Task1

Java不支持类的多继承——一个类不能同时继承多个父类，是为了“简单性” 和 “可靠性”：

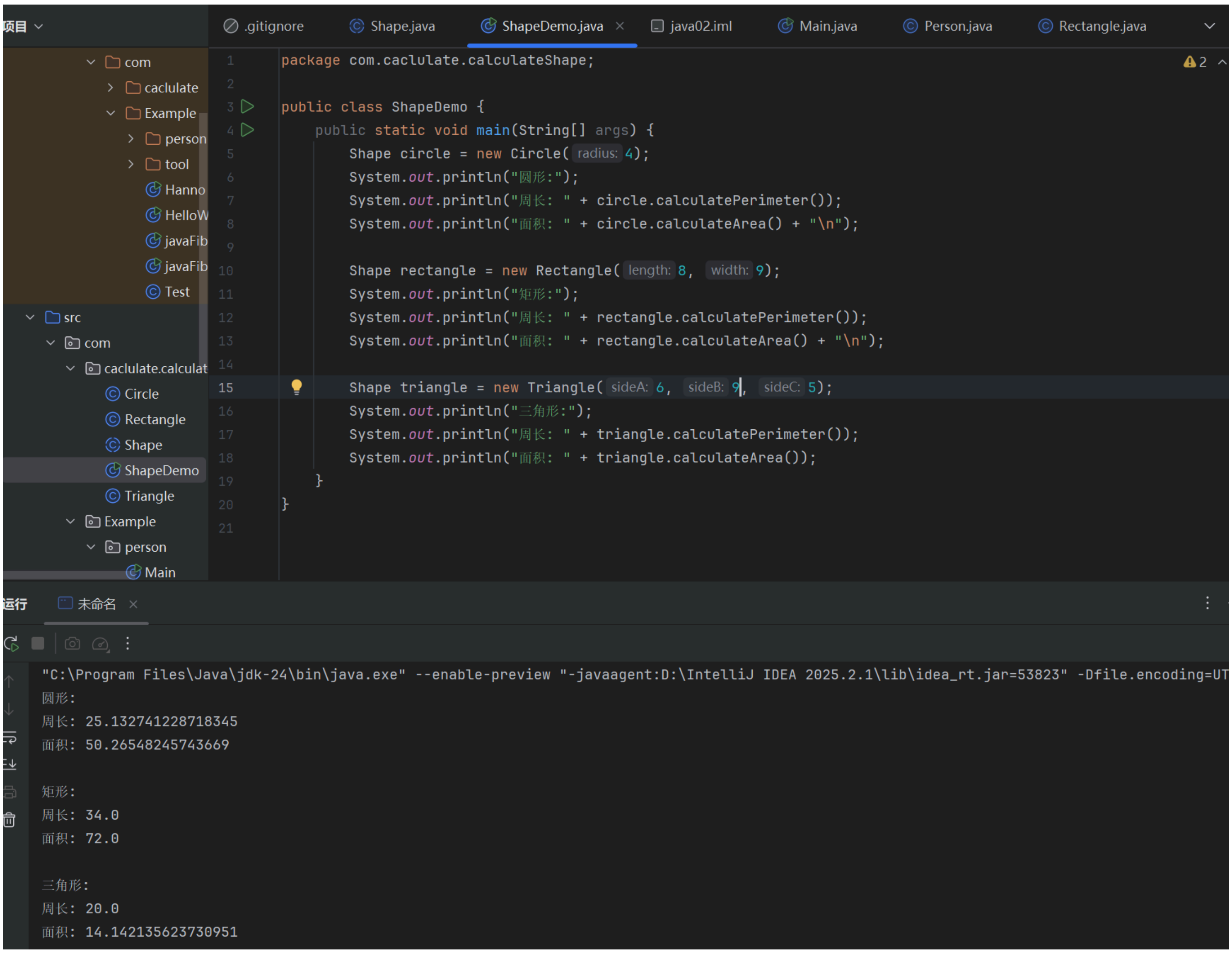
1.为了避免**“菱形继承冲突”**，当子类D试图继承都重写了A的两个父类B类C类，JVM无法判断该执行b的还是c的逻辑。

