## 6. Write a prolog program for the family tree.

```
% Facts

male(raj).

male(amit).

male(ravi).

male(arjun).


female(sita).

female(gita).

female(anita).

female(kavya).


parent(raj, amit).

parent(sita, amit).

parent(raj, gita).

parent(sita, gita).

parent(amit, ravi).

parent(anita, ravi).

parent(amit, kavya).

parent(anita, kavya).


% Rules

father(X, Y) :- male(X), parent(X, Y).

mother(X, Y) :- female(X), parent(X, Y).

grandparent(X, Y) :- parent(X, Z), parent(Z, Y).

sibling(X, Y) :- parent(Z, X), parent(Z, Y), X \= Y.
```

**Sample Queries:**

1. **father(raj, gita).**
2. **sibling(amit, gita).**
3. **grandparent(raj, kavya).**

**7. Write a program to solve N-Queens problem using Prolog.**

```
% Entry point
n_queens(N, Solution) :-
    range(1, N, Ns),
    permutation(Ns, Solution),
    safe(Solution).

% Create a list of numbers from Low to High
range(High, High, [High]).
range(Low, High, [Low|Rest]) :-
    Low < High,
    Next is Low + 1,
    range(Next, High, Rest).

% Ensure no two queens are attacking diagonally
safe([]).
safe([Queen|Others]) :-
    safe(Others),
    no_attack(Queen, Others, 1).

no_attack(_, [], _).
no_attack(Y, [Y1|YList], XDist) :-
    Y =\= Y1 + XDist,
    Y =\= Y1 - XDist,
    NextX is XDist + 1,
    no_attack(Y, YList, NextX).
```

**Sample Queries:**

```
n_queens(4, Solution).
```

**8. Write a program to solve 8 puzzle problem using Prolog.**

```
% Define the goal state
goal([1,2,3,
    4,5,6,
    7,8,0]).

% Moves: up, down, left, right
move([A,B,C,D,E,F,G,H,0], [A,B,0,D,E,F,G,H,C]). % left
move([A,B,C,D,E,F,G,0,H], [A,B,C,D,E,F,0,G,H]). % left
move([A,B,C,D,E,F,0,G,H], [A,B,C,D,E,F,G,0,H]). % left

move([A,B,0,D,E,F,G,H,I], [A,0,B,D,E,F,G,H,I]). % right
move([A,B,C,D,0,F,G,H,I], [A,B,C,D,F,0,G,H,I]). % right
move([A,B,C,D,E,F,G,0,I], [A,B,C,D,E,F,G,I,0]). % right

move([A,B,C,0,E,F,G,H,I], [0,B,C,A,E,F,G,H,I]). % up
move([A,B,C,D,0,F,G,H,I], [A,0,C,D,B,F,G,H,I]). % up
move([A,B,C,D,E,0,G,H,I], [A,B,0,D,E,C,G,H,I]). % up
move([A,B,C,D,E,F,0,H,I], [A,B,C,0,E,F,D,H,I]). % up
move([A,B,C,D,E,F,G,0,I], [A,B,C,D,0,F,G,E,I]). % up
move([A,B,C,D,E,F,G,H,0], [A,B,C,D,E,0,G,H,F]). % up

move([0,B,C,D,E,F,G,H,I], [D,B,C,0,E,F,G,H,I]). % down
move([A,0,C,D,E,F,G,H,I], [A,E,C,D,0,F,G,H,I]). % down
move([A,B,0,D,E,F,G,H,I], [A,B,F,D,E,0,G,H,I]). % down
move([A,B,C,0,E,F,G,H,I], [A,B,C,G,E,F,0,H,I]). % down
move([A,B,C,D,0,F,G,H,I], [A,B,C,D,H,F,G,0,I]). % down
move([A,B,C,D,E,0,G,H,I], [A,B,C,D,E,I,G,H,0]). % down

% Solve the puzzle using BFS
solve_bfs(Start, Solution) :-
```

```
    bfs([[Start]], Solution).


bfs([[State|Path]|_], [State|Path]) :-
    goal(State).


bfs([[State|Path]|Rest], Solution) :-
    findall([Next,State|Path],
        (move(State, Next), \+ member(Next, [State|Path])),
        NextMoves),
    append(Rest, NextMoves, NewQueue),
    bfs(NewQueue, Solution).
```

**Sample Queries:**

```
solve_bfs([1,2,3,4,0,6,7,5,8], Solution).
```

# 9. Case Study: Artificial Intelligence in Daily Life

## Topic: AI in Voice Assistants (e.g., Google Assistant, Siri, Alexa)

### Overview

Voice assistants are AI-powered programs that understand spoken commands and respond accordingly. They use **Natural Language Processing (NLP)**, **Machine Learning (ML)**, and **Speech Recognition** to communicate with users and perform tasks.

