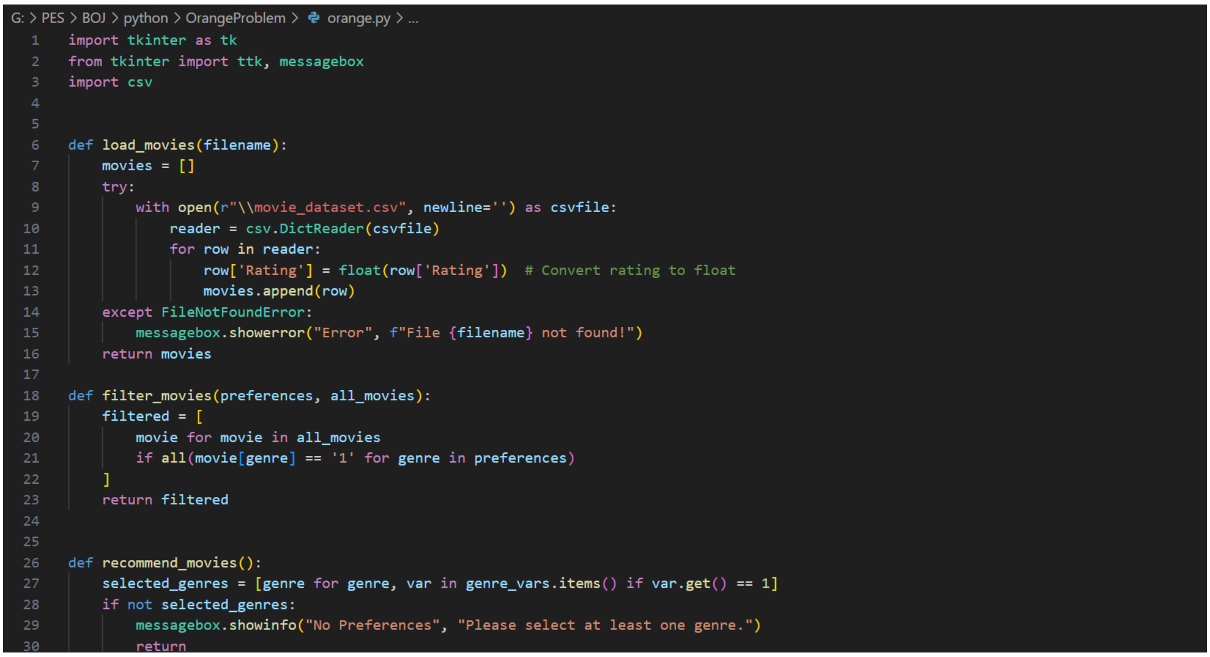
Orange Problem Solving

Srn: PES2UG24CS311

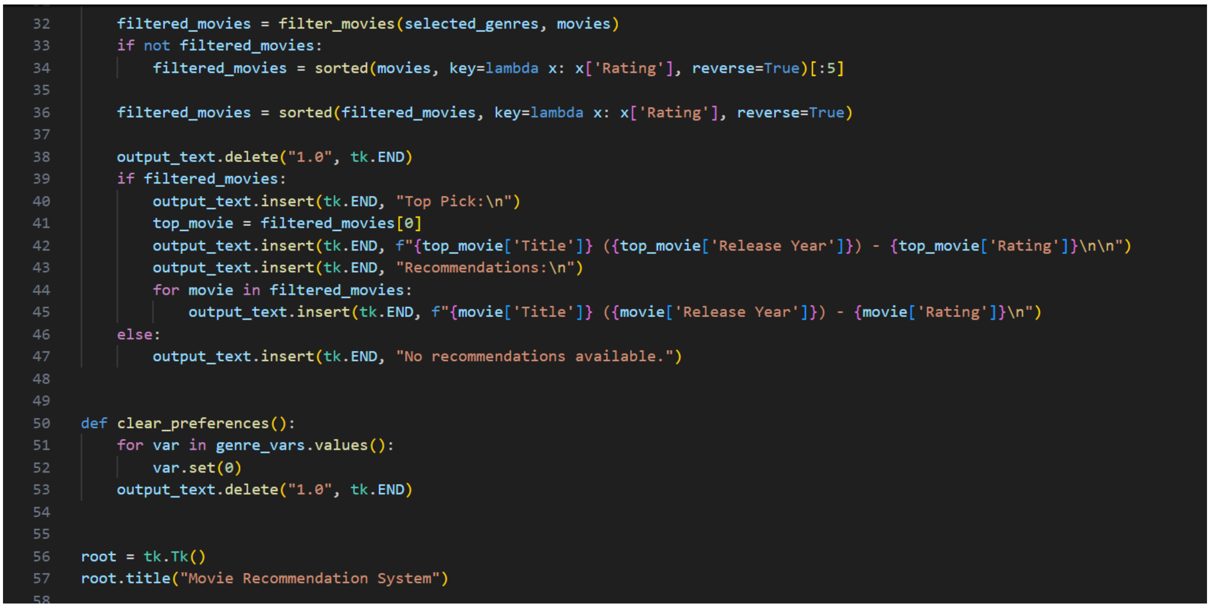
Section: D

Code:

