Задача 1

import java.util.Scanner;  
  
public class MatrixElementSwap {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 System.out.print("Введите количество строк: ");  
 int rows = scanner.nextInt();  
 System.out.print("Введите количество столбцов: ");  
 int cols = scanner.nextInt();  
  
 int[][] matrix = new int[rows][cols];  
 System.out.println("Введите элементы матрицы:");  
 for (int i = 0; i < rows; i++) {  
 for (int j = 0; j < cols; j++) {  
 matrix[i][j] = scanner.nextInt();  
 }  
 }  
  
 int minElement = matrix[0][0];  
 int minRow = 0, minCol = 0;  
 for (int i = 0; i < rows; i++) {  
 for (int j = 0; j < cols; j++) {  
 if (matrix[i][j] < minElement) {  
 minElement = matrix[i][j];  
 minRow = i;  
 minCol = j;  
 }  
 }  
 }  
  
 System.out.print("Введите строку целевого элемента: ");  
 int targetRow = scanner.nextInt() - 1;  
 System.out.print("Введите столбец целевого элемента: ");  
 int targetCol = scanner.nextInt() - 1;  
  
 if (minRow != targetRow) {  
 for (int j = 0; j < cols; j++) {  
 int temp = matrix[minRow][j];  
 matrix[minRow][j] = matrix[targetRow][j];  
 matrix[targetRow][j] = temp;  
 }  
 }  
  
 if (minCol != targetCol) {  
 for (int i = 0; i < rows; i++) {  
 int temp = matrix[i][minCol];  
 matrix[i][minCol] = matrix[i][targetCol];  
 matrix[i][targetCol] = temp;  
 }  
 }  
  
 System.out.println("Результат:");  
 for (int i = 0; i < rows; i++) {  
 for (int j = 0; j < cols; j++) {  
 System.out.print(matrix[i][j] + " ");  
 }  
 System.out.println();  
 }  
  
 scanner.close();  
 }  
}

Задача 2

import java.util.Arrays;

class Polynomial {

private double[] coefficients;

public Polynomial(double[] coefficients) {

this.coefficients = Arrays.copyOf(coefficients, coefficients.length);

}

public int getDegree() {

return coefficients.length - 1;

}

public Polynomial add(Polynomial other) {

int maxDegree = Math.max(this.getDegree(), other.getDegree());

double[] result = new double[maxDegree + 1];

for (int i = 0; i <= this.getDegree(); i++) {

result[i] += this.coefficients[i];

}

for (int i = 0; i <= other.getDegree(); i++) {

result[i] += other.coefficients[i];

}

return new Polynomial(result);

}

@Override

public String toString() {

StringBuilder sb = new StringBuilder();

for (int i = getDegree(); i >= 0; i--) {

if (coefficients[i] != 0) {

if (sb.length() > 0) {

sb.append(coefficients[i] > 0 ? " + " : " - ");

} else if (coefficients[i] < 0) {

sb.append("-");

}

double absCoeff = Math.abs(coefficients[i]);

if (absCoeff != 1 || i == 0) {

sb.append(absCoeff);

}

if (i > 0) {

sb.append("x");

if (i > 1) {

sb.append("^").append(i);

}

}

}

}

return sb.length() > 0 ? sb.toString() : "0";

}

public static Polynomial sumPolynomials(Polynomial[] polynomials) {

if (polynomials.length == 0) {

return new Polynomial(new double[]{0});

}

Polynomial sum = polynomials[0];

for (int i = 1; i < polynomials.length; i++) {

sum = sum.add(polynomials[i]);

}

return sum;

}

}

public class PolynomialTest {

public static void main(String[] args) {

Polynomial p1 = new Polynomial(new double[]{1, 2, 3}); // 3x^2 + 2x + 1

Polynomial p2 = new Polynomial(new double[]{0, 1, 1}); // x^2 + x

Polynomial p3 = new Polynomial(new double[]{5, 0, 0, 4}); // 4x^3 + 5

Polynomial[] polynomials = {p1, p2, p3};

