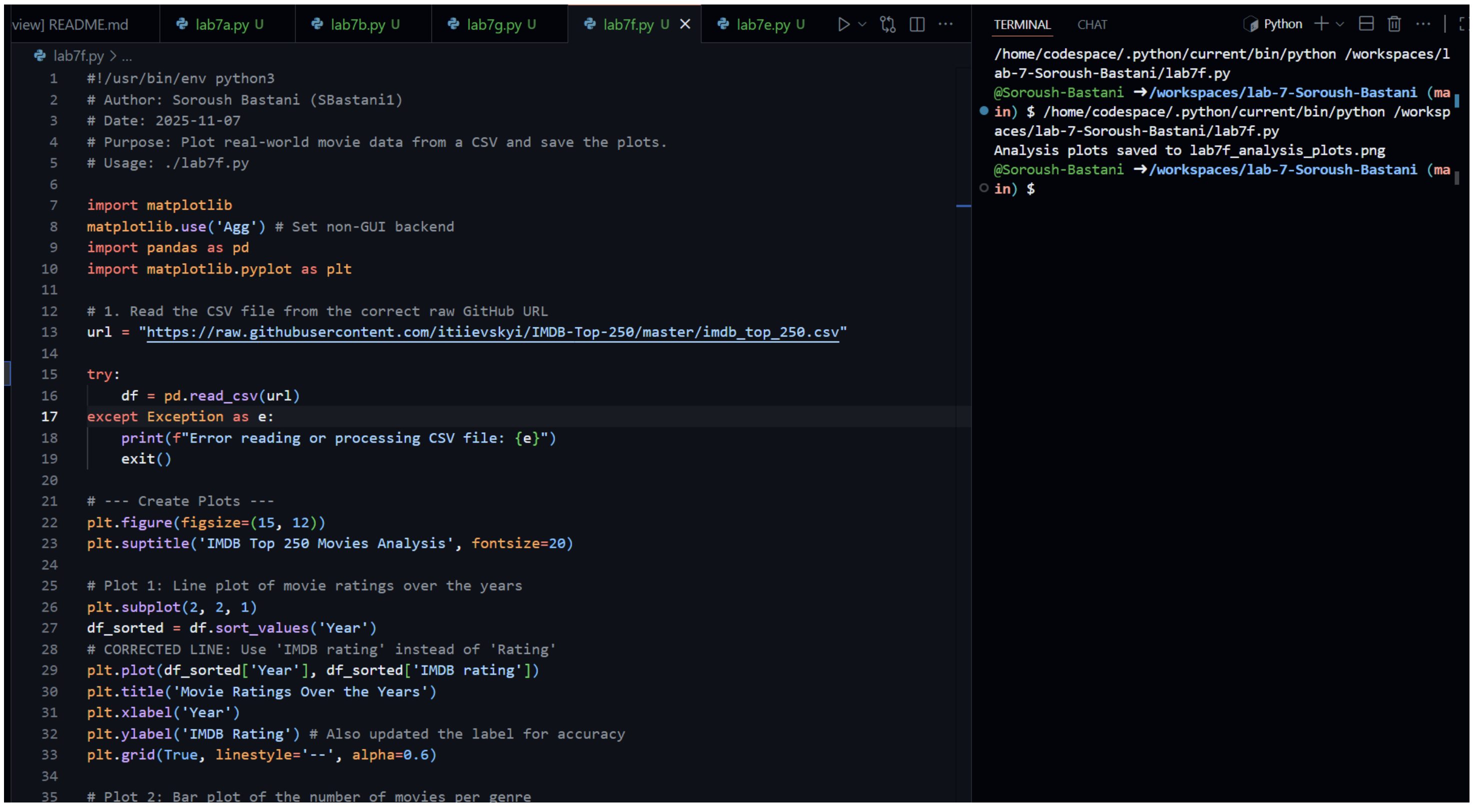


```python
import matplotlib
matplotlib.use('Agg') # Set non-GUI backend
import matplotlib.pyplot as plt

# 1. Data for four different plots
x = [1, 2, 3, 4, 5]
y1 = [2, 4, 6, 8, 10]
y2 = [1, 3, 5, 7, 9]
y3 = [5, 3, 6, 2, 7]
y4 = [10, 8, 6, 4, 2]

# 2. Create a figure to hold the subplots
plt.figure(figsize=(10, 8))
plt.suptitle('Comparison of Four Plots', fontsize=16)

# 3. Create subplots
plt.subplot(2, 2, 1)
plt.plot(x, y1, color='blue')
plt.title("Plot 1")

plt.subplot(2, 2, 2)
plt.plot(x, y2, color='green')
plt.title("Plot 2")

plt.subplot(2, 2, 3)
plt.plot(x, y3, color='red')
plt.title("Plot 3")

plt.subplot(2, 2, 4)
plt.plot(x, y4, color='purple')
plt.title("Plot 4")

# 4. Adjust layout
plt.tight_layout(rect=[0, 0.03, 1, 0.95]) # Adjust for suptitle

# 5. Save the entire figure
plt.savefig('lab7e_subplots.png')
print("Subplots figure saved to lab7e_subplots.png")
```

```
