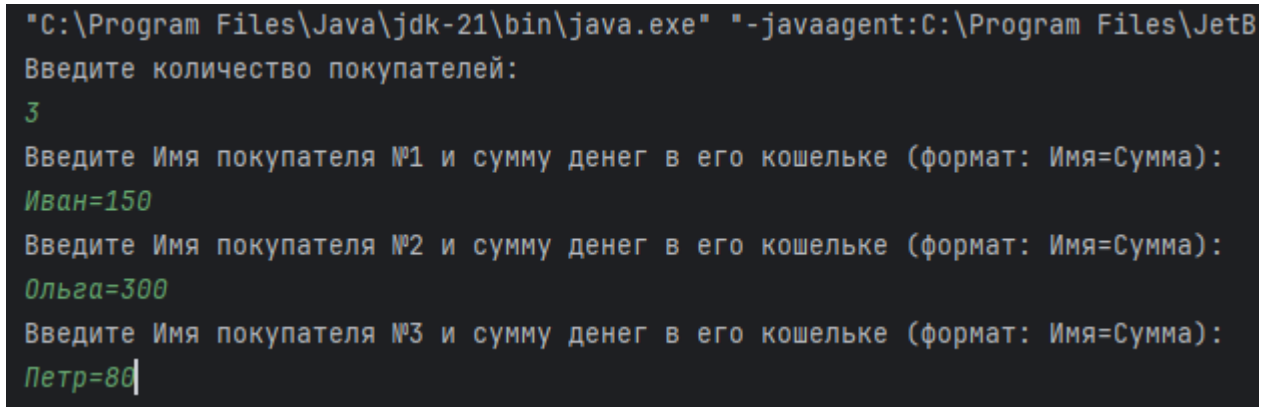
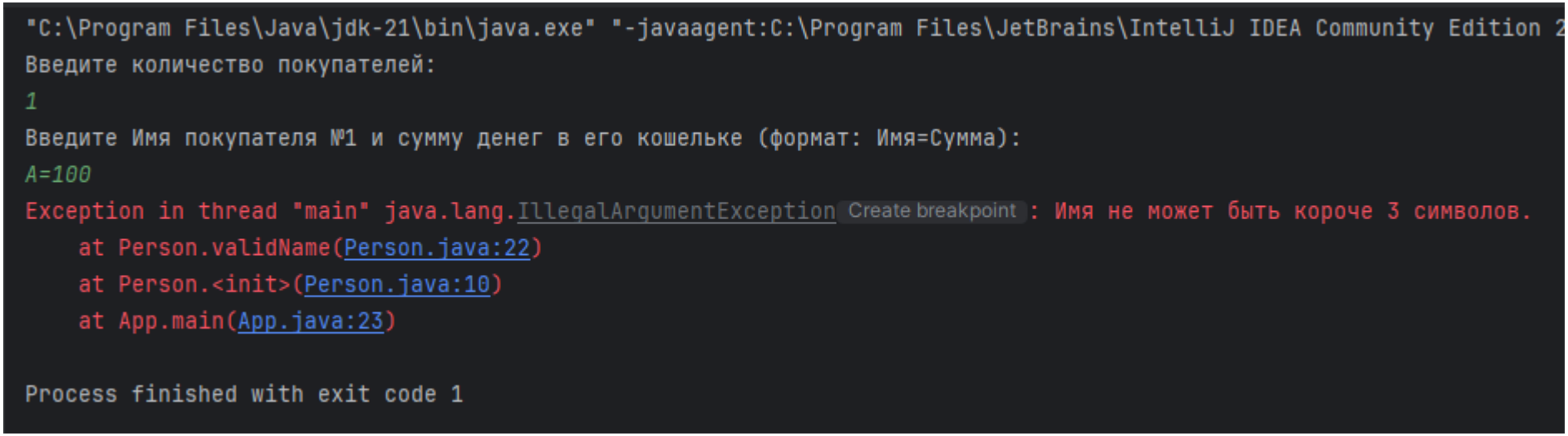
Тестовые данные для ДЗ - Выполнение программы в нормальном состоянии