2.为了避免**“继承链混乱”**，一个类可能有多个直接父类，每个父类又有自己的父类，导致继承链追踪困难；若多个父类定义了同名的成员变量（如 B 和 C 都有int count），子类 D 访问 `count` 时会出现 “变量名冲突”，需要额外语法区分（如B.this.count），增加代码冗余，维护困难

3.防止**功能冗余与权限冲突**，若 B 和 C 都提供了同一方法，D 继承后会同时拥有两个重复的方法，造成 “功能冗余”，且无法确定优先使用哪个；若 B 的访问修饰符是public，C 的访问修饰符是private，子类 D 继承时会出现 “权限不一致”，JVM 无法判断该方法的访问权限

Task2

测试入口



```java
package com.caclulate.calculateShape;

public class ShapeDemo {
    public static void main(String[] args) {
        Shape circle = new Circle( radius: 4);
        System.out.println("圆形:");
        System.out.println("周长: " + circle.calculatePerimeter());
        System.out.println("面积: " + circle.calculateArea() + "\n");

        Shape rectangle = new Rectangle( length: 8,  width: 9);
        System.out.println("矩形:");
        System.out.println("周长: " + rectangle.calculatePerimeter());
        System.out.println("面积: " + rectangle.calculateArea() + "\n");

        Shape triangle = new Triangle( sideA: 6,  sideB: 9,  sideC: 5);
        System.out.println("三角形:");
        System.out.println("周长: " + triangle.calculatePerimeter());
        System.out.println("面积: " + triangle.calculateArea());
    }
}
```

```
"C:\Program Files\Java\jdk-24\bin\java.exe" --enable-preview "-javaagent:D:\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=53823" -Dfile.encoding=UT
圆形:
周长: 25.132741228718345
面积: 50.26548245743669

矩形:
周长: 34.0
面积: 72.0

三角形:
周长: 20.0
面积: 14.142135623730951
```

圆形的@Override

```java
package com.caclulate.calculateShape;

public class Circle implements Shape {
    private double radius;

    public Circle(double radius) {
        if (radius <= 0) {
            throw new IllegalArgumentException("半径必须大于0");
        }
        this.radius = radius;
    }

    @Override
    public double calculatePerimeter() {
        return 2 * Math.PI * radius;
    }

    @Override
    public double calculateArea() {
        return Math.PI * radius * radius;
    }

    public double getRadius() {
        return radius;
    }

    public void setRadius(double radius) {
        if (radius <= 0) {
            throw new IllegalArgumentException("半径必须大于0");
        }
        this.radius = radius;
    }
}
```

