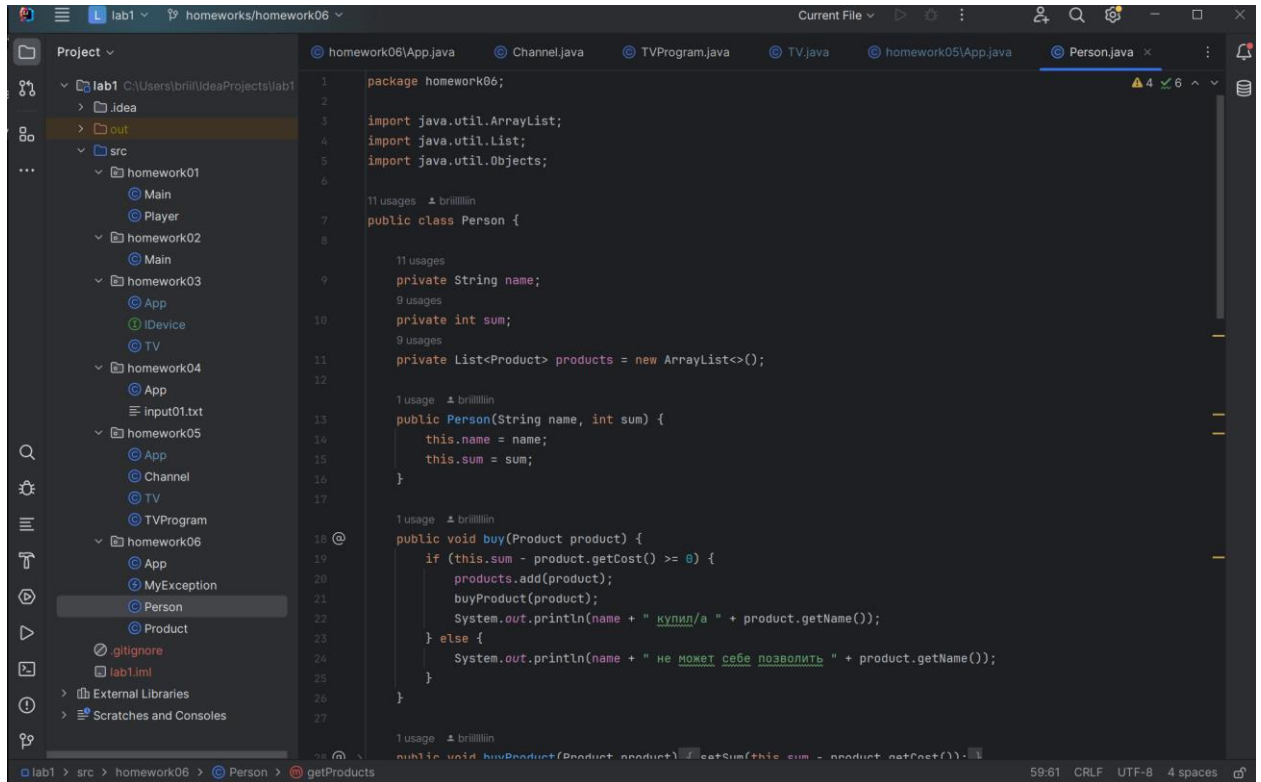
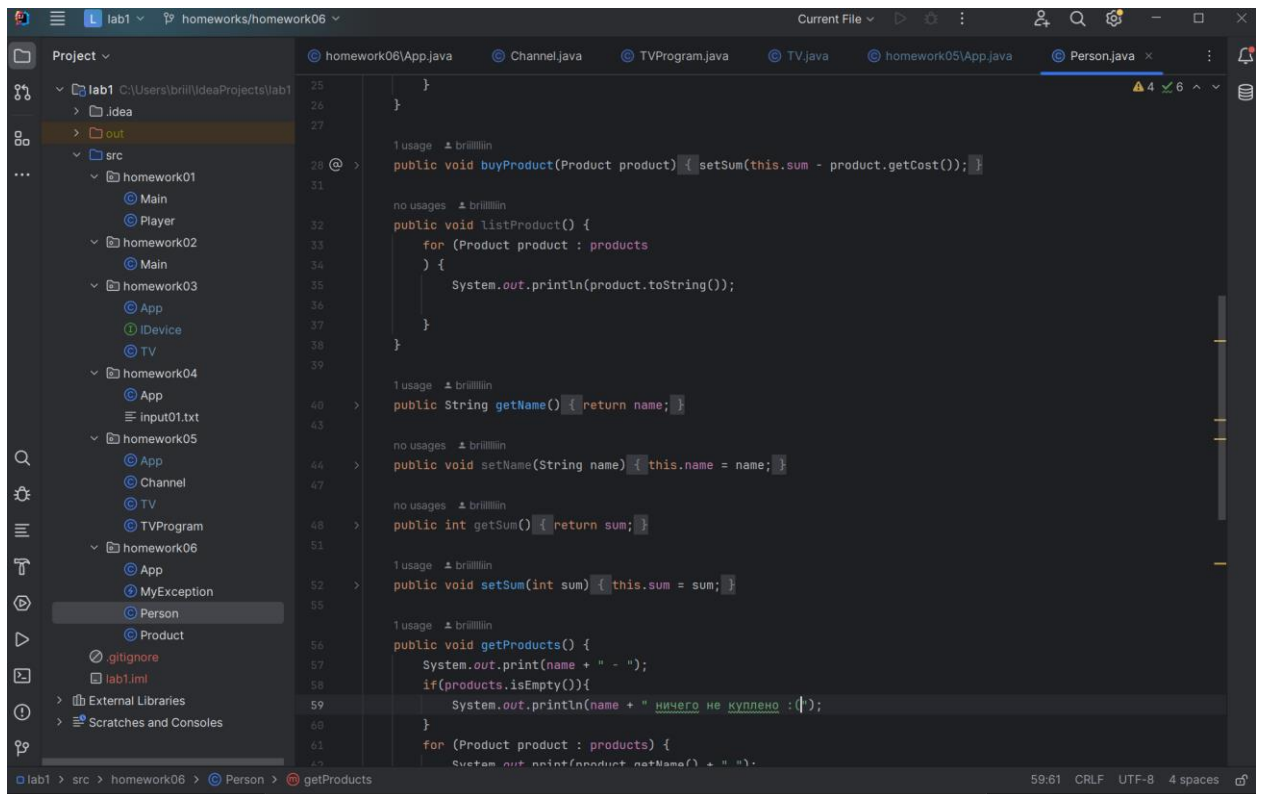