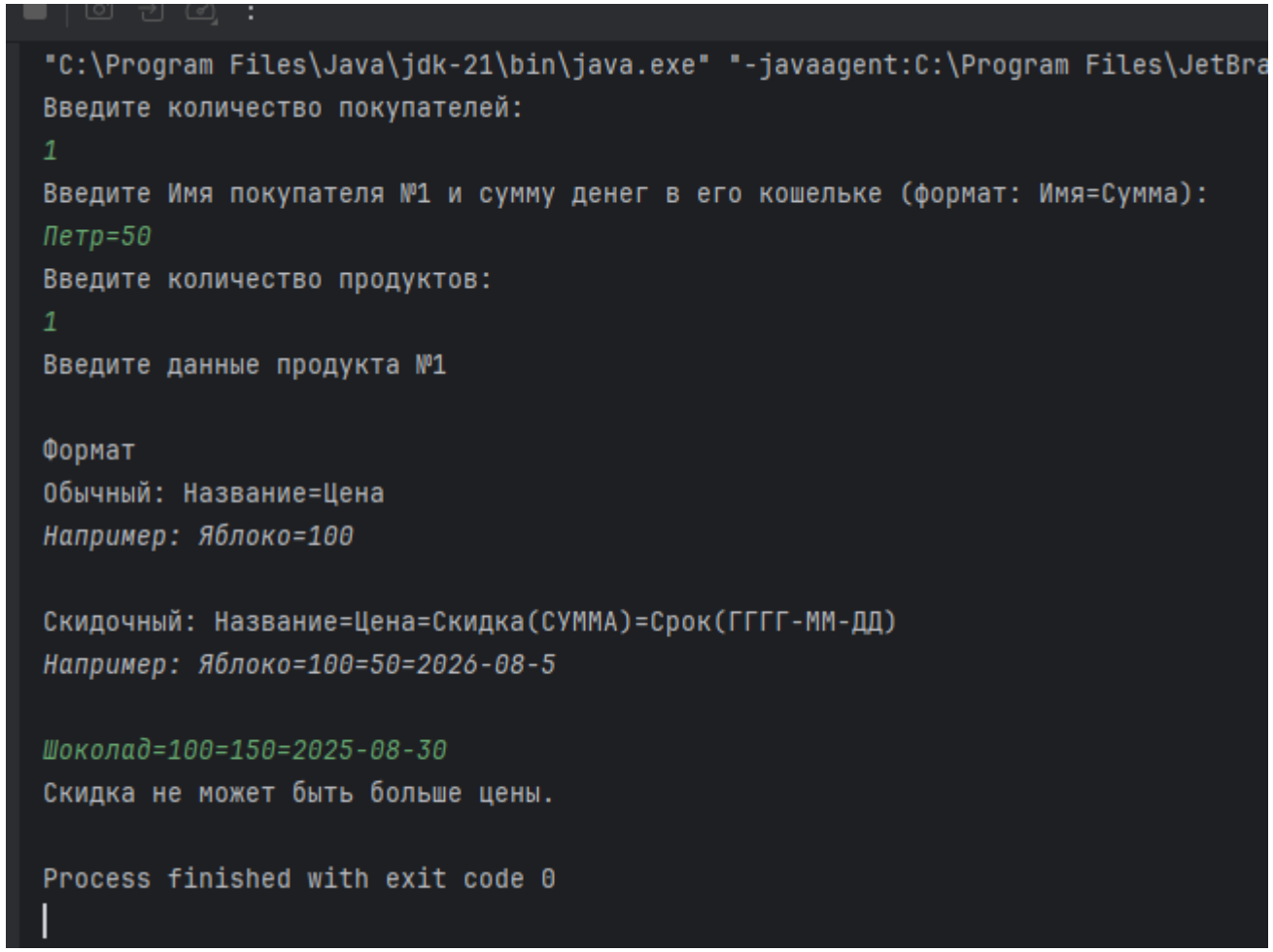
Ввод количество покупателей:	3
Имена покупателей:	Иван=150
	Ольга=300
	Петр=80
Ввод количество продуктов:	4
Ввод данных продукта:	Хлеб=50
	Молоко=80
	Шоколад=200=50=2026-08-17
	Яблоки=120=30=2024-08-17
Доступные продукты:	Хлеб
	Яблоки
	Шоколад
	Молоко
END	