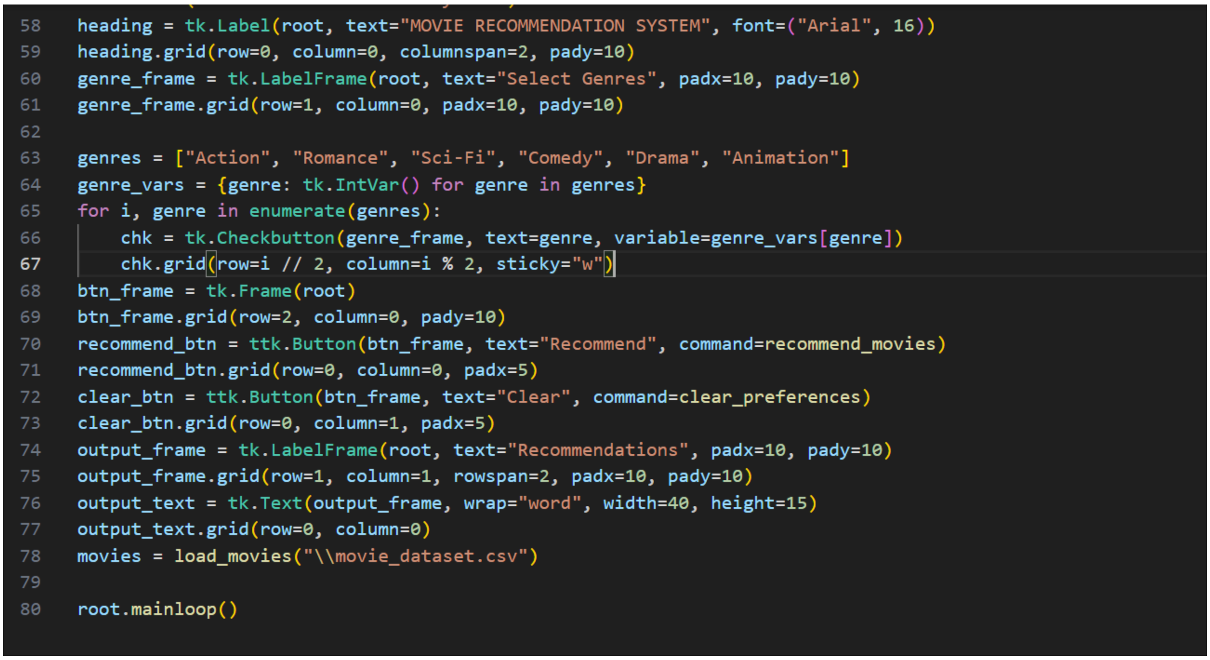
```
import tkinter as tk
from tkinter import ttk, messagebox
import csv

def load_movies(filename):
    movies = []
    try:
        with open(r"movie_dataset.csv", newline='') as csvfile:
            reader = csv.DictReader(csvfile)
            for row in reader:
                row['Rating'] = float(row['Rating'])  # Convert rating to float
                movies.append(row)
    except FileNotFoundError:
        messagebox.showerror("Error", f"File {filename} not found!")
    return movies

def filter_movies(preferences, all_movies):
    filtered = [
        movie for movie in all_movies
        if all(movie[genre] == '1' for genre in preferences)
    ]
    return filtered

def recommend_movies():
    selected_genres = [genre for genre, var in genre_vars.items() if var.get() == 1]
    if not selected_genres:
        messagebox.showinfo("No Preferences", "Please select at least one genre.")
        return

    filtered_movies = filter_movies(selected_genres, movies)
    if not filtered_movies:
        # If no match, show top 5 rated movies
        filtered_movies = sorted(movies, key=lambda x: x['Rating'], reverse=True)[:5]

    # Sort filtered movies by rating
    filtered_movies = sorted(filtered_movies, key=lambda x: x['Rating'], reverse=True)

    # Display results
    output_text.delete("1.0", tk.END)
    if filtered_movies:
        output_text.insert(tk.END, "Top Pick:\n")
        top_movie = filtered_movies[0]
        output_text.insert(tk.END, f"{top_movie['Title']} ({top_movie['Release Year']}) - 
{top_movie['Rating']}\n\n")
        output_text.insert(tk.END, "Recommendations:\n")
        for movie in filtered_movies:
            output_text.insert(tk.END, f"{movie['Title']} ({movie['Release Year']}) - 
{movie['Rating']}\n")
    else:
        output_text.insert(tk.END, "No recommendations available.")

# Clear preferences
def clear_preferences():
    for var in genre_vars.values():
        var.set(0)
    output_text.delete("1.0", tk.END)


root = tk.Tk()
root.title("Movie Recommendation System")
heading = tk.Label(root, text="MOVIE RECOMMENDATION SYSTEM",
font=("Arial", 16))
heading.grid(row=0, column=0, columnspan=2, pady=10)
genre_frame = tk.LabelFrame(root, text="Select Genres", padx=10, pady=10)
genre_frame.grid(row=1, column=0, padx=10, pady=10)
genres = ["Action", "Romance", "Sci-Fi", "Comedy", "Drama", "Animation"]
genre_vars = {genre: tk.IntVar() for genre in genres}
for i, genre in enumerate(genres):
    chk = tk.Checkbutton(genre_frame, text=genre, variable=genre_vars[genre])
    chk.grid(row=i // 2, column=i % 2, sticky="w")
btn_frame = tk.Frame(root)
btn_frame.grid(row=2, column=0, pady=10)
recommend_btn = ttk.Button(btn_frame, text="Recommend",
command=recommend_movies)
recommend_btn.grid(row=0, column=0, padx=5)
clear_btn = ttk.Button(btn_frame, text="Clear", command=clear_preferences)
clear_btn.grid(row=0, column=1, padx=5)
output_frame = tk.LabelFrame(root, text="Recommendations", padx=10, pady=10)
output_frame.grid(row=1, column=1, rowspan=2, padx=10, pady=10)
output_text = tk.Text(output_frame, wrap="word", width=40, height=15)
output_text.grid(row=0, column=0)
movies = load_movies("movie_dataset.csv")
root.mainloop()
```



```python
import tkinter as tk
from tkinter import ttk, messagebox
import csv


def load_movies(filename):
    movies = []
    try:
        with open(r"\\movie_dataset.csv", newline='') as csvfile:
            reader = csv.DictReader(csvfile)
            for row in reader:
                row['Rating'] = float(row['Rating'])  # Convert rating to float
                movies.append(row)
    except FileNotFoundError:
        messagebox.showerror("Error", f"File {filename} not found!")
    return movies

def filter_movies(preferences, all_movies):
    filtered = [
        movie for movie in all_movies
        if all(movie[genre] == '1' for genre in preferences)
    ]
    return filtered


def recommend_movies():
    selected_genres = [genre for genre, var in genre_vars.items() if var.get() == 1]
    if not selected_genres:
        messagebox.showinfo("No Preferences", "Please select at least one genre.")
        return
```