/home/codespace/.python/current/bin/python /workspaces/lab-7-Soroush-Bastani/la
b7e.py
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main) $ /home/codespace/.
python/current/bin/python /workspaces/lab-7-Soroush-Bastani/lab7e.py
Subplots figure saved to lab7e_subplots.png
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main) $
```

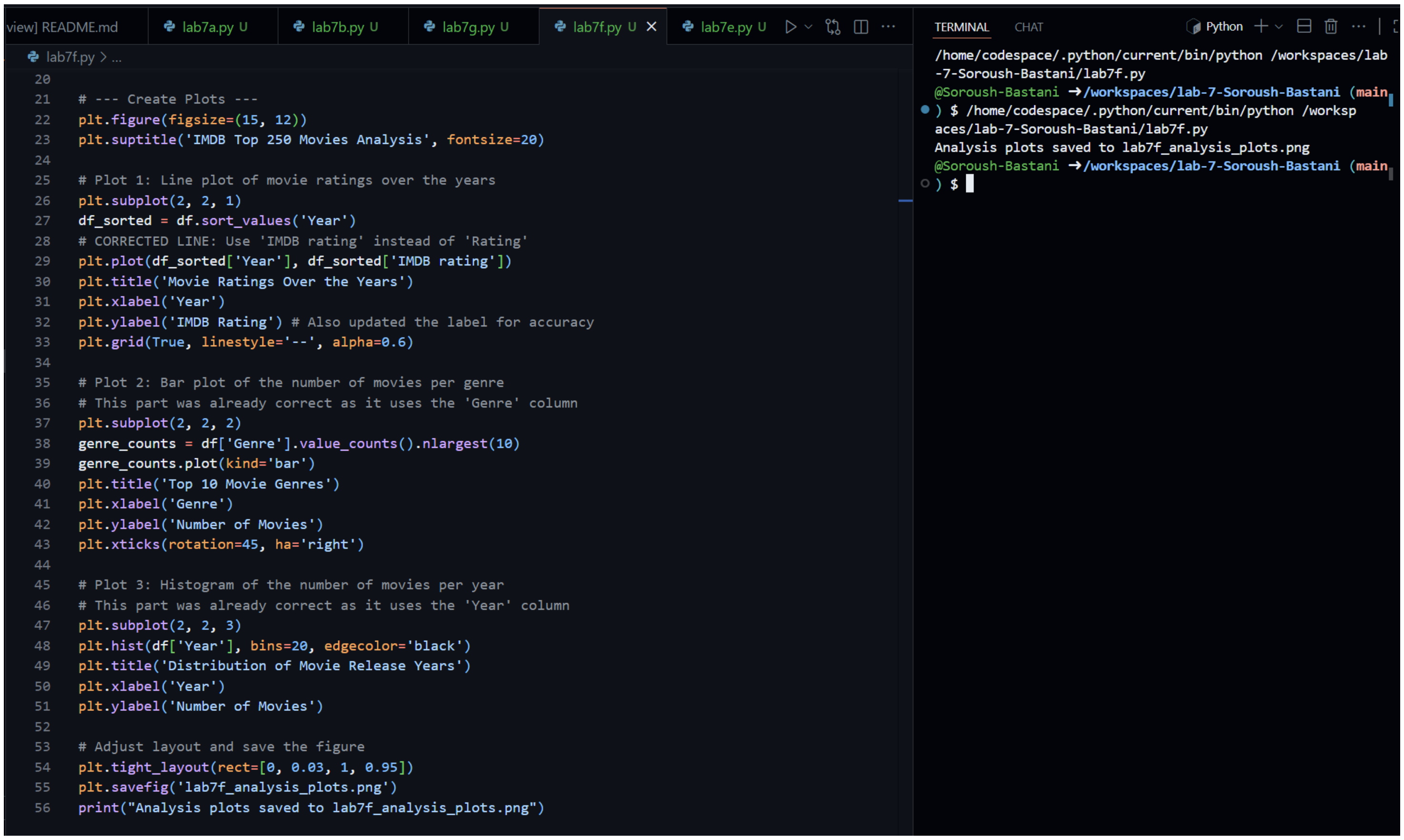


Lab7f



```python
#!/usr/bin/env python3
# Author: Soroush Bastani (SBastani1)
# Date: 2025-11-07
# Purpose: Plot real-world movie data from a CSV and save the plots.
# Usage: ./lab7f.py

import matplotlib
matplotlib.use('Agg') # Set non-GUI backend
import pandas as pd
import matplotlib.pyplot as plt

# 1. Read the CSV file from the correct raw GitHub URL
url = "https://raw.githubusercontent.com/itiievskyi/IMDB-Top-250/master/imdb_top_250.csv"

try:
    df = pd.read_csv(url)
except Exception as e:
    print(f"Error reading or processing CSV file: {e}")
    exit()