System.out.println("Полиномы:");

for (int i = 0; i < polynomials.length; i++) {

System.out.println("p" + (i + 1) + " = " + polynomials[i]);

}

Polynomial sum = Polynomial.sumPolynomials(polynomials);

System.out.println("\nСумма полиномов:");

System.out.println("sum = " + sum);

}

}

Задача 3

import java.util.ArrayList;  
import java.util.List;  
  
class Petal {  
 private String color;  
  
 public Petal(String color) {  
 this.color = color;  
 }  
  
 public String getColor() {  
 return color;  
 }  
}  
  
class Bud {  
 private List<Petal> petals;  
 private boolean isBlossomed;  
  
 public Bud(String color, int petalCount) {  
 petals = new ArrayList<>();  
 for (int i = 0; i < petalCount; i++) {  
 petals.add(new Petal(color));  
 }  
 isBlossomed = false;  
 }  
  
 public void blossom() {  
 isBlossomed = true;  
 }  
  
 public void wither() {  
 isBlossomed = false;  
 }  
  
 public String getColor() {  
 return petals.isEmpty() ? "Нет лепестков" : petals.get(0).getColor();  
 }  
  
 public boolean isBlossomed() {  
 return isBlossomed;  
 }  
}  
  
class Flower {  
 private Bud bud;  
  
 public Flower(String color, int petalCount) {  
 this.bud = new Bud(color, petalCount);  
 }  
  
 public void blossom() {  
 bud.blossom();  
 System.out.println("Цветок распустился!");  
 }  
  
 public void wither() {  
 bud.wither();  
 System.out.println("Цветок завял(");  
 }  
  
 public void displayBudColor() {  
 System.out.println("Цвет бутона: " + bud.getColor());  
 }  
  
 public String getState() {  
 return bud.isBlossomed() ? "Расцвел" : "Не расцвел";  
 }  
}  
  
public class FlowerDemo {  
 public static void main(String[] args) {  
 Flower rose = new Flower("Красный", 5);  
  
 rose.displayBudColor();  
 System.out.println("Исходное состояние: " + rose.getState());  
  
 rose.blossom();  
 System.out.println("После цветения: " + rose.getState());  
  
 rose.wither();  
 System.out.println("После увядания: " + rose.getState());  
 }  
}

Задача 4

import java.util.ArrayList;  
import java.util.List;  
  
public class Park {  
 private String name;  
 private List<Attraction> attractions;  
  
 public Park(String name) {  
 this.name = name;  
 this.attractions = new ArrayList<>();  
 }  
  
 public void addAttraction(String name, String workingHours, double price) {  
 attractions.add(new Attraction(name, workingHours, price));  
 }  
  
 public void getAttractions() {  
 for (Attraction attraction : attractions) {  
 System.out.println(attraction);  
 }  
 return ;  
 }  
  
 public class Attraction {  
 private String name;  
 private String workingHours;  
 private double price;  
  
 public Attraction(String name, String workingHours, double price) {  
 this.name = name;  
 this.workingHours = workingHours;  
 this.price = price;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public String getWorkingHours() {  
 return workingHours;  
 }  
  
 public double getPrice() {  
 return price;  
 }  
  
 @Override  
 public String toString() {  
 return "Attraction{" +  
 "name='" + name + '\'' +  
 ", workingHours='" + workingHours + '\'' +  
 ", price=" + price +  
 '}';  
 }  
 }  
  
 @Override  
 public String toString() {  
 return "Park{" +  
 "name='" + name + '\'' +  
 ", attractions=" + attractions +  
 '}';  
 }  
}

Класс Main

public class Main{  
 public static void main(String[] args) {  
 Park park = new Park("Юность");  
 park.addAttraction("Колесо оборзения", "9:00 - 20:00",350);  
 park.addAttraction("Амэрикански горки","10:00 - 20:00",100);  
 park.getAttractions();  
 }  
}