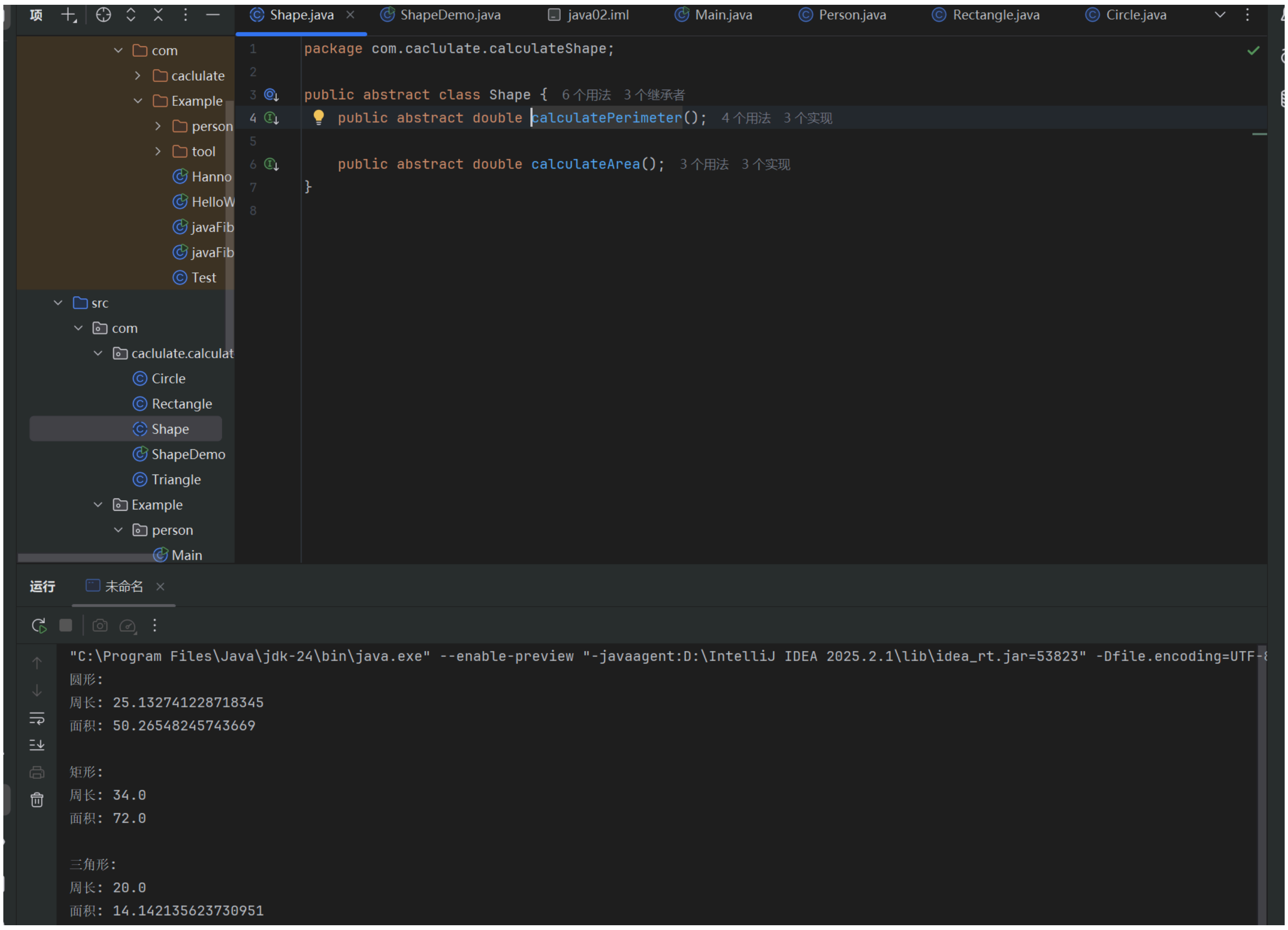
矩形的@Override

```
package com.caclulate.calculateShape;

public class Rectangle implements Shape {
    private double length;
    private double width;

    public Rectangle(double length, double width) {
        if (length <= 0 || width <= 0) {
            throw new IllegalArgumentException("长和宽必须大于0");
        }
        this.length = length;
        this.width = width;
    }

    @Override
    public double calculatePerimeter() {
        return 2 * (length + width);
    }

    @Override
    public double calculateArea() {
        return length * width;
    }
    public double getLength() {
        return length;
    }

    public void setLength(double length) {
        if (length <= 0) {
            throw new IllegalArgumentException("长度必须大于0");
        }
        this.length = length;
    }

    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        if (width <= 0) {
            throw new IllegalArgumentException("宽度必须大于0");
        }
        this.width = width;
    }
}
```

三角形的@Override

```
package com.caclulate.calculateShape;

public class Triangle implements Shape {
    private double sideA;
    private double sideB;
    private double sideC;

    public Triangle(double sideA, double sideB, double sideC) {
        if (!isValidTriangle(sideA, sideB, sideC)) {
            throw new IllegalArgumentException("这三条边不能构成三角形");
        }
        this.sideA = sideA;
        this.sideB = sideB;
        this.sideC = sideC;
    }

    private boolean isValidTriangle(double a, double b, double c) {
        return a > 0 && b > 0 && c > 0 &&
               a + b > c && a + c > b && b + c > a;
    }

    @Override
    public double calculatePerimeter() {
        return sideA + sideB + sideC;
    }
```