"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetB
Введите количество покупателей:
3
Введите Имя покупателя №1 и сумму денег в его кошельке (формат: Имя=Сумма):
Иван=150
Введите Имя покупателя №2 и сумму денег в его кошельке (формат: Имя=Сумма):
Ольга=300
Введите Имя покупателя №3 и сумму денег в его кошельке (формат: Имя=Сумма):
Петр=80

```
Доступные продукты: Хлеб, Молоко, Шоколад, Яблоки
Иван, введите название продукта для покупки (или END для завершения):
Хлеб

Иван купил Хлеб
У покупателя Иван осталось 100 ye.

Ольга, введите название продукта для покупки (или END для завершения):
Яблоки

Ольга купил Яблоки
У покупателя Ольга осталось 210 ye.

Петр, введите название продукта для покупки (или END для завершения):
Шоколад

Петр не может позволить себе Шоколад

У покупателя Петр осталось 80 ye.

Иван, введите название продукта для покупки (или END для завершения):
Молоко

Иван купил Молоко
У покупателя Иван осталось 20 ye.

Ольга, введите название продукта для покупки (или END для завершения):
Апельсины
Продукт отсутствует в списке заявленных.
Петр, введите название продукта для покупки (или END для завершения):
END

Иван - Хлеб, Молоко
Ольга - Яблоки
Петр - Ничего не куплено

Process finished with exit code 0
```

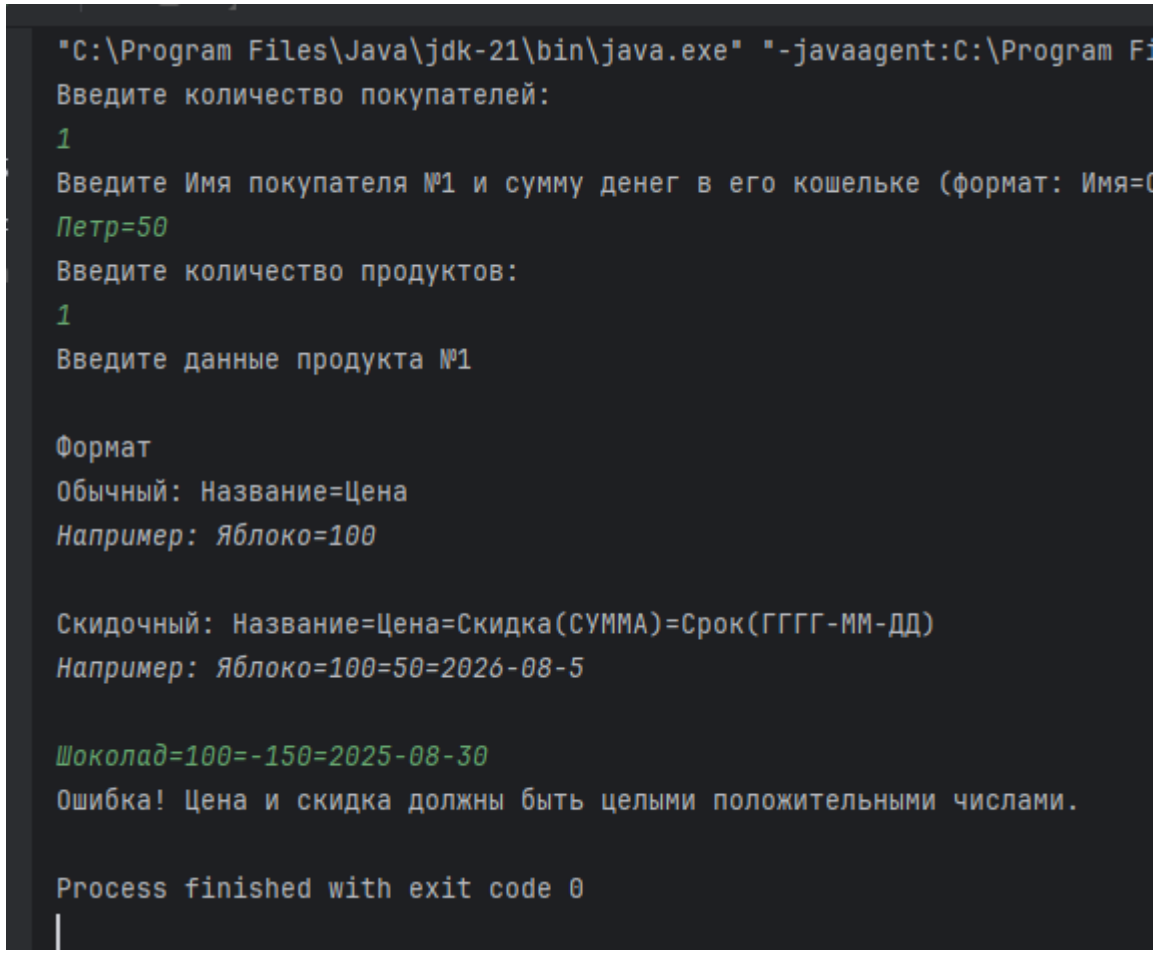
Имя не может быть короче 3 символов	A=100
Деньги не могут быть отрицательными	Борис=-10
Имя продукта не может быть пустым	=50
Скидка не может быть больше цены	Шоколад=100=150=2025-08-30
Цена и скидка должны быть целыми положительными числами	Шоколад=100=-150=2025-08-30
Если стоимость продукта или скидочного продукта 0 или отрицательная, то такая цена неправильна	Шоколад=100=0=2025-08-30