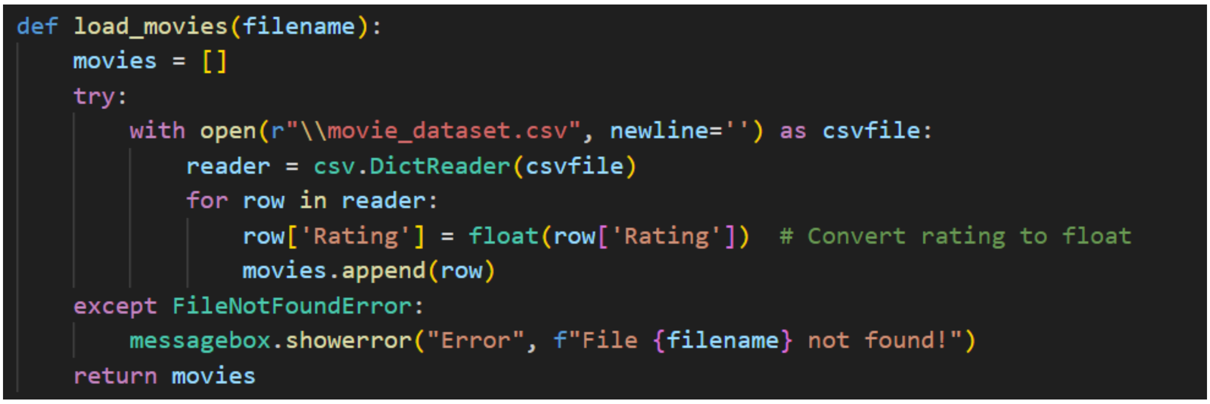


```python
    filtered_movies = filter_movies(selected_genres, movies)
    if not filtered_movies:
        filtered_movies = sorted(movies, key=lambda x: x['Rating'], reverse=True)[:5]

    filtered_movies = sorted(filtered_movies, key=lambda x: x['Rating'], reverse=True)

    output_text.delete("1.0", tk.END)
    if filtered_movies:
        output_text.insert(tk.END, "Top Pick:\n")
        top_movie = filtered_movies[0]
        output_text.insert(tk.END, f"{top_movie['Title']} ({top_movie['Release Year']}) - {top_movie['Rating']}\n\n")
        output_text.insert(tk.END, "Recommendations:\n")
        for movie in filtered_movies:
            output_text.insert(tk.END, f"{movie['Title']} ({movie['Release Year']}) - {movie['Rating']}\n")
    else:
        output_text.insert(tk.END, "No recommendations available.")


def clear_preferences():
    for var in genre_vars.values():
        var.set(0)
    output_text.delete("1.0", tk.END)


root = tk.Tk()
root.title("Movie Recommendation System")
```

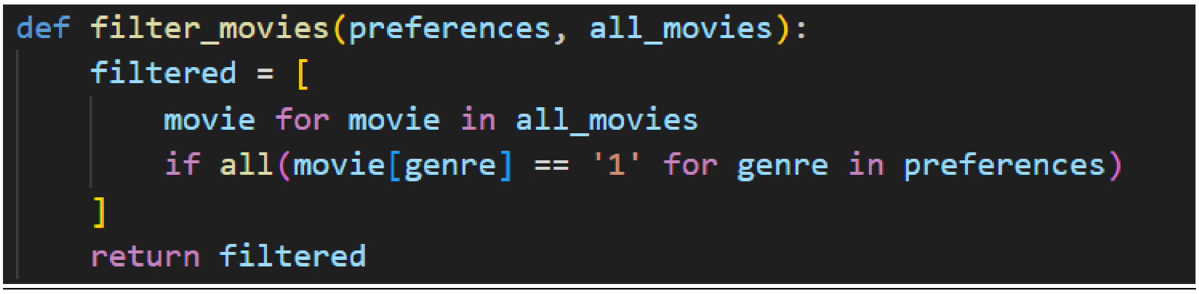


```python
heading = tk.Label(root, text="MOVIE RECOMMENDATION SYSTEM", font=("Arial", 16))
heading.grid(row=0, column=0, columnspan=2, pady=10)
genre_frame = tk.LabelFrame(root, text="Select Genres", padx=10, pady=10)
genre_frame.grid(row=1, column=0, padx=10, pady=10)

genres = ["Action", "Romance", "Sci-Fi", "Comedy", "Drama", "Animation"]
genre_vars = {genre: tk.IntVar() for genre in genres}
for i, genre in enumerate(genres):
    chk = tk.Checkbutton(genre_frame, text=genre, variable=genre_vars[genre])
    chk.grid(row=i // 2, column=i % 2, sticky="w")
btn_frame = tk.Frame(root)
btn_frame.grid(row=2, column=0, pady=10)
recommend_btn = ttk.Button(btn_frame, text="Recommend", command=recommend_movies)
recommend_btn.grid(row=0, column=0, padx=5)
clear_btn = ttk.Button(btn_frame, text="Clear", command=clear_preferences)
clear_btn.grid(row=0, column=1, padx=5)
output_frame = tk.LabelFrame(root, text="Recommendations", padx=10, pady=10)
output_frame.grid(row=1, column=1, rowspan=2, padx=10, pady=10)
output_text = tk.Text(output_frame, wrap="word", width=40, height=15)
output_text.grid(row=0, column=0)
movies = load_movies("\\movie_dataset.csv")

root.mainloop()
```