Задача 5

import java.util.\*;  
  
public class PoemCommonLettersFinder {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 System.out.println("Введите стихотворение (для завершения введите пустую строку):");  
  
 StringBuilder poemBuilder = new StringBuilder();  
 String line;  
 while (!(line = scanner.nextLine()).isEmpty()) {  
 poemBuilder.append(line).append("\n");  
 }  
  
 String poem = poemBuilder.toString().toLowerCase();  
 String[] words = poem.split("\\s+");  
  
 if (words.length == 0) {  
 System.out.println("Стихотворение не содержит слов.");  
 return;  
 }  
  
 Set<Character> commonLetters = new HashSet<>();  
 for (char c : words[0].toCharArray()) {  
 if (Character.isLetter(c)) {  
 commonLetters.add(c);  
 }  
 }  
  
 for (int i = 1; i < words.length; i++) {  
 Set<Character> currentWordLetters = new HashSet<>();  
 for (char c : words[i].toCharArray()) {  
 if (Character.isLetter(c)) {  
 currentWordLetters.add(c);  
 }  
 }  
 commonLetters.retainAll(currentWordLetters);  
 }  
  
 if (commonLetters.isEmpty()) {  
 System.out.println("Нет общих букв во всех словах стихотворения.");  
 } else {  
 System.out.println("Общие буквы во всех словах стихотворения: " + commonLetters);  
 }  
 }  
}

Задача 6

import java.io.\*;  
import java.util.ArrayList;  
import java.util.List;  
  
class ColorException extends Exception {  
 public ColorException(String message) {  
 super(message);  
 }  
}  
  
class PetalCountException extends Exception {  
 public PetalCountException(String message) {  
 super(message);  
 }  
}  
  
class Petal implements Serializable {  
 private String color;  
  
 public Petal(String color) throws ColorException {  
 if (color == null || color.trim().isEmpty()) {  
 throw new ColorException("Цвет лепестка не может быть пустым");  
 }  
 this.color = color;  
 }  
  
 public String getColor() {  
 return color;  
 }  
}  
  
class Bud implements Serializable {  
 private List<Petal> petals;  
 private boolean isBlossomed;  
  
 public Bud(String color, int petalCount) throws ColorException, PetalCountException {  
 if (petalCount <= 0) {  
 throw new PetalCountException("Количество лепестков должно быть положительным");  
 }  
 petals = new ArrayList<>();  
 for (int i = 0; i < petalCount; i++) {  
 petals.add(new Petal(color));  
 }  
 isBlossomed = false;  
 }  
  
 public void blossom() {  
 isBlossomed = true;  
 }  
  
 public void wither() {  
 isBlossomed = false;  
 }  
  
 public String getColor() throws ColorException {  
 if (petals.isEmpty()) {  
 throw new ColorException("У бутона нет лепестков");  
 }  
 return petals.get(0).getColor();  
 }  
  
 public boolean isBlossomed() {  
 return isBlossomed;  
 }  
}  
  
class Flower implements Serializable {  
 private Bud bud;  
  
 public Flower(String color, int petalCount) throws ColorException, PetalCountException {  
 this.bud = new Bud(color, petalCount);  
 }  
  
 public void blossom() {  
 bud.blossom();  
 System.out.println("Цветок распустился!");  
 }  
  
 public void wither() {  
 bud.wither();  
 System.out.println("Цветок завял(");  
 }  
  
 public void displayBudColor() {  
 try {  
 System.out.println("Цвет бутона: " + bud.getColor());  
 } catch (ColorException e) {  
 System.out.println("Ошибка: " + e.getMessage());  
 }  
 }  
  
 public String getState() {  
 return bud.isBlossomed() ? "Расцвел" : "Не расцвел";  
 }  
  
 public void saveToFile(String filename) throws IOException {  
 try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(filename))) {  
 oos.writeObject(this);  
 }  
 }  
  
 public static Flower loadFromFile(String filename) throws IOException, ClassNotFoundException {  
 try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(filename))) {  
 return (Flower) ois.readObject();  
 }  
 }  
  