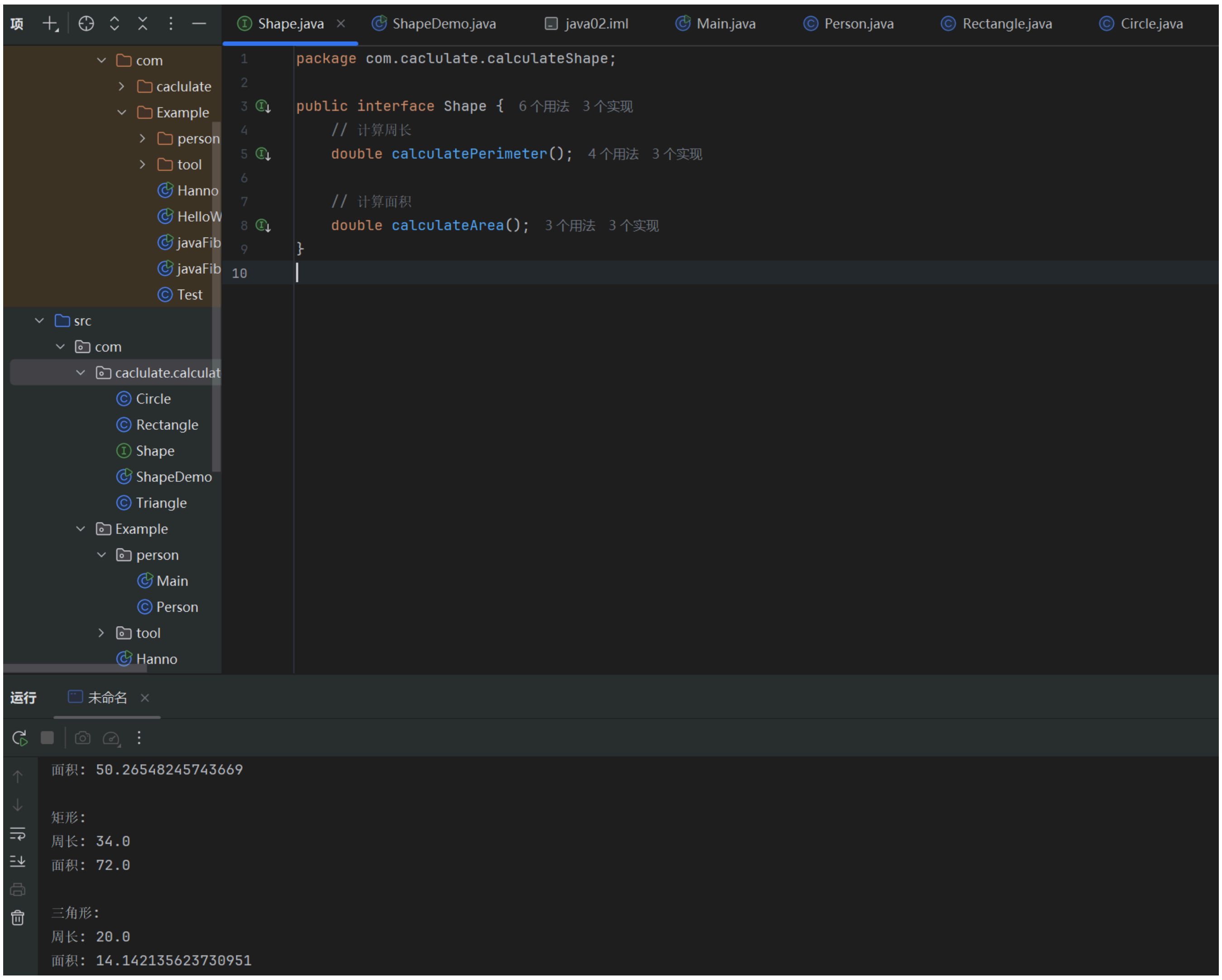
```java
    @Override
    public double calculateArea() {
        double s = calculatePerimeter() / 2; // 半周长
        return Math.sqrt(s * (s - sideA) * (s - sideB) * (s - sideC));
    }

    public double getSideA() {
        return sideA;
    }

    public void setSideA(double sideA) {
        if (!isValidTriangle(sideA, sideB, sideC)) {
            throw new IllegalArgumentException("这三条边不能构成三角形");
        }
        this.sideA = sideA;
    }

    public double getSideB() {
        return sideB;
    }

    public void setSideB(double sideB) {
        if (!isValidTriangle(sideA, sideB, sideC)) {
            throw new IllegalArgumentException("这三条边不能构成三角形");
        }
        this.sideB = sideB;
    }

    public double getSideC() {
        return sideC;
    }

    public void setSideC(double sideC) {
        if (!isValidTriangle(sideA, sideB, sideC)) {
            throw new IllegalArgumentException("这三条边不能构成三角形");
        }
        this.sideC = sideC;
    }
}
```



```java
package com.caclulate.calculateShape;

public abstract class Shape {  6 个用法  3 个继承者
    public abstract double calculatePerimeter();  4 个用法  3 个实现

    public abstract double calculateArea();  3 个用法  3 个实现
}
```

```
"C:\Program Files\Java\jdk-24\bin\java.exe" --enable-preview "-javaagent:D:\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=53823" -Dfile.encoding=UTF-
圆形:
周长: 25.132741228718345
面积: 50.26548245743669

矩形:
周长: 34.0
面积: 72.0

三角形:
周长: 20.0
面积: 14.142135623730951
```