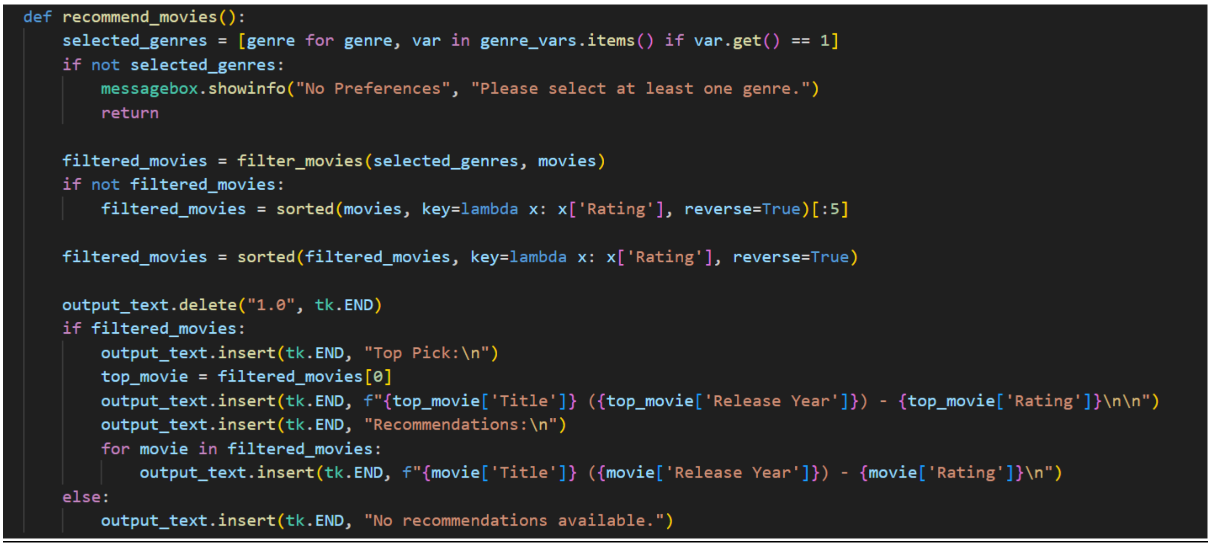
Functions:



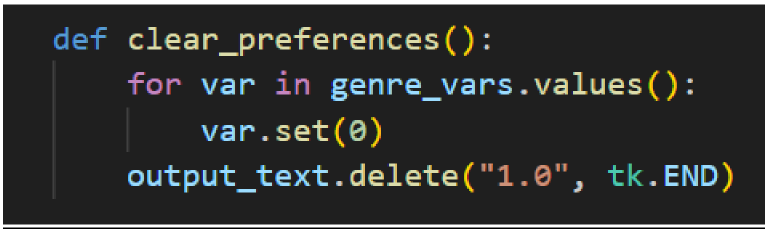
```python
def load_movies(filename):
    movies = []
    try:
        with open(r"\\movie_dataset.csv", newline='') as csvfile:
            reader = csv.DictReader(csvfile)
            for row in reader:
                row['Rating'] = float(row['Rating'])  # Convert rating to float
                movies.append(row)
    except FileNotFoundError:
        messagebox.showerror("Error", f"File {filename} not found!")
    return movies
```



```python
def filter_movies(preferences, all_movies):
    filtered = [
        movie for movie in all_movies
        if all(movie[genre] == '1' for genre in preferences)
    ]
    return filtered
```

