Scenario:

You are tasked with building a **Subscription Management System** for a streaming service company. The company offers different types of subscriptions to users, each with varying pricing structures, features, and benefits.

Currently, the company offers the following subscription plans:

- **Basic Subscription**: Offers access to standard-definition (SD) content with ads.
- **Premium Subscription**: Provides high-definition (HD) content with no ads, and allows access to exclusive features such as offline downloads.
- **Family Subscription**: Offers everything in the Premium Subscription but allows multiple users under the same subscription account.

The company plans to expand the subscription offerings in the future (e.g., **Enterprise Subscription** for businesses, **Student Subscription** for discounted rates), so they need a system that can handle these new types of subscriptions **without modifying the core application logic**.

Your task is to create a Windows Forms application that allows customers to subscribe to one of the available plans, displays relevant plan details, and calculates the total subscription cost based on the selected plan and additional user inputs such as the **number of users** and **subscription period (in months)**.

Question:

Part 1: Designing the Subscription Selection Form

1. **Task**: Create a Windows Forms application with the following features:
    - A form where the user can input details for their subscription:
        - **Subscription Plan**: A dropdown list containing Basic, Premium, and Family subscriptions.
        - **Number of Users**: (For Family Subscription only) Input box to specify the number of users sharing the subscription.
        - **Subscription Period (in months)**: Input box to specify how many months the subscription is for.

    When the user submits the form, the system should process the information, display the **subscription details** (plan type, number of users, period), and calculate the **total cost**.

Part 2: Subscription Plan Cost Calculation

2. **Task**: Implement the logic to calculate the cost for each subscription plan. The company has the following pricing structure:
    - **Basic Subscription**: $10 per month.
    - **Premium Subscription**: $20 per month.

- **Family Subscription**: $30 per month for the first user, and $5 for each additional user.

The cost calculation should be dynamic based on the user's input (number of users and months), and the system should ensure that Family Subscription charges based on the number of users only when the Family plan is selected.

Part 3: Handling Plan-Specific Features

3. **Task**: Extend the application so that it displays **plan-specific features** when the user selects a subscription plan. Each plan has different features:
    - **Basic Subscription**: Displays "Access to SD content" and "Ads included".
    - **Premium Subscription**: Displays "Access to HD content", "No Ads", and "Offline downloads".
    - **Family Subscription**: Displays "Access to HD content", "No Ads", "Offline downloads", and "Multiple users allowed".

    Ensure that the appropriate features for each plan are displayed when the user selects a subscription.

Part 4: Refactoring for Flexibility (Advanced)

4. **Task**: Refactor the subscription handling logic to make the system flexible and **open to new subscription plans** without modifying the existing code significantly. The company plans to introduce new subscription types in the future, such as:
    - **Enterprise Subscription**: Offers all Premium features, plus API access and custom content.
    - **Student Subscription**: A discounted version of the Premium plan for students, available for $15 per month.

    Your system should be designed in such a way that adding these new subscription types can be done by creating new classes **without modifying the main form logic**. You need to allow for these new subscription types while maintaining a clean and maintainable codebase.

    Hint: Think about how to **encapsulate object creation** and use **polymorphism** to handle plan-specific behaviors.