  
}  
  
public class FlowerDemo {  
 public static void main(String[] args) {  
 try {  
 Flower rose = new Flower("Красный", 5);  
 rose.displayBudColor();  
 System.out.println("Исходное состояние: " + rose.getState());  
  
 rose.blossom();  
 System.out.println("После цветения: " + rose.getState());  
  
 rose.wither();  
 System.out.println("После увядания: " + rose.getState());  
  
 rose.saveToFile("rose.dat");  
 System.out.println("Цветок сохранен в файл");  
  
 Flower loadedRose = Flower.loadFromFile("rose.dat");  
 System.out.println("Цветок загружен из файла");  
 loadedRose.displayBudColor();  
  
 // Пример обработки математической ошибки  
 try {  
 int result = divideByZero(10, 0);  
 } catch (ArithmeticException e) {  
 System.out.println("Математическая ошибка: " + e.getMessage());  
 }  
  
 //Вызов ошибки цвета  
 System.out.println("\nПопытка создать цветок с пустым цветом...");  
 Flower invalidFlower = new Flower("", 3);  
  
 } catch (ColorException e) {  
 System.out.println("Ошибка цвета: " + e.getMessage());  
 } catch (PetalCountException e) {  
 System.out.println("Ошибка количества лепестков: " + e.getMessage());  
 } catch (IOException e) {  
 System.out.println("Ошибка ввода/вывода: " + e.getMessage());  
 } catch (ClassNotFoundException e) {  
 System.out.println("Ошибка при загрузке класса: " + e.getMessage());  
 } catch (OutOfMemoryError e) {  
 System.out.println("Ошибка: недостаточно памяти");  
 System.out.println("Описание: " + e.getMessage());  
 }  
 }  
  
 // Пример метода, который может вызвать математическую ошибку  
 private static int divideByZero(int a, int b) {  
 return a / b;  
 }  
}

Задача 7

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.geom.\*;  
  
public class CocktailDrawing extends JPanel {  
  
 @Override  
 protected void paintComponent(Graphics g) {  
 super.paintComponent(g);  
 Graphics2D g2d = (Graphics2D) g;  
 g2d.setRenderingHint(RenderingHints.KEY\_ANTIALIASING, RenderingHints.VALUE\_ANTIALIAS\_ON);  
  
   
 g2d.setColor(Color.BLACK);  
 g2d.fillOval(115, 297, 70, 10);  
 g2d.setColor(Color.WHITE);  
 g2d.fillOval(117, 298, 65, 8);  
   
 g2d.setColor(Color.BLACK);  
 g2d.fillRect(145, 250, 10, 50);  
 g2d.setColor(Color.WHITE);  
 g2d.fillRect(148, 254, 3, 42);  
  
   
 g2d.setColor(Color.LIGHT\_GRAY);  
 g2d.fill(new Path2D.Double() {{  
 moveTo(105, 120);  
 lineTo(131, 250);  
 lineTo(169, 250);  
 lineTo(195, 120);  
 closePath();  
  
  
 }});  
  
 g2d.setColor(Color.DARK\_GRAY);  
 g2d.setStroke(new BasicStroke(2));  
 g2d.drawOval(130,160,10,10);  
 g2d.drawOval(160,145,15,15);  
 g2d.drawOval(150,190,13,13);  
 g2d.drawOval(135,130,17,17);  
 g2d.drawOval(130,220,14,14);  
 g2d.drawOval(130,192,12,12);  
 g2d.drawOval(155,225,11,11);  
  