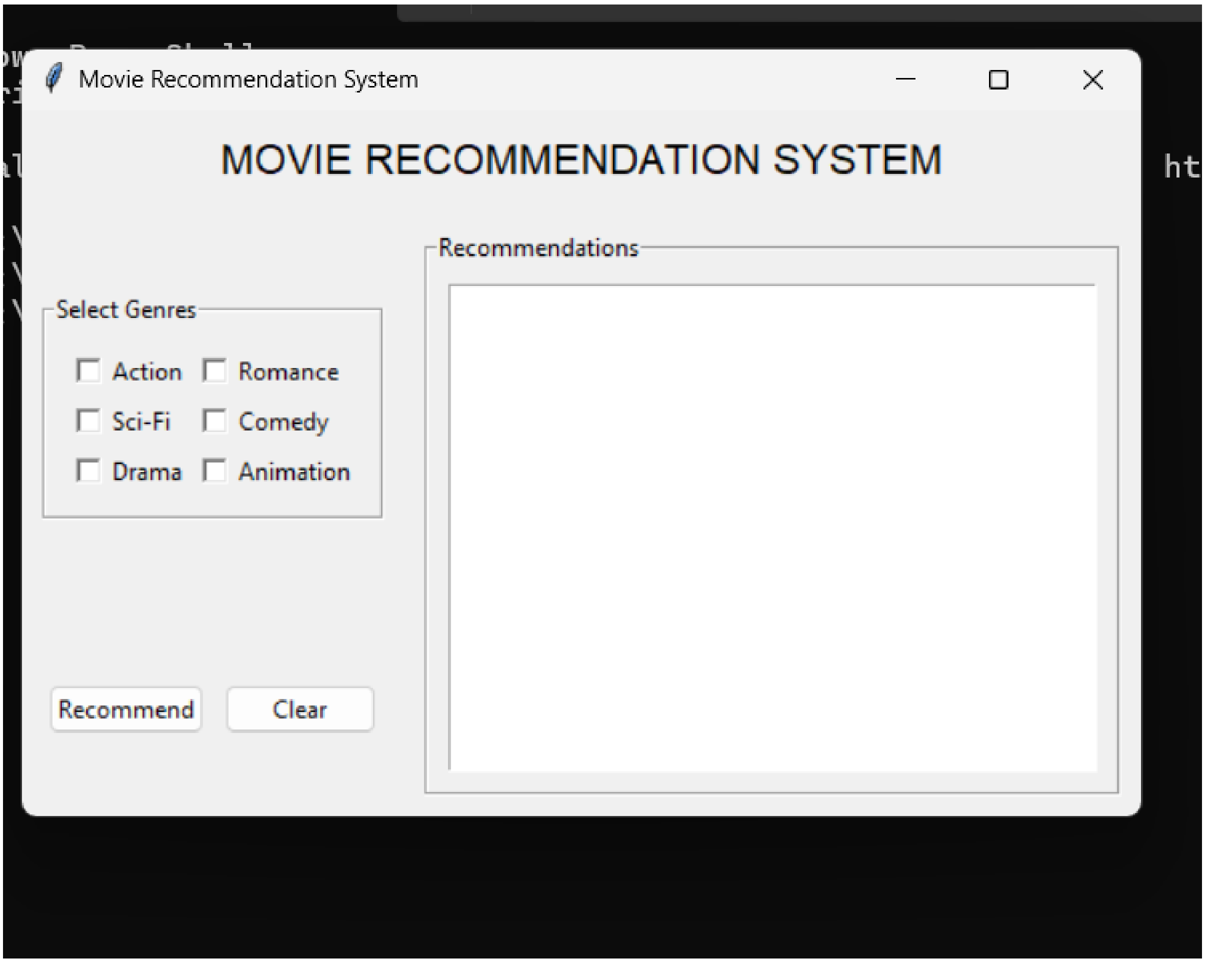


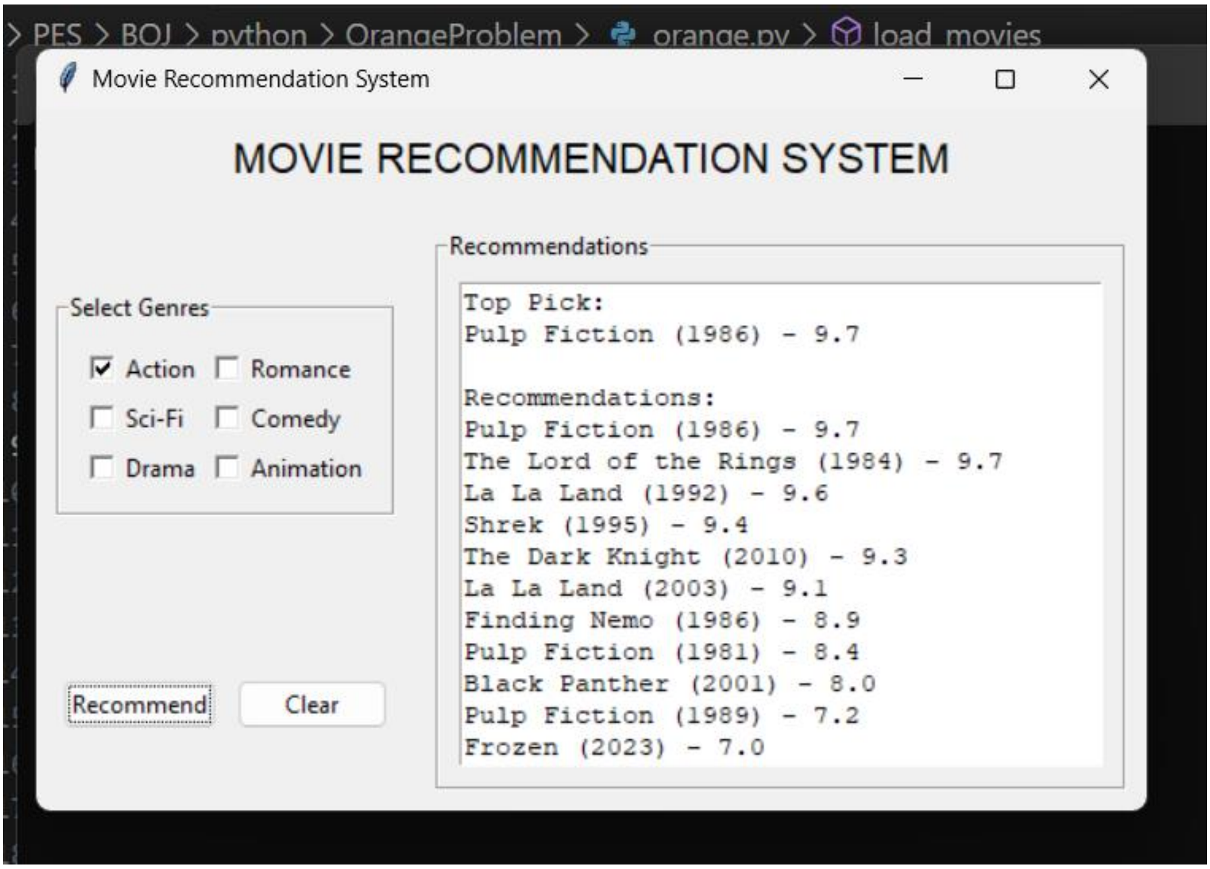
```python
def recommend_movies():
    selected_genres = [genre for genre, var in genre_vars.items() if var.get() == 1]
    if not selected_genres:
        messagebox.showinfo("No Preferences", "Please select at least one genre.")
        return

    filtered_movies = filter_movies(selected_genres, movies)
    if not filtered_movies:
        filtered_movies = sorted(movies, key=lambda x: x['Rating'], reverse=True)[:5]

    filtered_movies = sorted(filtered_movies, key=lambda x: x['Rating'], reverse=True)

    output_text.delete("1.0", tk.END)
    if filtered_movies:
        output_text.insert(tk.END, "Top Pick:\n")
        top_movie = filtered_movies[0]
        output_text.insert(tk.END, f"{top_movie['Title']} ({top_movie['Release Year']}) - {top_movie['Rating']}\n\n")
        output_text.insert(tk.END, "Recommendations:\n")
        for movie in filtered_movies:
            output_text.insert(tk.END, f"{movie['Title']} ({movie['Release Year']}) - {movie['Rating']}\n")
    else:
        output_text.insert(tk.END, "No recommendations available.")
```



```python
def clear_preferences():
    for var in genre_vars.values():
        var.set(0)
    output_text.delete("1.0", tk.END)
```

Output:


Movie Recommendation System
MOVIE RECOMMENDATION SYSTEM
Select Genres
□ Action
□ Romance
□ Sci-Fi
□ Comedy
□ Drama
□ Animation
Recommend
Clear
Recommendations


> PES > BOJ > python > OrangeProblem > orange.py > load_movies
Movie Recommendation System
Recommend
Clear