### How It Works

1. **Speech Recognition**: Converts your voice to text.
2. **NLP (Natural Language Processing)**: Understands the meaning of your words.
3. **Machine Learning**: Learns from your behavior and improves answers over time.
4. **Task Execution**: Performs tasks like playing music, setting alarms, or searching the web.

### Example Use-Cases

- "Hey Google, what's the weather today?" → AI fetches real-time weather data.
- "Set a reminder for 5 PM." → AI creates calendar events.
- "Play my workout playlist." → AI connects to your music app.

### Benefits

- Hands-free operation
- Saves time and effort
- Useful for elderly or visually impaired users
- Available 24x7

### Technologies Used

- **Speech Recognition** (like `speech_recognition` library)
- **Text-to-Speech (TTS)**
- **Deep Learning** (for understanding language patterns)
- **Cloud AI services** (e.g., Google Cloud, Amazon Alexa SDK)

### Real-World Impact

- Widely used in homes, smartphones, and cars
- Boosts productivity and accessibility
- Continuously evolving with better language understanding

# Case Study: AI in Recommendation Systems

### Topic: AI in Recommendation Systems (e.g., Netflix, Amazon, YouTube)

### How It Works

There are two main types:

1. **Content-Based Filtering**
    - Recommends items similar to what the user liked before.
    - Example: If you watched romantic movies, it suggests more romantic ones.
2. **Collaborative Filtering**
    - Recommends items that users with similar preferences liked.
    - Example: If you and another user liked the same movies, their favorites are recommended to you.
3. **Hybrid Model**
    - Combines both content-based and collaborative filtering for better results.

### Example Use-Cases

| Platform | Use of AI in Recommendations |
|---|---|
| Netflix | Suggests movies and shows based on your watch history |
| Amazon | Recommends products based on browsing and purchase patterns |
| YouTube | Suggests videos based on your likes and viewing history |
| Spotify | Creates playlists based on your listening habits |

### Benefits

- Saves time for users

- Increases user engagement
- Personalized user experience
- Increases sales or views for platforms

## Technologies Used

- **Machine Learning Algorithms**
  (e.g., K-Nearest Neighbors, Matrix Factorization)
- **User Behavior Data**
- **Python Libraries**:
  - `pandas`, `scikit-learn`, `surprise`, `TensorFlow`

## Real-World Impact

- Boosts customer satisfaction and retention
- Drives revenue for companies
- Enhances user experience

# Python Code:

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

# List of Indian comedy movie descriptions
movies = [
    "Hera Pheri - Two tenants and a landlord trying to make quick money",
    "Phir Hera Pheri - A continuation of their hilarious misadventures",
    "Welcome - Gangster wants to marry his sister to a decent man",
    "Chup Chup Ke - Man pretends to be mute and deaf, comedy of errors",
    "Bhool Bhulaiyaa - Psychological thriller with a comic twist",
    "Golmaal - Group of boys trick an old blind couple for shelter",
    "Dhamaal - Four friends looking for hidden treasure",
    "Hungama - Misunderstandings and chaos between different characters",
    "Andaz Apna Apna - Two slackers compete to win a rich girl’s heart",
    "Munna Bhai M.B.B.S. - Gangster joins medical college for his father"
]

# Convert text to TF-IDF feature vectors
tfidf = TfidfVectorizer()
tfidf_matrix = tfidf.fit_transform(movies)

# Compute cosine similarity matrix
similarity = cosine_similarity(tfidf_matrix)

# Recommend movies similar to 'Hera Pheri' (index 0)
selected_index = 0
recommended = similarity[selected_index].argsort()[::-1][1:4]  # Top 3
excluding itself
```

```
print("🎬 Movies similar to:", movies[selected_index].split(" - ")[0])
for idx in recommended:
    print("👉", movies[idx].split(" - ")[0])
```

Output:

```
Movies similar to: Hera Pheri
👉 Phir Hera Pheri
👉 Andaz Apna Apna
👉 Welcome
```