Object-Oriented Programming (OOP) Project: Social Network System

Introduction

This project involves designing and implementing a **social network system** using **Object-Oriented Programming (OOP) principles**. The system models a **gaming community** where users share their favorite games and manage their profiles dynamically.**

Class Definitions and Responsibilities

1. `Person` Class (Represents an individual user in the network)

Each `Person` instance represents a user with a unique name and a list of favorite games.

Attributes (Properties)

Attribute	Type	Description
`name`	`String`	The unique name of the user.
`games`	`List<String>`	A list of favorite games played by the user.

Methods (Behaviors)

Method	Description
`addGame(String game)`	Adds a game to the user's favorite games list.
`removeGame(String game)`	Removes a game from the favorite games list.
`getFavoriteGames() -> List<String>`	Returns a list of the user's favorite games.
`getName() -> String`	Returns the name of the user.

2. `SocialNetwork` Class (Manages the entire network of users)

The `SocialNetwork` class represents the entire network and is responsible for managing the users and their relationships.

Attributes (Properties)

Attribute	Type	Description
`person`	`Person`	A reference to a specific user (for example, the currently logged-in user).
`users`	`List<Person>`	A list of all `Person` objects representing users in the network.

Methods (Behaviors)

Method	Description
`addUser(Person user)`	Adds a new user to the network.
`removeUser(String name)`	Removes a user from the network based on the unique name.
`getUser(String name) -> Person`	Retrieves a `Person` object given a name.
`updatePerson(Person person)`	Updates the `person` attribute (for example, to change the current user context).
`getUsersWhoLike(String game) -> List<String>`	Finds and returns the names of users who like a specific game.

Example Usage

Creating a Social Network

Step 1: Initialize the Social Network

- Create a `SocialNetwork` object.
- Optionally, set a specific `Person` as the primary user (for example, a currently logged-in user).
- Add multiple users to the network.

Step 2: Manage Users

- Use `addUser()` to add new users with their favorite games.
- Use `removeUser()` to remove users from the network.
- Use `getUser()` to retrieve a user's details by their name.

Step 3: Retrieve Data Based on Games

- Use `getUsersWhoLike()` to find which users favor a specific game.

```
# Create the social network
network = SocialNetwork()

# Add users
user1 = Person("John", ["The Movie: The Game", "The Legend of Corgi"])
user2 = Person("Alice", ["Dinosaur Diner", "The Movie: The Game"])
user3 = Person("Bob", ["The Legend of Corgi", "Dinosaur Diner"])

network.addUser(user1)
network.addUser(user2)
network.addUser(user3)

# Optionally, update the current user
network.updatePerson(user1)

# Retrieve a specific user by name
retrievedUser = network.getUser("Alice")

# Find who likes a specific game
usersWhoLikeGame = network.getUsersWhoLike("Dinosaur Diner")  # Expected output: ["A
```

Optional Enhancements (Advanced Features)

- **Friend Recommendations** – Suggest friends based on common game interests.
- **Game Recommendations** – Suggest games based on users' favorite games.
- **Save & Load Network Data** – Store and retrieve network data using a database or file system.
- **Graph Visualization** – Display the network as a graph using visualization tools.

Summary

Feature	Details
Classes	`Person`, `SocialNetwork`
Data Structures	`Person` object and `List<Person>` for users
Key Methods	Add/Remove User, Update Person, Retrieve User, Get Users Who Like a Specific Game