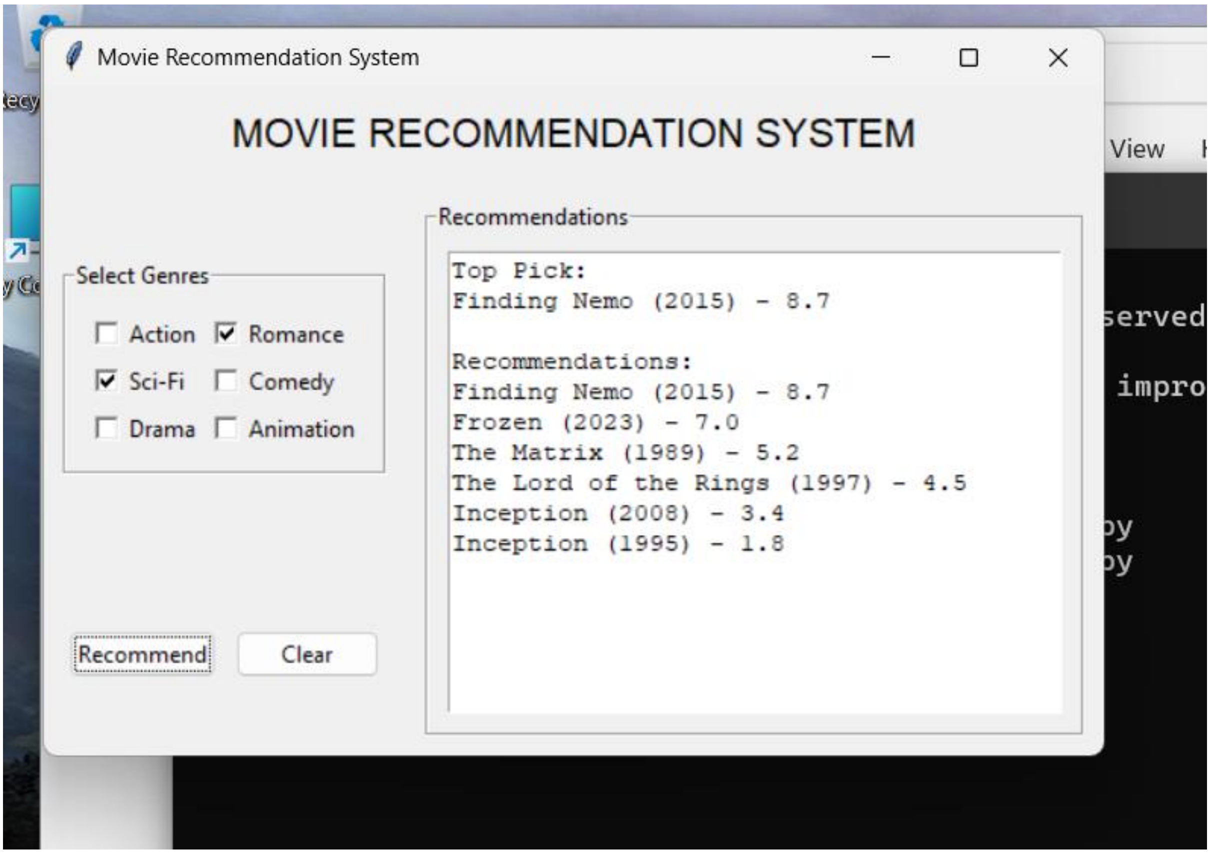
MOVIE RECOMMENDATION SYSTEM

Select Genres

☑ Action ☐ Romance
☐ Sci-Fi ☐ Comedy
☐ Drama ☐ Animation

Recommendations

```
Top Pick:
Pulp Fiction (1986) - 9.7

Recommendations:
Pulp Fiction (1986) - 9.7
The Lord of the Rings (1984) - 9.7
La La Land (1992) - 9.6
Shrek (1995) - 9.4
The Dark Knight (2010) - 9.3
La La Land (2003) - 9.1
Finding Nemo (1986) - 8.9
Pulp Fiction (1981) - 8.4
Black Panther (2001) - 8.0
Pulp Fiction (1989) - 7.2
Frozen (2023) - 7.0
```