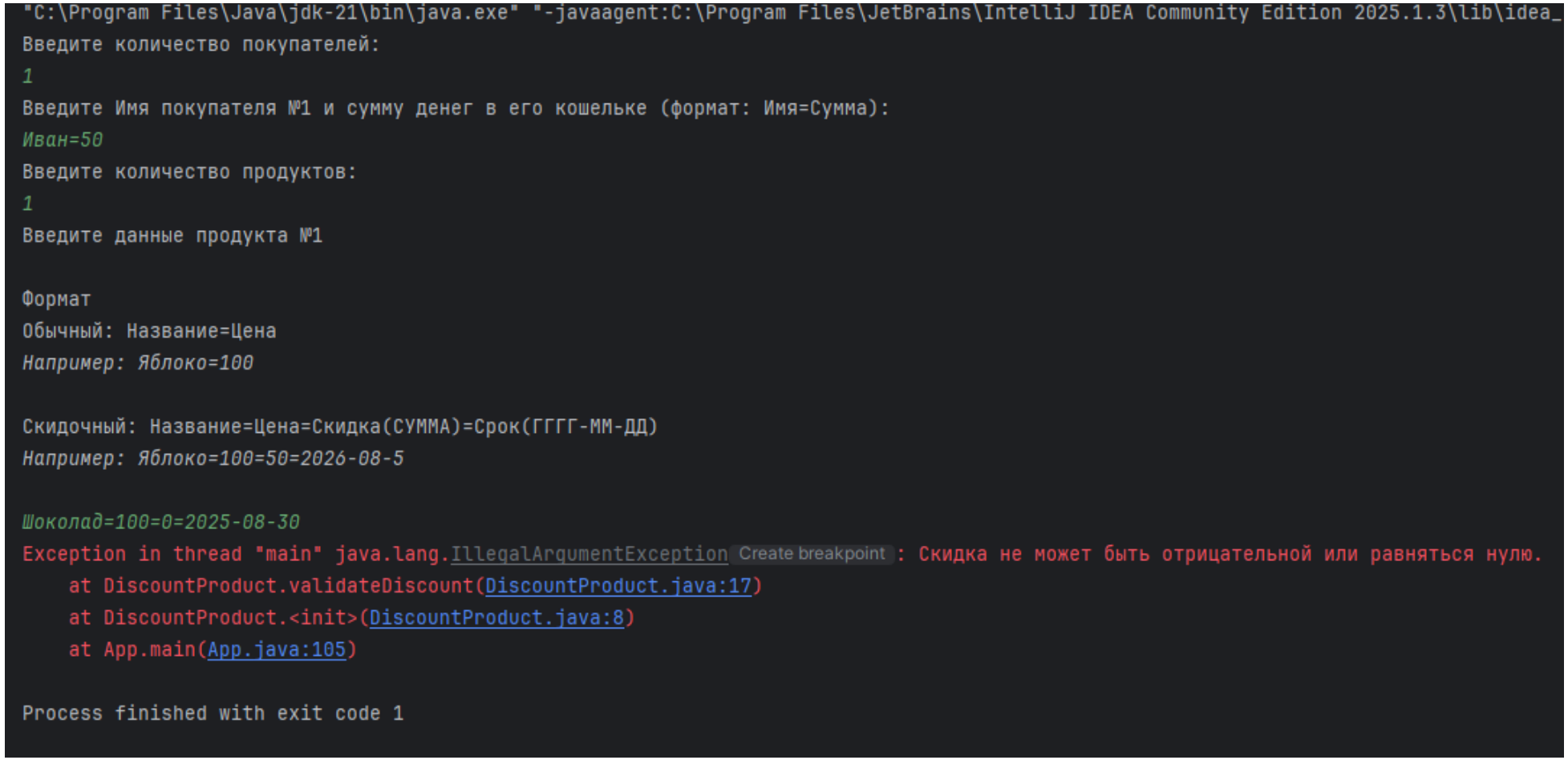



"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBra
Введите количество покупателей:
1
Введите Имя покупателя №1 и сумму денег в его кошельке (формат: Имя=Сумма):
Петр=50
Введите количество продуктов:
1
Введите данные продукта №1
Формат
Обычный: Название=Цена
Например: Яблоко=100
Скидочный: Название=Цена=Скидка(СУММА)=Срок(ГГГГ-ММ-ДД)
Например: Яблоко=100=50=2026-08-5
Шоколад=100=150=2025-08-30
Скидка не может быть больше цены.
Process finished with exit code 0

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2
Введите количество покупателей:
1
Введите Имя покупателя №1 и сумму денег в его кошельке (формат: Имя=Сумма):
Борис=-10
Exception in thread "main" java.lang.IllegalArgumentException Create breakpoint : Деньги не могут быть отрицательными.
    at Person.validMoney(Person.java:28)
    at Person.<init>(Person.java:11)
    at App.main(App.java:23)

Process finished with exit code 1
```

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Commun
Введите количество покупателей:
1
Введите Имя покупателя №1 и сумму денег в его кошельке (формат: Имя=Сумма):
=50
Exception in thread "main" java.lang.IllegalArgumentException Create breakpoint : Имя не может быть пустым.
    at Person.validName(Person.java:19)
    at Person.<init>(Person.java:10)