 // Рисуем соломинку  
 g2d.setColor(Color.BLACK);  
 g2d.setStroke(new BasicStroke(7));  
 g2d.draw(new Line2D.Double(180, 80, 140, 240));  
 g2d.setColor(Color.WHITE);  
 g2d.setStroke(new BasicStroke(3));  
 g2d.draw(new Line2D.Double(180, 80, 140, 240));  
  
  
 g2d.draw(new Path2D.Double() {{  
 g2d.setColor(Color.BLACK);  
 g2d.setStroke(new BasicStroke(3));  
 moveTo(100, 100);  
 lineTo(130, 250);  
 lineTo(170, 250);  
 lineTo(200, 100);  
 lineTo(100, 100);  
 }});  
  
 g2d.draw(new Path2D.Double() {{  
 g2d.setStroke(new BasicStroke(3));  
 g2d.setColor(Color.DARK\_GRAY);  
 moveTo(105, 120);  
 lineTo(195, 120);  
 }});  
  
 }  
  
 public static void main(String[] args) {  
 JFrame frame = new JFrame("Cocktail Drawing");  
 frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 frame.add(new CocktailDrawing());  
 frame.setSize(300, 400);  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
 }  
}

Задача 8

public class StringArrayGUI extends JFrame {  
 private StringArray array1;  
 private StringArray array2;  
 private JTextArea outputArea;  
 private JTextField inputField;  
 private JComboBox<String> arraySelector;  
  
 public StringArrayGUI() {  
 array1 = new StringArray(5);  
 array2 = new StringArray(5);  
  
 setTitle("StringArray GUI");  
 setSize(500, 400);  
 setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
  
 JPanel mainPanel = new JPanel(new BorderLayout());  
  
 // Input panel  
 JPanel inputPanel = new JPanel();  
 inputField = new JTextField(20);  
 JButton addButton = new JButton("Add String");  
 arraySelector = new JComboBox<>(new String[]{"Array 1", "Array 2"});  
 inputPanel.add(arraySelector);  
 inputPanel.add(inputField);  
 inputPanel.add(addButton);  
  
 // Output area  
 outputArea = new JTextArea(15, 40);  
 outputArea.setEditable(false);  
 JScrollPane scrollPane = new JScrollPane(outputArea);  
  
 // Buttons panel  
 JPanel buttonPanel = new JPanel();  
 JButton printButton = new JButton("Print Arrays");  
 JButton concatenateButton = new JButton("Concatenate");  
 JButton mergeButton = new JButton("Merge");  
 buttonPanel.add(printButton);  
 buttonPanel.add(concatenateButton);  
 buttonPanel.add(mergeButton);  
  
 mainPanel.add(inputPanel, BorderLayout.NORTH);  
 mainPanel.add(scrollPane, BorderLayout.CENTER);  
 mainPanel.add(buttonPanel, BorderLayout.SOUTH);  
  
 add(mainPanel);  
  
 // Add action listeners  
 addButton.addActionListener(e -> addString());  
 printButton.addActionListener(e -> printArrays());  
 concatenateButton.addActionListener(e -> concatenateArrays());  
 mergeButton.addActionListener(e -> mergeArrays());  
 }  
  
 private void addString() {  
 String input = inputField.getText();  
 if (!input.isEmpty()) {  
 StringArray selectedArray = arraySelector.getSelectedIndex() == 0 ? array1 : array2;  
 for (int i = 0; i < selectedArray.getSize(); i++) {  
 if (selectedArray.getString(i) == null) {  
 selectedArray.setString(i, input);  
 inputField.setText("");  
 outputArea.append("Added \"" + input + "\" to " + arraySelector.getSelectedItem() + "\n");  
 return;  
 }  
 }  
 outputArea.append("Array is full. Cannot add more strings.\n");  
 }  
 }  
  
 private void printArrays() {  
 outputArea.setText(""); // Clear previous output  
 outputArea.append("Array 1:\n");  
 array1.printAll(outputArea);  
 outputArea.append("\nArray 2:\n");  
 array2.printAll(outputArea);  
 }  
  
 private void concatenateArrays() {  
 StringArray result = array1.concatenate(array2);  
 outputArea.setText("Concatenated Array:\n");  
 result.printAll(outputArea);  
 }  
  
 private void mergeArrays() {  
 StringArray result = array1.merge(array2);  
 outputArea.setText("Merged Array:\n");  
 result.printAll(outputArea);  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.invokeLater(() -> new StringArrayGUI().setVisible(true));  
 }  
}

