Game Catalog and Purchase

1.Single Responsibility Principle :

- The **Game** class includes the properties of a game (title, genre, and price).
- **MutiPlayerGame** and **SinglePlayerGame** subclasses extend Game class for specialized game types.
- **GameCatalog** interface handles the responsibility of displaying the catalog and purchasing a game.
- The **PaymentMethod** interface focuses on the responsibility of processing payments.
- Each class and interface has a clear, single responsibility.

```
package com.ilp.entity;
//Single Responsibility Principle,//Open/Closed Principle
public class Game {
  private String title;
  private String genre;
  private double price;
  public Game(String title, String genre, double price) {
    this.title = title;
    this.genre = genre;
    this.price = price;
  }
  public String getTitle() {
    return title;
  }
  public String getGenre() {
    return genre;
  }
  public double getPrice() {
    return price;
  }
}
package com.ilp.interfaces;
public interface GameCatalog {
  void displayCatalog();
}

package com.ilp.interfaces;
public interface PaymentMethod {
  void processPayment(double amount);
}
```

2. Open/Closed Principle :

- The **Game** class is closed for modification but open for extension, demonstrated by the creation of **MutiPlayerGame** and **SinglePlayerGame** subclasses.
- Allowing extension without modifying existing code.
- The **PaymentBill** class implementing the **GeneratedBill** interface is closed for modification, but new billing features can be added by creating new classes that implement the **GeneratedBill** interface.
- The **PaymentBill** class is open for extension without altering its existing code.

```
package com.ilp.interfaces;
public interface GeneratedBill {
  void paymentMessage();
        void generateBill(double amount);
}
package com.ilp.services;
import com.ilp.interfaces.GeneratedBill;
//Open/Closed Principle
public class PaymentBill extends PaymentDone implements GeneratedBill {
        public PaymentBill() {
              paymentMessage();
           }
        public void generateBill(double amount) {
                System.out.println("Generated Bill of amount:"+amount);
        }
}
package com.ilp.services;
public class PaymentDone{
  public void paymentMessage() {
            System.out.println("Thank you for the purchase!");
  }
}
```

3. Liskov's Substitution Principle:

- **MutiPlayerGame** and **SinglePlayerGame** are substitutable for instances of the base class **Game**.

```
package com.ilp.entity;
//Liskov's Substitution Principle
public class MultiPlayerGame extends Game {
     private int maxPlayers;
     public MultiPlayerGame(String title, String genre, double price, int maxPlayers) {
        super(title, genre, price);
        this.maxPlayers = maxPlayers;
```

```
    }
    public int getMaxPlayers() {
        return maxPlayers;
    }
}
```

```
package com.ilp.entity;
//Liskov's Substitution Principle
public class SinglePlayerGame extends Game {
    public SinglePlayerGame(String title, String genre, double price) {
        super(title, genre, price);
    }
}
```

4. Interface Segregation Principle :

- The **PaymentMethod** interface includes only methods necessary for payment processing.
- The code provides a specific interface for payment.

```
package com.ilp.interfaces;
//Interface Segregation Principle
public interface PaymentMethod {
    void processPayment(double amount);
}
```

5. Dependency Inversion Principle:

- The **CatalogServices** class depends on abstractions **(GameCatalog** and **PaymentMethod)** rather than concrete implementations.
- **CreditCardPayment** and **PayPalPayment** implement the **PaymentMethod** interface.
- The code relies on abstractions and injecting dependencies.

```
package com.ilp.interfaces;
//Interface segregation Principle
public interface GameCatalog {
    void displayCatalog();
}
```

```java
package com.ilp.interfaces;
//Interface Segregation Principle
public interface PaymentMethod {
  void processPayment(double amount);
}
```

```java
package com.ilp.services;
import com.ilp.entity.Game;
import com.ilp.entity.MultiPlayerGame;
import com.ilp.entity.SinglePlayerGame;
import com.ilp.interfaces.PaymentMethod;
import com.ilp.interfaces.GameCatalog;
import com.ilp.interfaces.GamePurchase;
import java.util.ArrayList;
import java.util.List;
//Dependency Inversion Principle
public class CatalogServices implements GameCatalog,GamePurchase {
    private List<Game> purchasedGames;
    public CatalogServices() {
        this.purchasedGames = new ArrayList<>();
    }
    public void displayCatalog() {
        System.out.println("Catalog:");
        for (Game game : getAvailableGames()) {
            System.out.println("Game: "+game.getTitle());
            System.out.println("Price: Rs." + game.getPrice());
            System.out.println("Genre: "+game.getGenre());
            if (game instanceof MultiPlayerGame) {
                int maxPlayers = ((MultiPlayerGame) game).getMaxPlayers();
                System.out.println("Max Players: " + maxPlayers);
            }
            System.out.println();
        }
    }
    @Override
    public void purchaseGame(Game game, PaymentMethod paymentMethod) {
        paymentMethod.processPayment(game.getPrice());
        PaymentBill paymentBill = new PaymentBill();
        paymentBill.generateBill(game.getPrice());

        purchasedGames.add(game);
        System.out.println("Game purchased: " + game.getTitle());
        paymentBill.paymentMessage();
    }
```

```
        private List<Game> getAvailableGames() {
            List<Game> availableGames = new ArrayList<>();
            availableGames.add(new SinglePlayerGame("GhostRunner", "Action", 2200.00));
            availableGames.add(new MultiPlayerGame("Sekiro", "Adventure",2500.00,2));

            return availableGames;
        }
    }
```

```
package com.ilp.services;
import com.ilp.interfaces.PaymentMethod;
public class CreditCardPayment implements PaymentMethod {
  public void processPayment(double amount) {
    System.out.println("Processing credit card payment: Rs." + amount);
  }
}
package com.ilp.services;
import com.ilp.interfaces.PaymentMethod;
public class PayPalPayment implements PaymentMethod {
  public void processPayment(double amount) {
    System.out.println("Processing PayPal payment: Rs." + amount);
  }
}
```