    at App.main(App.java:23)

Process finished with exit code 1
```


"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Fi
Введите количество покупателей:
1
Введите Имя покупателя №1 и сумму денег в его кошельке (формат: Имя=
Петр=50
Введите количество продуктов:
1
Введите данные продукта №1
Формат
Обычный: Название=Цена
Например: Яблоко=100
Скидочный: Название=Цена=Скидка(СУММА)=Срок(ГГГГ-ММ-ДД)
Например: Яблоко=100=50=2026-08-5
Шоколад=100=-150=2025-08-30
Ошибка! Цена и скидка должны быть целыми положительными числами.
Process finished with exit code 0


"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1.3\lib\idea_
Введите количество покупателей:
1
Введите Имя покупателя №1 и сумму денег в его кошельке (формат: Имя=Сумма):
Иван=50
Введите количество продуктов:
1
Введите данные продукта №1
Формат
Обычный: Название=Цена
Например: Яблоко=100
Скидочный: Название=Цена=Скидка(СУММА)=Срок(ГГГГ-ММ-ДД)
Например: Яблоко=100=50=2026-08-5
Шоколад=100=0=2025-08-30
Exception in thread "main" java.lang.IllegalArgumentException Create breakpoint : Скидка не может быть отрицательной или равняться нулю.
at DiscountProduct.validateDiscount(DiscountProduct.java:17)
at DiscountProduct.<init>(DiscountProduct.java:8)
at App.main(App.java:105)
Process finished with exit code 1

