Zhubatkanov

Aidyn

1)

```
package geometry;

import java.io.Serializable;

public abstract class Shape implements Serializable {     private static final long
serialVersionUID = 1L;

    public abstract double calculateArea(); }
```

```
package geometry;
public class Circle extends Shape {     private double radius;

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

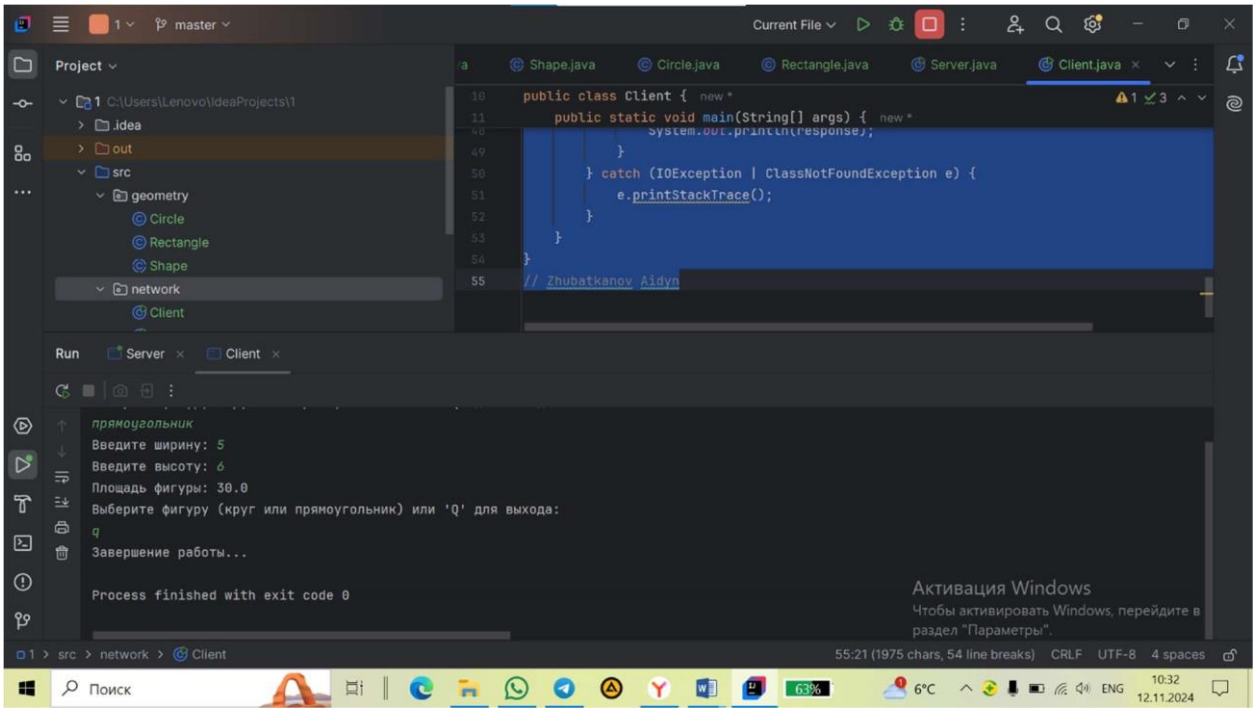
```
package geometry;

public class Rectangle extends Shape {     private double
width;     private double height;

    public Rectangle(double width, double height) {         this.width = width;
this.height = height;
    }

    @Override
    public double calculateArea() {         return width * height;
    }
}
```

```
package network;  import
geometry.Shape; import
java.io.*; import java.net.*;
 public class Server {
```

```
    public static void main(String[] args) {
        try (ServerSocket serverSocket = new ServerSocket(5000)) {
            System.out.println("Сервер запущен и ожидает подключения...");            while (true) {
                try (Socket socket = serverSocket.accept();
                     ObjectInputStream input = new
ObjectInputStream(socket.getInputStream());
                     ObjectOutputStream output = new
ObjectOutputStream(socket.getOutputStream())) {

                    System.out.println("Клиент подключен");