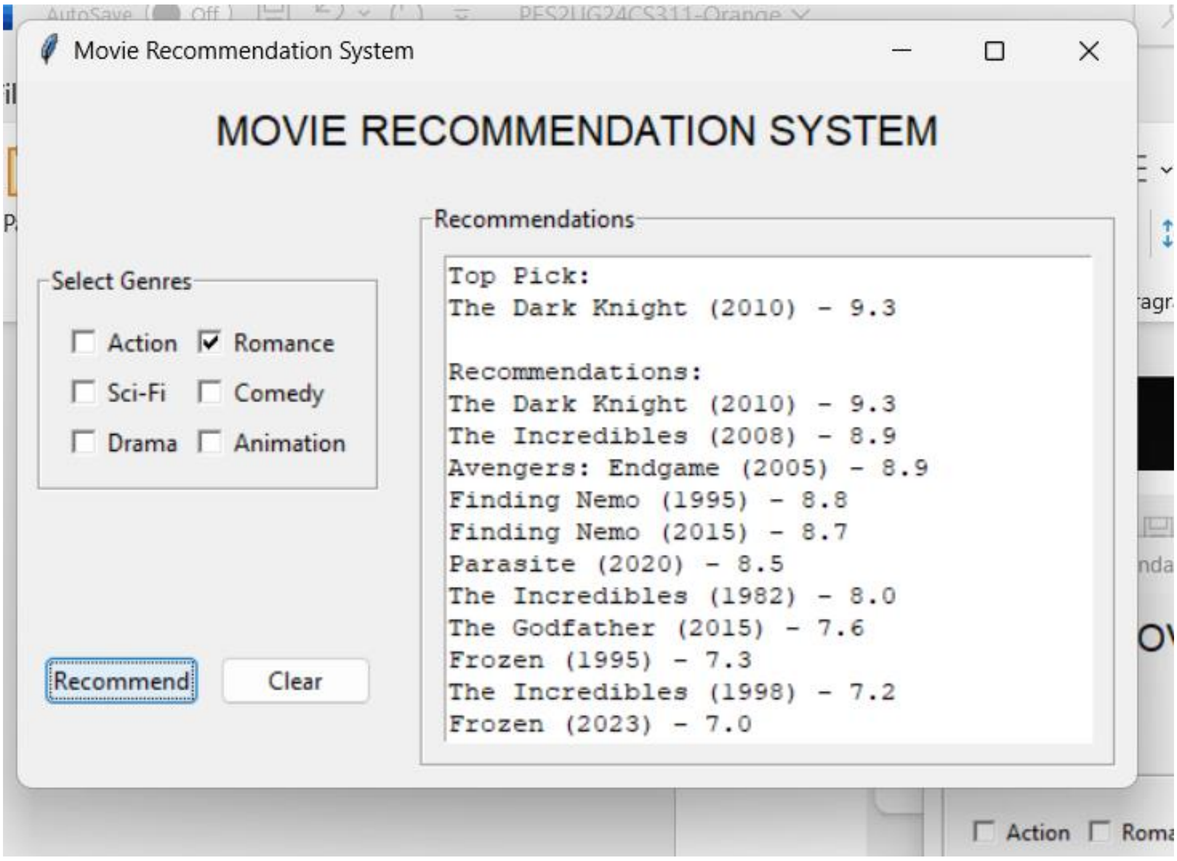
MOVIE RECOMMENDATION SYSTEM

Select Genres

☐ Action ☑ Romance
☑ Sci-Fi ☐ Comedy
☐ Drama ☐ Animation

Recommendations

```
Top Pick:
Finding Nemo (2015) - 8.7

Recommendations:
Finding Nemo (2015) - 8.7
Frozen (2023) - 7.0
The Matrix (1989) - 5.2
The Lord of the Rings (1997) - 4.5
Inception (2008) - 3.4
Inception (1995) - 1.8
```



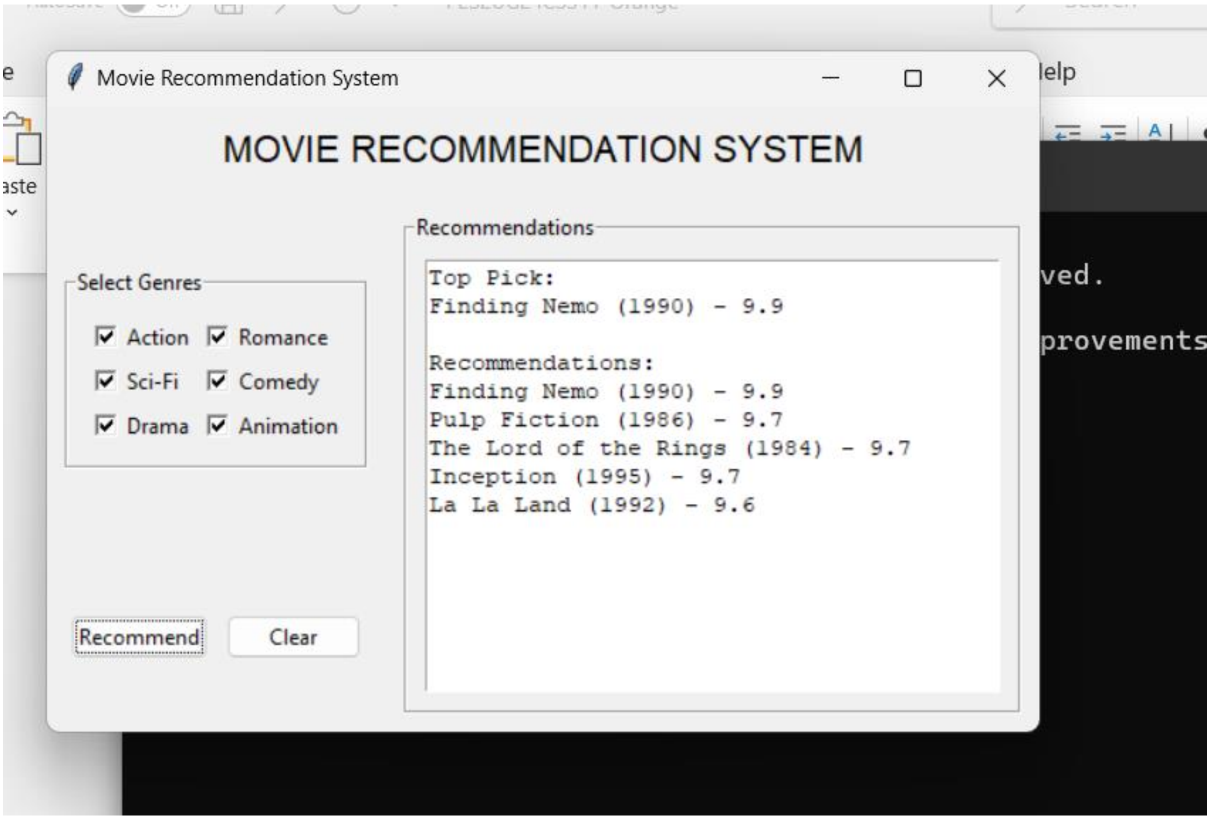
MOVIE RECOMMENDATION SYSTEM

Select Genres

□ Action ☑ Romance
□ Sci-Fi □ Comedy
□ Drama □ Animation

Recommendations

```
Top Pick:
The Dark Knight (2010) - 9.3

Recommendations:
The Dark Knight (2010) - 9.3
The Incredibles (2008) - 8.9
Avengers: Endgame (2005) - 8.9
Finding Nemo (1995) - 8.8
Finding Nemo (2015) - 8.7
Parasite (2020) - 8.5
The Incredibles (1982) - 8.0
The Godfather (2015) - 7.6
Frozen (1995) - 7.3
The Incredibles (1998) - 7.2
Frozen (2023) - 7.0
```