App.java

```java
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class App {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        List<Person> persons = new ArrayList<>();
        List<Product> products = new ArrayList<>();

        System.out.println("Введите количество покупателей: ");
        int peopleAmount = Integer.parseInt(scanner.nextLine().trim());

        for (int i = 0; i < peopleAmount; i++) {
            System.out.println("Введите Имя покупателя №" + (i + 1) + " и сумму денег в его кошельке (формат: Имя=Сумма):
");
            String[] peopleDB = scanner.nextLine().trim().split("=");
            if (peopleDB.length != 2) {
                System.out.println("Ошибка! Формат: Имя пользователя=Сумма (например: Борис=10)");
                i--;
                continue;
            } else {
                int money = Integer.parseInt(peopleDB[1].trim());
                Person person = new Person(peopleDB[0].trim(), money);
                persons.add(person);
            }
        }

        System.out.println("Введите количество продуктов:");
        int productAmount = Integer.parseInt(scanner.nextLine().trim());

        for (int i = 0; i < productAmount; i++) {
            System.out.println("Введите данные продукта №" + (i+1));
            System.out.println("\nФормат");
            System.out.println("Обычный: Название=Цена");
            System.out.println("\u001B[3mНапример: Яблоко=100\u001B[23m\n");
            System.out.println("Скидочный: Название=Цена=Скидка(СУММА)=Срок(ГГГГ-ММ-ДД)");
            System.out.println("\u001B[3mНапример: Яблоко=100=50=2026-08-5\u001B[23m\n");
            String[] productDB = scanner.nextLine().trim().split("=");
```

```
        if (productDB.length == 2) {
            String name = productDB[0].trim();
            String priceInput = productDB[1].trim();

            if (!priceInput.matches("\\d+")) {
                System.out.println("Ошибка! Цена должна быть целым положительным числом.");
                System.exit(0);
            }

            int price = Integer.parseInt(priceInput);

            if (name.isEmpty()) {
                System.out.println("Имя продукта не может быть пустым.");
                System.exit(0);
            } else if (name.length() < 3) {
                System.out.println("Имя продукта не может быть короче 3 символов.");
                System.exit(0);
            } else if (name.matches("\\d+")) {
                System.out.println("Имя продукта не может состоять только из цифр.");
                System.exit(0);
            } else if (price <= 0) {
                System.out.println("Цена продукта должна быть положительной.");
                System.exit(0);
            }

            Product product = new Product(name, price);
            products.add(product);

        } else if (productDB.length == 4) {
            String name = productDB[0].trim();
            String priceInput = productDB[1].trim();
            String discountInput = productDB[2].trim();
            String expiry = productDB[3].trim();

            if (!priceInput.matches("\\d+") || !discountInput.matches("\\d+")) {
                System.out.println("Ошибка! Цена и скидка должны быть целыми положительными числами.");
                System.exit(0);
            }

            int price = Integer.parseInt(priceInput);
            int discount = Integer.parseInt(discountInput);
```

```
            if (name.isEmpty()) {
                System.out.println("Имя продукта не может быть пустым.");
                System.exit(0);
            } else if (name.length() < 3) {
                System.out.println("Имя продукта не может быть короче 3 символов.");
                System.exit(0);
            } else if (name.matches("\\d+")) {
                System.out.println("Имя продукта не может состоять только из цифр.");
                System.exit(0);
            } else if (price <= 0) {
                System.out.println("Цена продукта должна быть положительной.");
                System.exit(0);
            } else if (discount < 0) {
                System.out.println("Скидка не может быть отрицательной.");
                System.exit(0);
            } else if (discount > price) {
                System.out.println("Скидка не может быть больше цены.");
                System.exit(0);
            } else if (expiry.isEmpty()) {
                System.out.println("Срок действия скидки не может быть пустым.");
                System.exit(0);
            }

            Product product = new DiscountProduct(name, price, discount, expiry);
            products.add(product);

        } else {
            System.out.println("Ошибка! Неверный формат. Допустимые форматы:\n"
                    + "Обычный продукт: Название=Цена\n"
                    + "Скидочный продукт: Название=Цена=Скидка=Срок");
            i--;
        }
    }
    System.out.print("Доступные продукты: ");
    for (int i = 0; i < products.size(); i++) {
        if (i > 0) System.out.print(", ");
        System.out.print(products.get(i).getName());
    }
    System.out.println();
    boolean buying = true;
```

```
        while (buying) {
            for (Person person : persons) {
                System.out.println(person.getName() + ", введите название продукта для покупки (или END для
завершения):");
                String input = scanner.nextLine().trim();

                if (input.equalsIgnoreCase("END")) {
                    buying = false;
                    break;
                }

                Product selected = null;
                for (Product NameProduct : products) {
                    if (NameProduct.getName().equalsIgnoreCase(input)) {
                        selected = NameProduct;
                        break;
                    }
                }

                if (selected != null) {
                    person.buy(selected);
                } else {
                    System.out.println("Продукт отсутствует в списке заявленных.");
                }
            }
        }

        System.out.println();

        for (Person person : persons) {
            if (person.getProducts().isEmpty()) {
                System.out.println(person.getName() + " - Ничего не куплено");
            } else {
                StringBuilder estr = new StringBuilder();
                for (Product product : person.getProducts()) {
                    if (estr.length() > 0) {
                        estr.append(", ");
                    }
                    estr.append(product.getName());
                }
                System.out.println(person.getName() + " - " + estr.toString());
```

```
            }
        }
        scanner.close();
    }
}
```