                    Shape shape = (Shape) input.readObject();                    if (shape != null) {
                        double area = shape.calculateArea();                    output.writeObject("Площадь фигуры: " +
area);                        output.flush();
                        System.out.println("Площадь отправлена клиенту: " + area);
                    } else {                        break;
                    }
                } catch (ClassNotFoundException e) {
                    e.printStackTrace();
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

```
package network;

import geometry.Circle; import
geometry.Rectangle; import
geometry.Shape; import java.io.*;
import java.net.*; import
java.util.Scanner;

public class Client {
    public static void main(String[] args) {
        try (Socket socket = new Socket("localhost", 5000);
             ObjectOutputStream output = new
ObjectOutputStream(socket.getOutputStream());
             ObjectInputStream input = new
ObjectInputStream(socket.getInputStream());
             Scanner scanner = new Scanner(System.in)) {

            while (true) {
                System.out.println("Выберите фигуру (круг или прямоугольник) или 'Q' для выхода:");
```

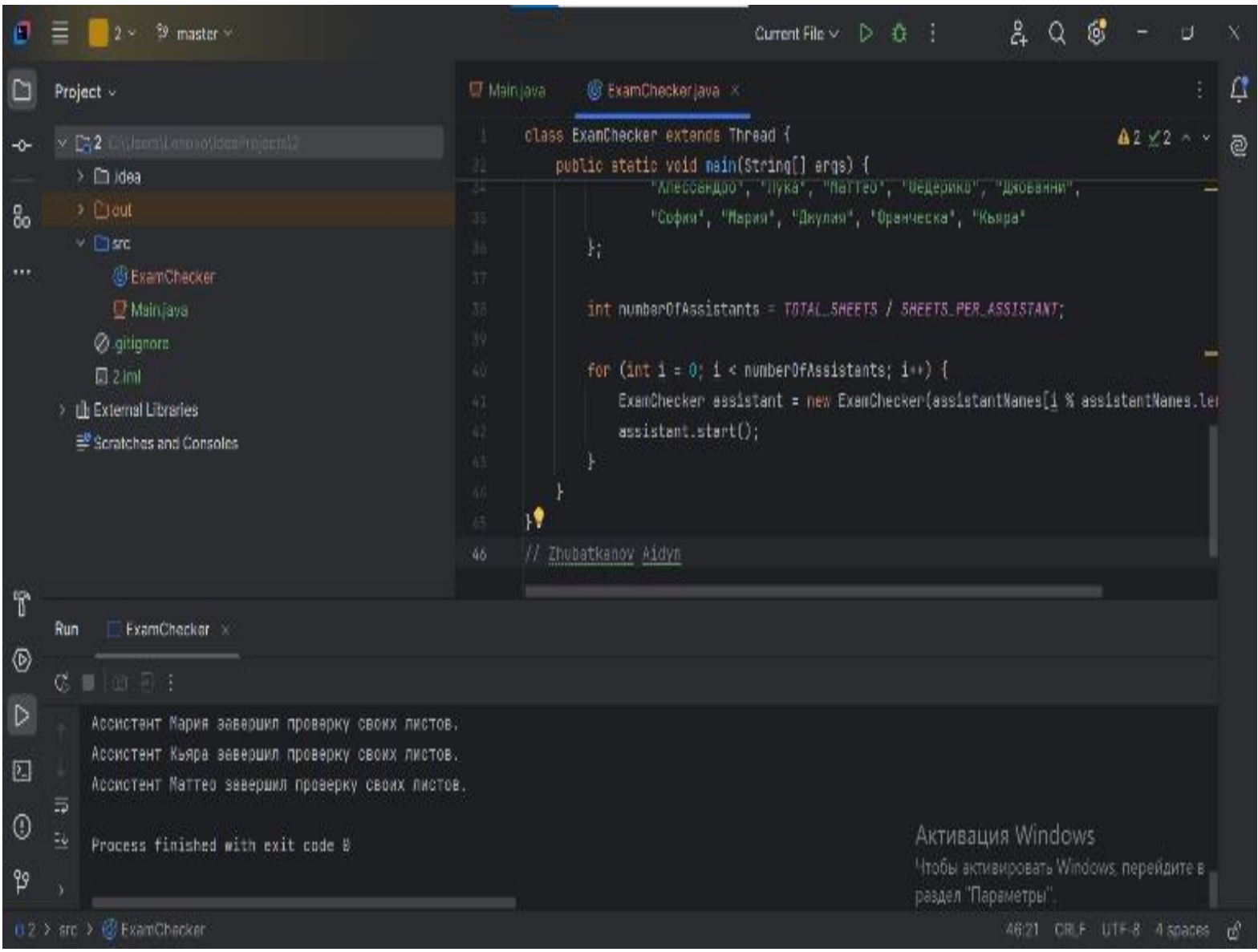
```
            String choice = scanner.nextLine();

            if (choice.equalsIgnoreCase("q")) {
                System.out.println("Завершение работы...");                    output.writeObject(null);
                output.flush();
                break;
            }

            Shape shape = null;
            if (choice.equalsIgnoreCase("круг")) {                    System.out.print("Введите
радиус: ");                    double radius = scanner.nextDouble();                    shape = new
Circle(radius);
            } else if (choice.equalsIgnoreCase("прямоугольник")) {
                System.out.print("Введите ширину: ");                    double width =
scanner.nextDouble();                    System.out.print("Введите высоту: ");
double height = scanner.nextDouble();                    shape = new Rectangle(width, height);
            }
            scanner.nextLine();

            output.writeObject(shape);                output.flush();
            // otvet server
            String response = (String) input.readObject();
            System.out.println(response);
        }
    } catch (IOException | ClassNotFoundException e) {
        e.printStackTrace();
    }
  }
}
// Zhubatkanov Aidyn
```



