**📝 Exercise 1: Interface - Music Player**

**💡 Problem Statement:**

Create an interface to **control music playback**.

**Instructions:**

1. Create an interface **IMusicPlayer** with methods:
   * **Play(string song)**
   * **Pause()**
   * **Stop()**
2. Implement the interface in two classes:
   * **SpotifyPlayer**:
     + Prints **"Playing [song] on Spotify"**
     + Prints **"Spotify playback paused"**
     + Prints **"Spotify playback stopped"**
   * **AppleMusicPlayer**:
     + Prints **"Playing [song] on Apple Music"**
     + Prints **"Apple Music playback paused"**
     + Prints **"Apple Music playback stopped"**
3. In the **Main()** method:
   * Create objects of both classes using the **IMusicPlayer** reference.
   * Call all methods to demonstrate functionality.

**📝 Exercise 2: Interface - Banking Operations**

**💡 Problem Statement:**

Create an interface to perform **basic banking operations**.

**Instructions:**

1. Create an interface **IBankAccount** with methods:
   * **Deposit(double amount)**
   * **Withdraw(double amount)**
   * **CheckBalance()**
2. Implement the interface in two classes:
   * **SavingsAccount**: Implements all methods with basic print statements
   * **CurrentAccount**: Implements all methods with basic print statements
3. In the **Main()** method:
   * Create objects of **SavingsAccount** and **CurrentAccount** using the **IBankAccount** reference.
   * Perform **deposit**, **withdrawal**, and **balance check** for both accounts.

**📝 Exercise 3: Abstract Class - Ticket Booking System**

**💡 Problem Statement:**

Create an abstract class to handle **ticket booking** for different modes of transport.

**Instructions:**

1. Create an abstract class **TicketBooking** with:
   * An **abstract method**: **BookTicket(int seats)**
   * A **concrete method**: **ShowBookingInfo()** to print **"Booking tickets"**
2. Create derived classes:
   * **BusBooking**: Implements **BookTicket()** to print **"Booking x bus tickets"**
   * **FlightBooking**: Implements **BookTicket()** to print **"Booking x flight tickets"**
3. In the **Main()** method:
   * Create objects of **BusBooking** and **FlightBooking** using the **TicketBooking** reference.
   * Call **ShowBookingInfo()** and **BookTicket()** for both objects.

**📝 Exercise 4: Abstract Class - Food Ordering System**

**💡 Problem Statement:**

Create an abstract class to manage **food ordering** from different restaurants.

**Instructions:**

1. Create an abstract class **FoodOrder** with:
   * An **abstract method**: **PlaceOrder(string item)**
   * A **concrete method**: **OrderInfo()** to print **"Placing food order"**
2. Create derived classes:
   * **FastFoodOrder**: Implements **PlaceOrder()** to print **"Order placed for [item] at Fast Food Outlet"**
   * **FineDiningOrder**: Implements **PlaceOrder()** to print **"Order placed for [item] at Fine Dining Restaurant"**
3. In the **Main()** method:
   * Create objects of **FastFoodOrder** and **FineDiningOrder** using the **FoodOrder** reference.
   * Call **OrderInfo()** and **PlaceOrder()** for both objects.