# --- Create Plots ---
plt.figure(figsize=(15, 12))
plt.suptitle('IMDB Top 250 Movies Analysis', fontsize=20)

# Plot 1: Line plot of movie ratings over the years
plt.subplot(2, 2, 1)
df_sorted = df.sort_values('Year')
# CORRECTED LINE: Use 'IMDB rating' instead of 'Rating'
plt.plot(df_sorted['Year'], df_sorted['IMDB rating'])
plt.title('Movie Ratings Over the Years')
plt.xlabel('Year')
plt.ylabel('IMDB Rating') # Also updated the label for accuracy
plt.grid(True, linestyle='--', alpha=0.6)

# Plot 2: Bar plot of the number of movies per genre
```

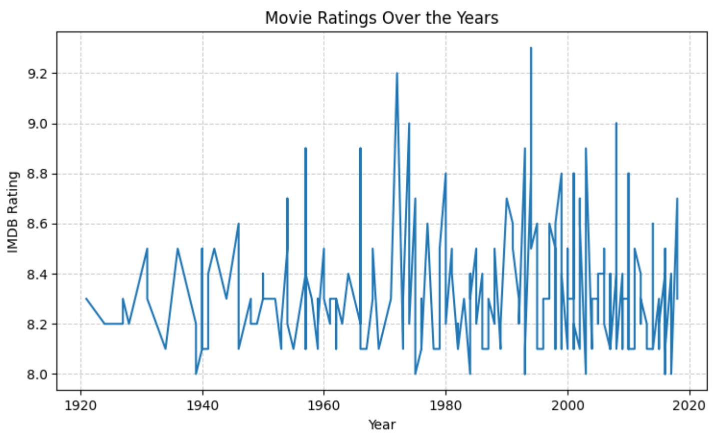
```
/home/codespace/.python/current/bin/python /workspaces/l
ab-7-Soroush-Bastani/lab7f.py
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (ma
in) $ /home/codespace/.python/current/bin/python /worksp
aces/lab-7-Soroush-Bastani/lab7f.py
Analysis plots saved to lab7f_analysis_plots.png
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (ma
in) $