Класс StringArray

class StringArray {  
 private String[] elements;  
 private int[] lengths;  
  
 public StringArray(int size) {  
 elements = new String[size];  
 lengths = new int[size];  
 }  
  
 public void setString(int index, String str) {  
 if (index < 0 || index >= elements.length) {  
 throw new IndexOutOfBoundsException("Индекс выходит за границы массива");  
 }  
 elements[index] = str;  
 lengths[index] = str.length();  
 }  
  
 public String getString(int index) {  
 if (index < 0 || index >= elements.length) {  
 throw new IndexOutOfBoundsException("Индекс выходит за границы массива");  
 }  
 return elements[index];  
 }  
  
 public int getLength(int index) {  
 if (index < 0 || index >= lengths.length) {  
 throw new IndexOutOfBoundsException("Индекс выходит за границы массива");  
 }  
 return lengths[index];  
 }  
  
 public int getSize() {  
 return elements.length;  
 }  
  
 public StringArray concatenate(StringArray other) {  
 StringArray result = new StringArray(this.elements.length + other.elements.length);  
 int index = 0;  
 for (String str : this.elements) {  
 if (str != null) {  
 result.setString(index++, str);  
 }  
 }  
 for (String str : other.elements) {  
 if (str != null) {  
 result.setString(index++, str);  
 }  
 }  
 return result;  
 }  
  
 public StringArray merge(StringArray other) {  
 Set<String> uniqueStrings = new HashSet<>();  
 for (String str : this.elements) {  
 if (str != null) {  
 uniqueStrings.add(str);  
 }  
 }  
 for (String str : other.elements) {  
 if (str != null) {  
 uniqueStrings.add(str);  
 }  
 }  
 StringArray result = new StringArray(uniqueStrings.size());  
 int index = 0;  
 for (String str : uniqueStrings) {  
 result.setString(index++, str);  
 }  
 return result;  
 }  
  
 public void printAll(JTextArea outputArea) {  
 for (int i = 0; i < elements.length; i++) {  
 if (elements[i] != null) {  
 outputArea.append(i + ": " + elements[i] + "\n");  
 }  
 }  
 }  
}

Задача 9

Класс MatematikoGame

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.\*;  
import java.util.Random;  
  
public class MatematikoGame extends JFrame {  
 private JButton[][] gameBoard;  
 private JLabel currentNumberLabel;  
 private JLabel playerScoreLabel;  
 private JLabel computerScoreLabel;  
 private JButton drawCardButton;  
  
 private int currentNumber;  
 private int playerScore = 0;  
 private int computerScore = 0;  
 private Random random = new Random();  
 private ComputerPlayer computerPlayer = new ComputerPlayer();  
 private String[][] boardState = new String[5][5];  
  
 public MatematikoGame() {  
 setTitle("Математико");  
 setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 setSize(500, 600);  
 setLayout(new BorderLayout());  
  
 // Панель игрового поля  
 JPanel boardPanel = new JPanel(new GridLayout(5, 5));  
 gameBoard = new JButton[5][5];  
 for (int i = 0; i < 5; i++) {  
 for (int j = 0; j < 5; j++) {  
 gameBoard[i][j] = new JButton();  
 gameBoard[i][j].setFont(new Font("Arial", Font.BOLD, 20));  
 gameBoard[i][j].addActionListener(new CellClickListener(i, j));  
 boardPanel.add(gameBoard[i][j]);  
 boardState[i][j] = "";  
 }  
 }  
 add(boardPanel, BorderLayout.CENTER);  
  