Product.java

```
public class Product {
    private String name;
    private int cost;

    public Product (String name, int cost) {
        if (name.trim().equals("")) {
            System.out.println("Имя не может быть пустым.");
            System.exit(0);
        } else if (name.trim().length() < 3) {
            System.out.println("Имя не может быть короче 3 символов.");
            System.exit(0);
        } else if (cost < 0) {
            System.out.println("Деньги не могут быть отрицательными.");
            System.exit(0);
        } else if (name.trim().matches("\\d+")) {
            System.out.println("Имя не должно содержать только цифры.");
            System.exit(0);
        } else {
            this.name = name;
            this.cost = cost;
        }
    }

    public String getName() {
        return name;
    }
```

```
    public int getCost() {
        return cost;
    }

    public String toString() {
        return name;
    }

    public int hashCode() {
        return name.hashCode() + cost;
    }

    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (!(obj instanceof Product)) return false;
        Product other = (Product) obj;
        return this.name.equals(other.name) && this.cost == other.cost;
    }

}
```

DiscountProduct.java

```
public class DiscountProduct extends Product {
    private final int discount;
    private final String expiryDate;

    public DiscountProduct(String name, int cost, int discount, String expiryDate) {
        super(name, cost);

        validateDiscount(cost, discount);
        validateExpiry(expiryDate);

        this.discount = discount;
        this.expiryDate = expiryDate;
    }
```

```
    private void validateDiscount(int cost, int discount) {
        if (discount <= 0) {
            throw new IllegalArgumentException("Скидка не может быть отрицательной или равняться нулю.");
        }
        if (discount > cost) {
            throw new IllegalArgumentException("Скидка не может быть больше цены.");
        }
    }

    private void validateExpiry(String expiryDate) {
        if (expiryDate == null || expiryDate.trim().isEmpty()) {
            throw new IllegalArgumentException("Срок действия скидки не может быть пустым.");
        }
    }

    @Override
    public int getCost() {
        return super.getCost() - discount;
    }

    @Override
    public boolean equals(Object obj) {
        if (!super.equals(obj)) return false;
        if (!(obj instanceof DiscountProduct other)) return false;
        return discount == other.discount && expiryDate.equals(other.expiryDate);
    }

    @Override
    public int hashCode() {
        return super.hashCode() + discount * 31 + expiryDate.hashCode();
    }
}
```

Person.java

```
import java.util.ArrayList;
import java.util.List;
```

```java
public class Person {
    private String name;
    private int money;
    private List<Product> products;

    public Person(String name, int money) {
        validName(name);
        validMoney(money);
        this.name = name;
        this.money = money;
        this.products = new ArrayList<>();
    }

    private void validName(String name) {
        if (name == null || name.trim().isEmpty()) {
            throw new IllegalArgumentException("Имя не может быть пустым.");
        }
        if (name.trim().length() < 3) {
            throw new IllegalArgumentException("Имя не может быть короче 3 символов.");
        }
    }

    private void validMoney(int money) {
        if (money < 0) {
            throw new IllegalArgumentException("Деньги не могут быть отрицательными.");
        }
    }

    public String getName() {
        return name;
    }

    public int getMoney() {
        return money;
    }

    public List<Product> getProducts() {
        return products;
    }
```

```
    public void buy(Product product) {
        int price = product.getCost();

        if (money >= price) {
            money -= price;
            products.add(product);
            System.out.println("\n" + name + " купил " + product.getName());
            System.out.println("У покупателя " + name + " осталось " + money + " ye.\n");
        } else {
            System.out.println("\n" + name + " не может позволить себе " + product.getName() + "\n");
            System.out.println("У покупателя " + name + " осталось " + money + " ye.\n");
        }
    }

    @Override
    public String toString() {
        return name;
    }

    @Override
    public int hashCode() {
        return name.hashCode() + money;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (!(obj instanceof Person)) return false;
        Person other = (Person) obj;
        return this.name.equals(other.name) && this.money == other.money;
    }
}
```