```

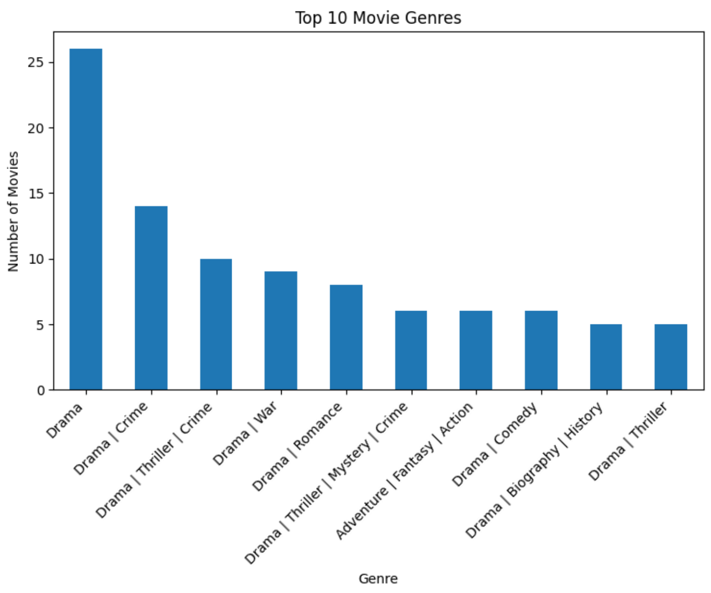


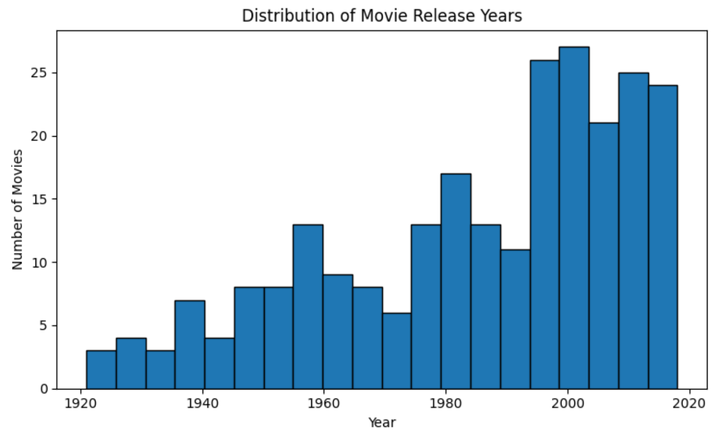
```python

# --- Create Plots ---
plt.figure(figsize=(15, 12))
plt.suptitle('IMDB Top 250 Movies Analysis', fontsize=20)

# Plot 1: Line plot of movie ratings over the years
plt.subplot(2, 2, 1)
df_sorted = df.sort_values('Year')
# CORRECTED LINE: Use 'IMDB rating' instead of 'Rating'
plt.plot(df_sorted['Year'], df_sorted['IMDB rating'])
plt.title('Movie Ratings Over the Years')
plt.xlabel('Year')
plt.ylabel('IMDB Rating') # Also updated the label for accuracy
plt.grid(True, linestyle='--', alpha=0.6)

# Plot 2: Bar plot of the number of movies per genre
# This part was already correct as it uses the 'Genre' column
plt.subplot(2, 2, 2)
genre_counts = df['Genre'].value_counts().nlargest(10)
genre_counts.plot(kind='bar')
plt.title('Top 10 Movie Genres')
plt.xlabel('Genre')
plt.ylabel('Number of Movies')
plt.xticks(rotation=45, ha='right')