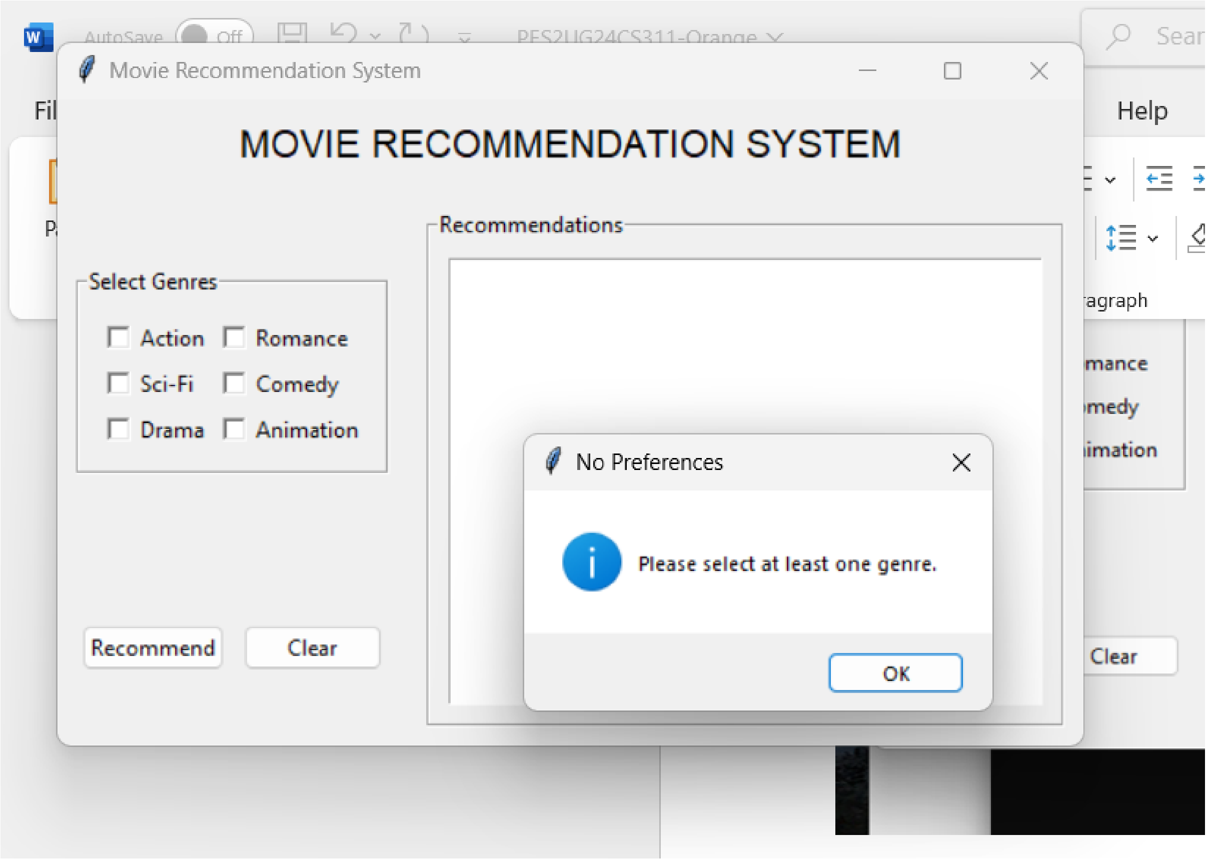
MOVIE RECOMMENDATION SYSTEM

Select Genres

☑ Action ☑ Romance
☑ Sci-Fi ☑ Comedy
☑ Drama ☑ Animation

Recommendations

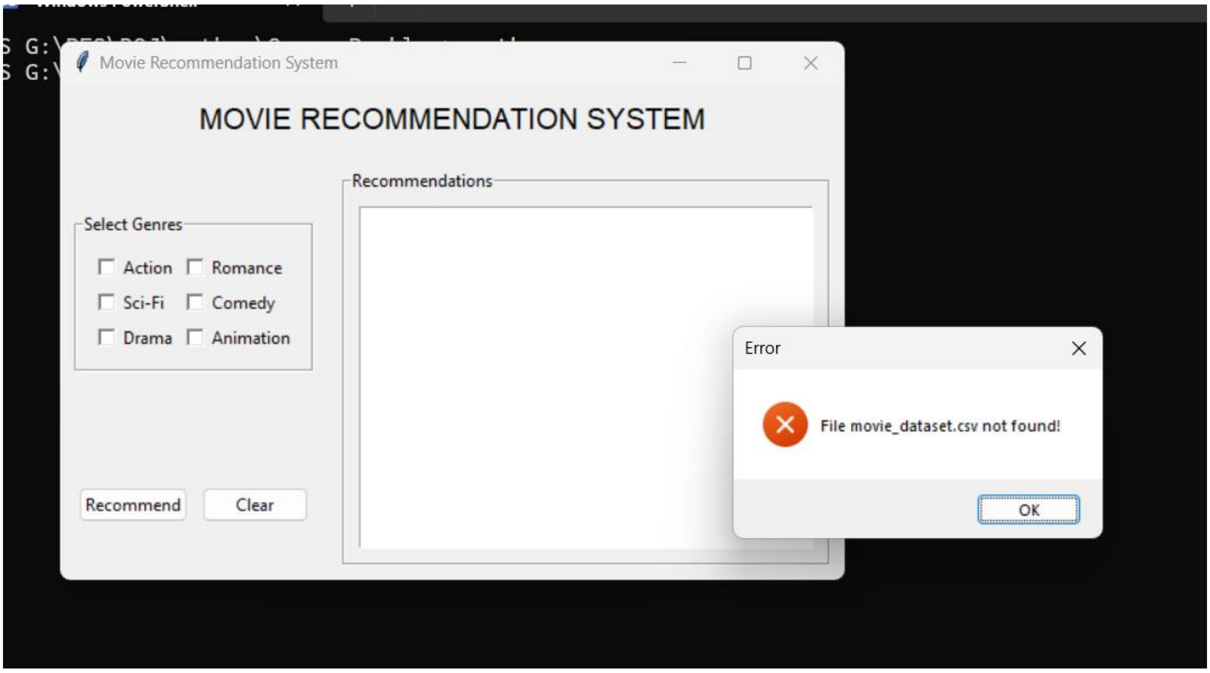
```
Top Pick:
Finding Nemo (1990) - 9.9

Recommendations:
Finding Nemo (1990) - 9.9
Pulp Fiction (1986) - 9.7
The Lord of the Rings (1984) - 9.7
Inception (1995) - 9.7
La La Land (1992) - 9.6
```



MOVIE RECOMMENDATION SYSTEM

No Preferences

Please select at least one genre.



MOVIE RECOMMENDATION SYSTEM

Error

File movie_dataset.csv not found!