In [2]:

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

In [6]:

```
movies = pd.read_csv(r"C:\Users\Rachitha\Desktop\Movie-Recommender-System\Dataset\movies.csv")
movies.head()
```

Out[6]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

In [7]:

```
print("Missing genres:", movies['genres'].isnull().sum())
movies['genres'] = movies['genres'].fillna('')
```

```
Missing genres: 0
```

In [8]:

```
tfidf = TfidfVectorizer(stop_words='english')
tfidf_matrix = tfidf.fit_transform(movies['genres'])
print("TF-IDF Matrix Shape:", tfidf_matrix.shape)
```

```
TF-IDF Matrix Shape: (9742, 23)
```

In [15]:

```
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)
print("Cosine Similarity Matrix Shape :", cosine_sim.shape)
```

```
Cosine Similarity Matrix Shape : (9742, 9742)
```

In [16]:

```
indices = pd.Series(movies.index , index=movies['title']).drop_duplicates()
print("\nMovie title to index mapping (first 10):")
print(indices.head(10))
```

```
Movie title to index mapping (first 10):
title
Toy Story (1995)                      0
Jumanji (1995)                        1
Grumpier Old Men (1995)               2
Waiting to Exhale (1995)              3
Father of the Bride Part II (1995)    4
Heat (1995)                           5
Sabrina (1995)                        6
Tom and Huck (1995)                   7
Sudden Death (1995)                   8
GoldenEye (1995)                      9
dtype: int64
```

In [22]:

```
def get_recommendations(title,cosine_sim=cosine_sim):
    idx = indices.get(title)
    if idx is None:
        return "Movie not found in dataset."
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = sorted(sim_scores , key=lambda x: x[1], reverse=True)[1:6]
    movie_indices = [i[0] for i in sim_scores]
    return movies['title'].iloc[movie_indices].tolist()
```

In [29]:

```
print("Recommendations for 'Toy Story (1995)':")
print(get_recommendations("Toy Story (1995)"))
print("\nRecommendations for 'Heat (1995)':")
print(get_recommendations("Heat (1995)"))
```

```
Recommendations for 'Toy Story (1995)':
['Antz (1998)', 'Toy Story 2 (1999)', 'Adventures of R
ocky and Bullwinkle, The (2000)', "Emperor's New Groov
e, The (2000)", 'Monsters, Inc. (2001)']

Recommendations for 'Heat (1995)':
['Assassins (1995)', 'Die Hard: With a Vengeance (199
5)', 'Net, The (1995)', 'Natural Born Killers (1994)',
'Judgment Night (1993)']
```