# Plot 3: Histogram of the number of movies per year
# This part was already correct as it uses the 'Year' column
plt.subplot(2, 2, 3)
plt.hist(df['Year'], bins=20, edgecolor='black')
plt.title('Distribution of Movie Release Years')
plt.xlabel('Year')
plt.ylabel('Number of Movies')

# Adjust layout and save the figure
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.savefig('lab7f_analysis_plots.png')
print("Analysis plots saved to lab7f_analysis_plots.png")
```

```
/home/codespace/.python/current/bin/python /workspaces/lab
-7-Soroush-Bastani/lab7f.py
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main
) $ /home/codespace/.python/current/bin/python /worksp
aces/lab-7-Soroush-Bastani/lab7f.py
Analysis plots saved to lab7f_analysis_plots.png
@Soroush-Bastani ➜ /workspaces/lab-7-Soroush-Bastani (main
) $ █
```

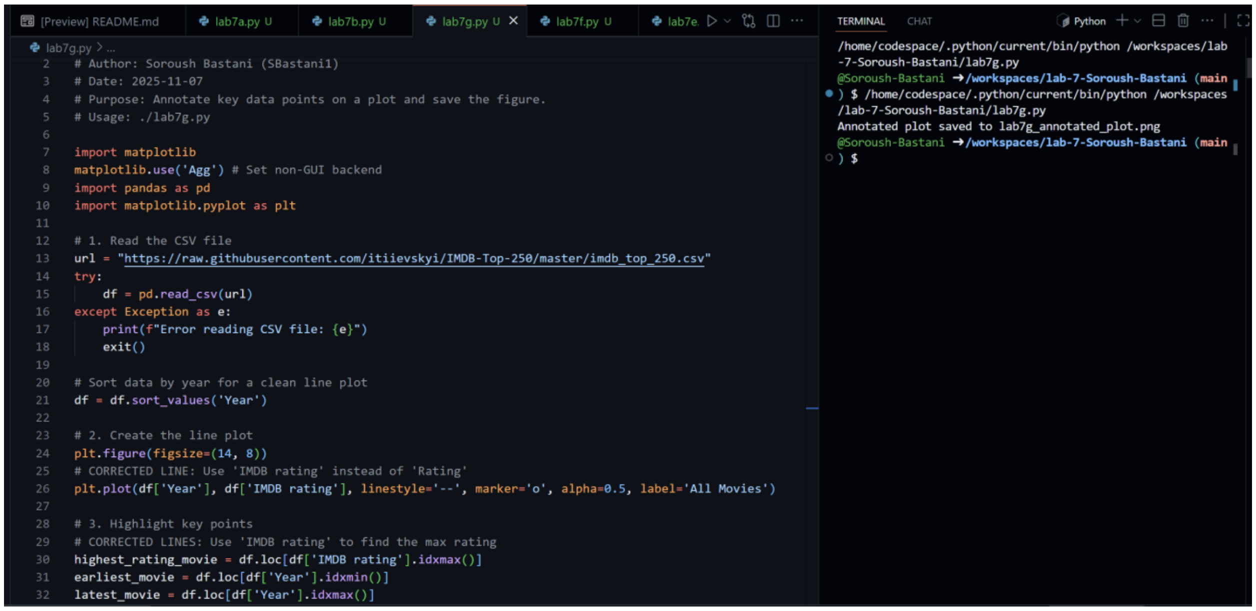
IMDB Top 250 Movies Analysis





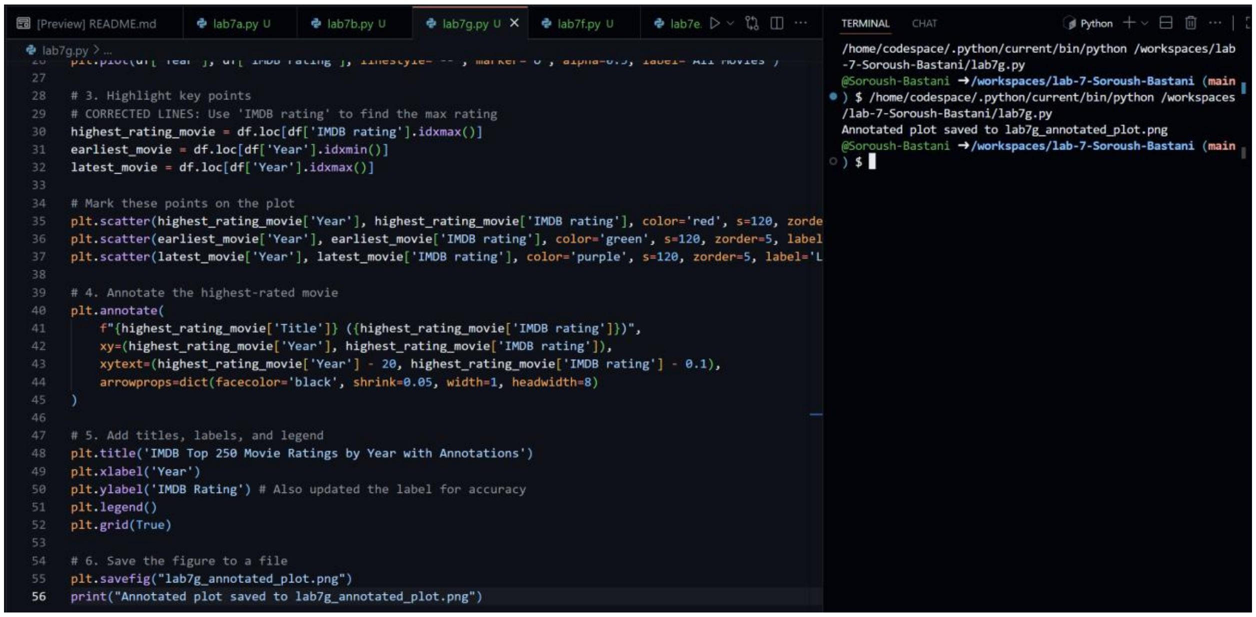


Lab7g