Класс, описывающий сущность «Покупатель»



```
1 package homework06;
2
3 import java.util.ArrayList;
4 import java.util.List;
5 import java.util.Objects;
6
7 11 usages 1 brillian
8 public class Person {
9
10 11 usages
11     private String name;
12 9 usages
13     private int sum;
14 9 usages
15     private List<Product> products = new ArrayList<>();
16
17 1 usage 1 brillian
18     public Person(String name, int sum) {
19         this.name = name;
20         this.sum = sum;
21     }
22
23 1 usage 1 brillian
24     public void buy(Product product) {
25         if (this.sum - product.getCost() >= 0) {
26             products.add(product);
27             buyProduct(product);
28             System.out.println(name + " купил/а " + product.getName());
29         } else {
30             System.out.println(name + " не может себе позволить " + product.getName());
31         }
32     }
33
34 1 usage 1 brillian
35     public void buyProduct(Product product) { setSum(this.sum - product.getCost()); }
```



```
25     }
26
27 1 usage 1 brillian
28     public void buyProduct(Product product) { setSum(this.sum - product.getCost()); }
29
30 no usages 1 brillian
31     public void listProduct() {
32         for (Product product : products)
33         ) {
34             System.out.println(product.toString());
35         }
36     }
37
38 1 usage 1 brillian
39     public String getName() { return name; }
40
41 no usages 1 brillian
42     public void setName(String name) { this.name = name; }
43
44 no usages 1 brillian
45     public int getSum() { return sum; }
46
47 1 usage 1 brillian
48     public void setSum(int sum) { this.sum = sum; }
49
50 1 usage 1 brillian
51     public void getProducts() {
52         System.out.print(name + " - ");
53         if(products.isEmpty()){
54             System.out.println(name + " ничего не куплено :(");
55         }
56         for (Product product : products) {
57             System.out.print(product.getName() + " ");
58         }
59     }
60
61 1 usage 1 brillian
```

Класс, описывающий сущность «Продавец»

The screenshot shows the IntelliJ IDEA IDE with the 'Product.java' file open in the 'homework06' package. The code defines a 'Product' class with attributes 'name' and 'cost', and methods for getting and setting these values. The IDE interface includes a project explorer on the left, a code editor in the center, and a status bar at the bottom.

```
package homework06;

import java.util.Objects;

16 usages  ▲ brillian
public class Product {

    1 usage  ▲ brillian
    public Product(String name, int cost) {
        this.name = name;
        this.cost = cost;
    }

    4 usages  ▲ brillian
    public String getName() { return name; }

    no usages  ▲ brillian
    public void setName(String name) { this.name = name; }

    7 usages
    private String name;

    2 usages  ▲ brillian
    public int getCost() { return cost; }

    no usages  ▲ brillian
    public void setCost(int cost) { this.cost = cost; }

    7 usages
    private int cost;

    ▲ brillian
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
```