2)

```
class ExamChecker extends Thread {
    private static final int TOTAL_SHEETS = 500;
    private static final int SHEETS_PER_ASSISTANT = 50;
    private static final int CHECK_PER_ITERATION = 6;
    private String assistantName;
    private int sheetsToCheck;

    public ExamChecker(String assistantName) {
        this.assistantName = assistantName;
        this.sheetsToCheck = SHEETS_PER_ASSISTANT;
    }

    @Override
    public void run() {
        try {
            while (sheetsToCheck > 0) {
                int checkNow =
Math.min(CHECK_PER_ITERATION, sheetsToCheck);
                sheetsToCheck -= checkNow;

                System.out.println("Ассистент " + assistantName
+ " проверяет " + checkNow +
                        " листов. Осталось проверить: " +
sheetsToCheck + " листов.");


                Thread.sleep((int) (Math.random() * 200 +
100));
```

```
            }
            System.out.println("Ассистент " + assistantName +
" завершил проверку своих листов.");
        } catch (InterruptedException e) {
            System.out.println("Ассистент " + assistantName +
" был прерван.");
        }
    }

    public static void main(String[] args) {
        String[] assistantNames = {
                "Алессандро", "Лука", "Маттео", "Федерико",
"Джованни",
                "София", "Мария", "Джулия", "Франческа",
"Кьяра"
        };

        int numberOfAssistants = TOTAL_SHEETS /
SHEETS_PER_ASSISTANT;

        for (int i = 0; i < numberOfAssistants; i++) {
            ExamChecker assistant = new
ExamChecker(assistantNames[i %
assistantNames.length]);
            assistant.start();
        }
    }
}
```


2
master
Current File
Project
2
idea
out
src
ExamChecker
Main.java
.gitignore
2.iml
External Libraries
Scratches and Consoles
Main.java
ExamChecker.java
class ExamChecker extends Thread {
public static void main(String[] args) {
"Алессандро", "Лука", "Маттео", "Федерико", "Джованни",
"Софи", "Мария", "Джулия", "Франческа", "Кьяра"
};
int numberOfAssistants = TOTAL_SHEETS / SHEETS_PER_ASSISTANT;
for (int i = 0; i < numberOfAssistants; i++) {
ExamChecker assistant = new ExamChecker(assistantNames[i % assistantNames.le
assistant.start();
}
}
}
// Zhubatkanov Aidyn
Run
ExamChecker
Ассистент Мария завершил проверку своих листов.
Ассистент Кьяра завершил проверку своих листов.
Ассистент Маттео завершил проверку своих листов.
Process finished with exit code 0
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".
2 > src > ExamChecker
46:21
CRLF
UTF-8
4 spaces