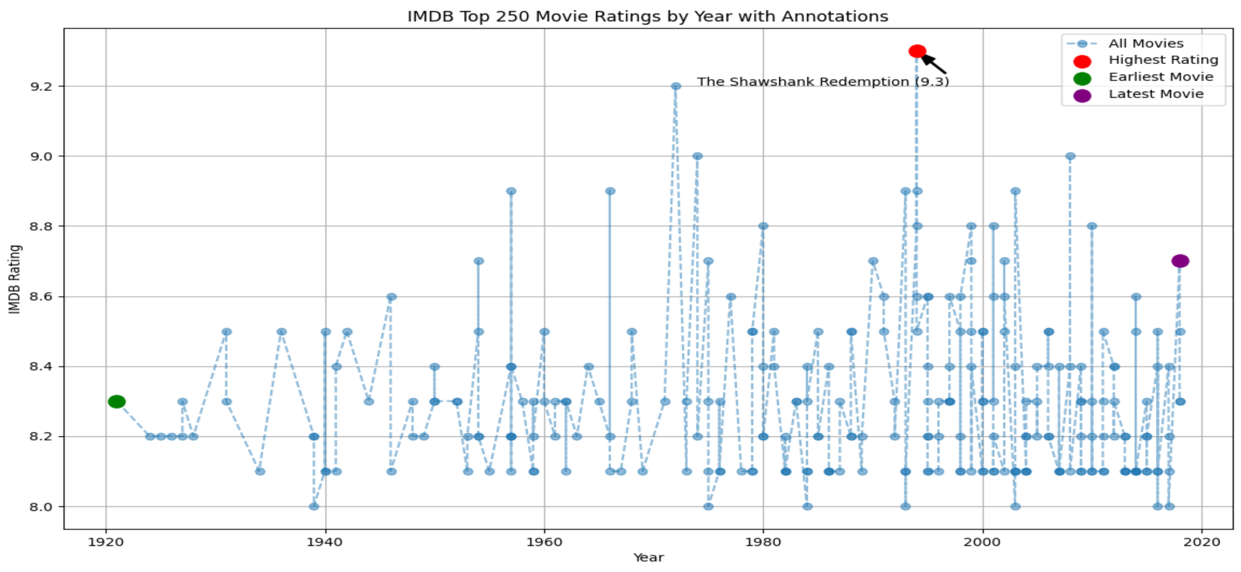
```python
# Author: Soroush Bastani (SBastani1)
# Date: 2025-11-07
# Purpose: Annotate key data points on a plot and save the figure.
# Usage: ./lab7g.py

import matplotlib
matplotlib.use('Agg') # Set non-GUI backend
import pandas as pd
import matplotlib.pyplot as plt

# 1. Read the CSV file
url = "https://raw.githubusercontent.com/itiievskyi/IMDB-Top-250/master/imdb_top_250.csv"
try:
    df = pd.read_csv(url)
except Exception as e:
    print(f"Error reading CSV file: {e}")
    exit()

# Sort data by year for a clean line plot
df = df.sort_values('Year')

# 2. Create the line plot
plt.figure(figsize=(14, 8))
# CORRECTED LINE: Use 'IMDB rating' instead of 'Rating'
plt.plot(df['Year'], df['IMDB rating'], linestyle='--', marker='o', alpha=0.5, label='All Movies')

# 3. Highlight key points
# CORRECTED LINES: Use 'IMDB rating' to find the max rating
highest_rating_movie = df.loc[df['IMDB rating'].idxmax()]
earliest_movie = df.loc[df['Year'].idxmin()]
latest_movie = df.loc[df['Year'].idxmax()]
```

```
/home/codespace/.python/current/bin/python /workspaces/lab
-7-Soroush-Bastani/lab7g.py
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main
) $ /home/codespace/.python/current/bin/python /workspaces
/lab-7-Soroush-Bastani/lab7g.py
Annotated plot saved to lab7g_annotated_plot.png
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main
) $
```



```python
plt.plot(df['Year'], df['IMDB rating'], linestyle='--', marker='o', alpha=0.5, label='All Movies')

# 3. Highlight key points
# CORRECTED LINES: Use 'IMDB rating' to find the max rating
highest_rating_movie = df.loc[df['IMDB rating'].idxmax()]