插口版



```java
package com.caclulate.calculateShape;

public interface Shape {
    // 计算周长
    double calculatePerimeter();

    // 计算面积
    double calculateArea();
}
```

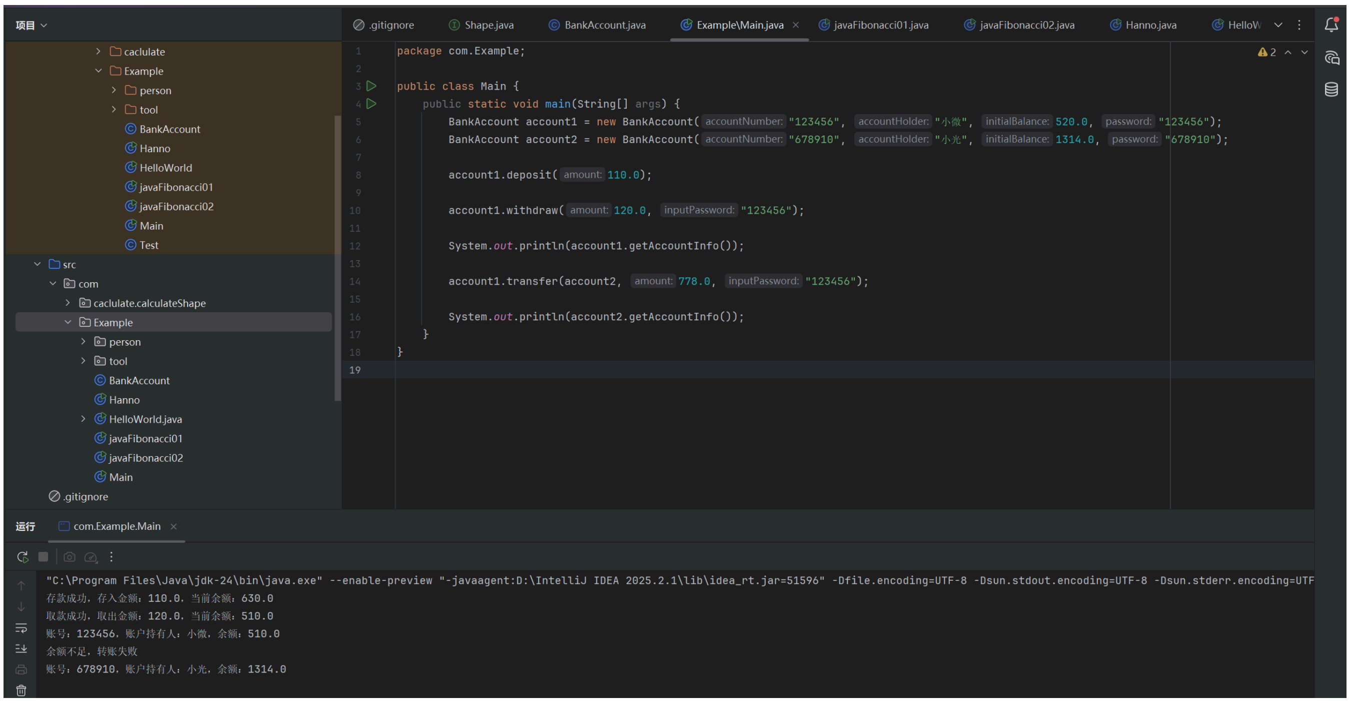
```
面积: 50.26548245743669

矩形:
周长: 34.0
面积: 72.0

三角形:
周长: 20.0
面积: 14.142135623730951
```