Исключение для неверного ввода

The screenshot shows the IntelliJ IDEA IDE with the 'MyException.java' file open in the 'homework06' package. The code defines a 'MyException' class that extends 'Exception'. The IDE interface is consistent with the previous screenshot, showing the project explorer, code editor, and status bar.

```
package homework06;

5 usages  ▲ brillian
public class MyException extends Exception{

    4 usages  ▲ brillian
    public MyException(String message) { super(message); }

    }

9
```

Взаимодействие с классами Product и Person

```
1 package homework06;
2
3 import java.util.ArrayList;
4 import java.util.Iterator;
5 import java.util.List;
6 import java.util.Scanner;
7
8 public class App {
9
10     @usage ± brilliin
11     public static Person createPerson(String name, int sum) {
12         return new Person(name, sum);
13     }
14
15     @usage ± brilliin
16     public static Product createProduct(String name, int cost) {
17         return new Product(name, cost);
18     }
19
20     @usage ± brilliin
21     public static Person getPerson(String name, List<Person> personList) {
22
23         Iterator<Person> iter = personList.iterator();
24
25         while (iter.hasNext()) {
26             Person person = iter.next();
27             if (person.getName().equals(name)) {
28                 return person;
29             }
30         }
31         return null;
32     }
33 }
```

```
31 @usage ± brilliin
32 public static Product getProduct(String name, List<Product> productList) {
33
34     Iterator<Product> iter = productList.iterator();
35
36     while (iter.hasNext()) {
37         Product product = iter.next();
38         if (product.getName().equals(name)) {
39             return product;
40         }
41     }
42     return null;
43 }
44
45 ± brilliin
46 public static void main(String[] args) throws MyException {
47     List<Person> personList = new ArrayList<>();
48     List<Product> productList = new ArrayList<>();
49
50     Scanner scanner = new Scanner(System.in);
51     String line = " ";
52     while (true) {
53         System.out.println("Введите покупателя формата <Имя покупателя> +
54             " = <Сумма, которая у него имеется>, если хотите приступить к вводу продуктов - введите 1");
55         line = scanner.nextLine();
56
57         if (line.isEmpty()) {
58             break;
59         }
60         String[] person = line.split( regex: " = ");
61
62         String name = person[0];
63         if (name.isEmpty()) {
64             continue;
65         }
66     }
67 }
```

```
58 }
59 String[] person = line.split( regex: "=" );
60
61 String name = person[0];
62 if (name.isEmpty()) {
63     throw new MyException("Имя не может быть пустым");
64 }
65 String total = person[1];
66 int sum = Integer.parseInt(total.trim());
67 if (sum < 0) {
68     throw new MyException("Деньги не могут быть отрицательными");
69 }
70
71 personList.add(createPerson(name, sum));
72
73 }
74
75
76
77 while (true) {
78     System.out.println("Введите продукт формата <Название продукта> +
79                         " = <Стоимость продукта>");
80     line = scanner.nextLine();
81
82     if (line.isEmpty()) {
83         break;
84     }
85     String[] product = line.split( regex: "=" );
86     String name = product[0];
87     if (name.isEmpty()) {
88         throw new MyException("Имя не может быть пустым");
89     }
90     String total = product[1];
91     int sum = Integer.parseInt(total.trim());
92     if (sum < 0) {
93         throw new MyException("Деньги не могут быть отрицательными");
94     }
95     productList.add(createProduct(name, sum));
96 }
97
98 while (true) {
99     String answer = scanner.nextLine();
100
101     if (answer.equals("END")) {
102         for (Person person : personList) {
103             person.getProducts();
104         }
105         break;
106     }
107     String[] data = answer.split( regex: "=" );
108
109     Person currPerson = getPerson(data[0], personList);
110     Product currProduct = getProduct(data[1], productList);
111
112     if (currPerson != null && currProduct != null) {
113         currPerson.buy(currProduct);
114     }
115 }
```

```
88 throw new MyException("Имя не может быть пустым");
89 }
90 String total = product[1];
91 int sum = Integer.parseInt(total.trim());
92 if (sum < 0) {
93     throw new MyException("Деньги не могут быть отрицательными");
94 }
95 productList.add(createProduct(name, sum));
96
97 }
98
99 while (true) {
100     String answer = scanner.nextLine();
101
102     if (answer.equals("END")) {
103         for (Person person : personList) {
104             person.getProducts();
105         }
106         break;
107     }
108     String[] data = answer.split( regex: "=" );
109
110     Person currPerson = getPerson(data[0], personList);
111     Product currProduct = getProduct(data[1], productList);
112
113     if (currPerson != null && currProduct != null) {
114         currPerson.buy(currProduct);
115     }
116 }
```