earliest_movie = df.loc[df['Year'].idxmin()]
latest_movie = df.loc[df['Year'].idxmax()]

# Mark these points on the plot
plt.scatter(highest_rating_movie['Year'], highest_rating_movie['IMDB rating'], color='red', s=120, zorde
plt.scatter(earliest_movie['Year'], earliest_movie['IMDB rating'], color='green', s=120, zorder=5, label
plt.scatter(latest_movie['Year'], latest_movie['IMDB rating'], color='purple', s=120, zorder=5, label='L

# 4. Annotate the highest-rated movie
plt.annotate(
    f"{highest_rating_movie['Title']} ({highest_rating_movie['IMDB rating']})",
    xy=(highest_rating_movie['Year'], highest_rating_movie['IMDB rating']),
    xytext=(highest_rating_movie['Year'] - 20, highest_rating_movie['IMDB rating'] - 0.1),
    arrowprops=dict(facecolor='black', shrink=0.05, width=1, headwidth=8)
)

# 5. Add titles, labels, and legend
plt.title('IMDB Top 250 Movie Ratings by Year with Annotations')
plt.xlabel('Year')
plt.ylabel('IMDB Rating') # Also updated the label for accuracy
plt.legend()
plt.grid(True)

# 6. Save the figure to a file
plt.savefig("lab7g_annotated_plot.png")
print("Annotated plot saved to lab7g_annotated_plot.png")
```

```
/home/codespace/.python/current/bin/python /workspaces/lab
-7-Soroush-Bastani/lab7g.py
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main
) $ /home/codespace/.python/current/bin/python /workspaces
/lab-7-Soroush-Bastani/lab7g.py
Annotated plot saved to lab7g_annotated_plot.png
@Soroush-Bastani ➜/workspaces/lab-7-Soroush-Bastani (main
) $
```



Thanks for reading!