Task3

BankAccount类

```java
package com.Example;

public class BankAccount {
    private String accountNumber;
    private String accountHolder;
    private double balance;
    private String password; // 敏感信息，需要严格保护

    public BankAccount(String accountNumber, String accountHolder, double initialBalance,
String password) {
        //TODO
        this.accountNumber = accountNumber;
        this.accountHolder = accountHolder;
        this.balance = initialBalance;
        this.password = password;
    }

    public double getBalance() {
        return balance;
    }
    void deposit(double amount) {
        //TODO
        this.balance += amount;
        System.out.println("存款成功，存入金额：" + amount + "，当前余额：" + this.balance);

    }

    boolean withdraw(double amount, String inputPassword) {
        //TODO
        if (!this.password.equals(inputPassword)) {
            System.out.println("密码错误，取款失败");
            return false;
        }
        if (amount > this.balance) {
            System.out.println("余额不足，取款失败");
            return false;
        }
        this.balance -= amount;
        System.out.println("取款成功，取出金额：" + amount + "，当前余额：" + this.balance);
        return true;
    }

    boolean transfer(BankAccount recipient, double amount, String inputPassword) {
        //TODO
        if (!this.password.equals(inputPassword)) {
        System.out.println("密码错误，转账失败");
        }
        if (amount > this.balance) {
            System.out.println("余额不足，转账失败");
            return false;
        }
        this.balance -= amount;
        System.out.println("转账成功，转出金额：" + amount + "，当前余额：" + this.balance);
        return false;
    }

    String getAccountInfo() {
        //TODO
        return "账号：" + accountNumber + "，账户持有人：" + accountHolder + "，余额：" +
balance;
    }
    // 只需修改可见性
    private boolean validatePassword(String inputPassword) {
        return true;
    }
    // 只需修改可见性
    private boolean validateAmount(double amount) {
        return true;
    }



}
```

Main类



```java
package com.Example;

public class Main {
    public static void main(String[] args) {
        BankAccount account1 = new BankAccount( accountNumber: "123456",  accountHolder: "小微",  initialBalance: 520.0,  password: "123456");
        BankAccount account2 = new BankAccount( accountNumber: "678910",  accountHolder: "小光",  initialBalance: 1314.0,  password: "678910");

        account1.deposit( amount: 110.0);

        account1.withdraw( amount: 120.0,  inputPassword: "123456");

        System.out.println(account1.getAccountInfo());

        account1.transfer(account2,  amount: 778.0,  inputPassword: "123456");

        System.out.println(account2.getAccountInfo());
    }
}
```

```
"C:\Program Files\Java\jdk-24\bin\java.exe" --enable-preview "-javaagent:D:\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=51596" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF
存款成功，存入金额：110.0，当前余额：630.0
取款成功，取出金额：120.0，当前余额：510.0
账号：123456，账户持有人：小微，余额：510.0
余额不足，转账失败
账号：678910，账户持有人：小光，余额：1314.0
```