1) Movie recommendation systems are algorithms or tools designed to suggest movies to users based on factors such as preference, rating, genre, viewing history, etc. Netflix, YouTube, and Amazon Prime are all examples of movie recommendation systems that are used daily all across the globe.
   The three main types of filtering used for recommendations are **content-based filtering, collaborative filtering,** and **hybrid systems**. In **user-based** collaborative filtering, the systems finds users similar to your tastes and recommends movies they would like. In **item-based** collaborative filtering, the system finds movies similar to those you liked, and recommends those movies. User-based systems are intuitive and easy to implement, while item-based systems are more scalable and stable.
   The **random walk** approach builds a bipartite graph to represent users and items, and simulates a random walk to discover connections between those users and items. An edge on the graph represents an interaction between a user and a movie, such as likes and ratings. This method captures deep, complex relationships and works well in cold start scenarios with limited data.

2) The MovieLens 100K dataset contained a large amount of data containing users, movies, and ratings. There were a total of 943 unique users, 1681 unique movies, and 100,000 total ratings. Each rating was on a scale from 1 to 5 depending on how much a user liked a movie. The systems used were user-based filtering, item-based filtering, pixie inspired graph algorithm, and random walk recommendation system (weighted pixie). Each of the mini datasets were converted into cleaned matrices without duplicate values using pandas dataframes.

3) The **user based** collaborative filtering technique recommended the top 5 highest rated movies to users based on the preferences of similar users. The **item based** collaborative filtering technique recommended the top 5 highest rated movies that were similar to a given movie based on the user's ratings. For example, the output listed the top 5 movies similar to Jurassic Park that the user would like. The **random walk pixie based** technique used a user-movie bipartite graph to recommend the top 5 highest rated movies by simulating a random walk for a given user or movie. The walk would start at a user, and at each step, randomly move to a connected node and keep track of how many times each movie is visited. After completing the walk, the output displays the top 5 movies with the highest visit count. This approach is very effective because it recommends the most popular and high rated movies and displays all user-movie relationships on a graph.

4) The **user based** and **item based** collaborative filtering functions used cosine similarity and the merge method to merge the movie IDs with movie names and display the top rated movies.
   The adjacency list used the merge, groupby, and transform methods and a for loop to iterate through the rows in the list. The nodes in the graph represent users and movies. The graph for a user represents the movie IDs of all the movies the user has rated. The graph for a movie represents the user IDs of all the users that have rated that movie.

The random walk was performed by assigning a user to a node and looping through the walk length to find neighbors. The random.choice function was used to randomly select the next node in the graph. The visited movies were ranked by a visit count, which is the number of times a movie was visited in the graph.

5) Outputs:
User based collaborative filtering

	Ranking	title
0	1	Star Kid (1997)
1	2	Santa with Muscles (1996)
2	3	Entertaining Angels: The Dorothy Day Story (1996)
3	4	Prefontaine (1997)
4	5	Marlene Dietrich: Shadow and Light (1996)

- **Pros**: high personalization, simple to implement
- **Cons**: poor scalability, struggles with sparse data, user preferences change

Item based collaborative filtering

	Ranking	title
0	1	Top Gun (1986)
1	2	Speed (1994)
2	3	Raiders of the Lost Ark (1981)
3	4	Empire Strikes Back, The (1980)
4	5	Indiana Jones and the Last Crusade (1989)

- **Pros**: more scalable, more stable, good with sparse data
- **Cons**: less personalization, cold start

Random walk

	Ranking	title	visit_count
0	1	Immortal Beloved (1994)	2
1	2	Dante's Peak (1997)	1
2	3	Liar Liar (1997)	1
3	4	Man Who Knew Too Little, The (1997)	1
4	5	Target (1995)	1

- **Pros**: deeper connections, flexible, cold start
- **Cons**: intensive, hard to interpret, requires graph construction

6) I learned how to display recommended movies to users using user based and item based collaborative filtering and random walks. I used pandas dataframes, data cleaning, and data exploration functions editing or removing values in a dataset.
The recommendation system can be improved with **hybrid models**, which combine multiple approaches. For example, a mix of content based and collaborative filtering can use both user based and item based techniques to recommend movies and can help in cold start situations where a movie or user is new. More **advanced options** would be to use matrix factorization, deep learning, or graph neural networks (GNNs) for additional features.

A few **real world applications** that use recommendation systems are streaming platforms such as Netflix and Hulu, e-commerce sites such as Amazon and Ebay, social media platforms such as YouTube and TikTok, and education platforms such as Coursera and Khan Academy.