 // Панель информации  
 JPanel infoPanel = new JPanel(new GridLayout(4, 1));  
 currentNumberLabel = new JLabel("Текущее число: ", JLabel.CENTER);  
 playerScoreLabel = new JLabel("Счет игрока: 0", JLabel.CENTER);  
 computerScoreLabel = new JLabel("Счет компьютера: 0", JLabel.CENTER);  
 drawCardButton = new JButton("Начать игру");  
 drawCardButton.addActionListener(e -> {  
 drawCard();  
 drawCardButton.setEnabled(false);  
 });  
  
 infoPanel.add(currentNumberLabel);  
 infoPanel.add(playerScoreLabel);  
 infoPanel.add(computerScoreLabel);  
 infoPanel.add(drawCardButton);  
 add(infoPanel, BorderLayout.SOUTH);  
  
 setVisible(true);  
 }  
  
 private void drawCard() {  
 currentNumber = random.nextInt(13) + 1;  
 currentNumberLabel.setText("Текущее число: " + currentNumber);  
  
 if (isGameOver()) {  
 endGame();  
 }  
 }  
  
 private class CellClickListener implements ActionListener {  
 private int row, col;  
  
 public CellClickListener(int row, int col) {  
 this.row = row;  
 this.col = col;  
 }  
  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 if (boardState[row][col].isEmpty()) {  
 boardState[row][col] = String.valueOf(currentNumber);  
 gameBoard[row][col].setText(String.valueOf(currentNumber));  
 playerScore = ScoringLogic.calculateTotalScore(boardState);  
 playerScoreLabel.setText("Счет игрока: " + playerScore);  
  
 if (!isGameOver()) {  
 computerMove();  
 } else {  
 endGame();  
 }  
 }  
 }  
 }  
  
 private void computerMove() {  
 drawCard(); // Получаем новое число для компьютера  
 int[] move = computerPlayer.makeMove(boardState, currentNumber);  
 boardState[move[0]][move[1]] = String.valueOf(currentNumber);  
 gameBoard[move[0]][move[1]].setText(String.valueOf(currentNumber));  
 computerScore = ScoringLogic.calculateTotalScore(boardState);  
 computerScoreLabel.setText("Счет компьютера: " + computerScore);  
  
 if (isGameOver()) {  
 endGame();  
 } else {  
 drawCard(); // Получаем новое число для игрока  
 }  
 }  
  
 private boolean isGameOver() {  
 for (int i = 0; i < 5; i++) {  
 for (int j = 0; j < 5; j++) {  
 if (boardState[i][j].isEmpty()) {  
 return false;  
 }  
 }  
 }  
 return true;  
 }  
  
 private void endGame() {  
 String message;  
 if (playerScore > computerScore) {  
 message = "Вы выиграли!\nВаш счет: " + playerScore + "\nСчет компьютера: " + computerScore;  
 } else if (computerScore > playerScore) {  
 message = "Компьютер выиграл!\nВаш счет: " + playerScore + "\nСчет компьютера: " + computerScore;  
 } else {  
 message = "Ничья!\nВаш счет: " + playerScore + "\nСчет компьютера: " + computerScore;  
 }  
  
 int choice = JOptionPane.showConfirmDialog(this, message + "\nХотите начать новую игру?", "Игра окончена", JOptionPane.YES\_NO\_OPTION);  
 if (choice == JOptionPane.YES\_OPTION) {  
 resetGame();  
 } else {  
 System.exit(0);  
 }  
 }  
  
 private void resetGame() {  
 for (int i = 0; i < 5; i++) {  
 for (int j = 0; j < 5; j++) {  
 boardState[i][j] = "";  
 gameBoard[i][j].setText("");  
 }  
 }  
 playerScore = 0;  
 computerScore = 0;  
 playerScoreLabel.setText("Счет игрока: 0");  
 computerScoreLabel.setText("Счет компьютера: 0");  
 drawCardButton.setEnabled(true);  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.invokeLater(() -> new MatematikoGame());  
 }  
}

Класс ScoringLogic

class ScoringLogic {  
 public static int calculateScore(String[] line, boolean isDiagonal) {  
 int[] counts = new int[14]; // Индексы 1-13 для чисел, 0 не используется  
 for (String cell : line) {  
 if (!cell.isEmpty()) {  
 counts[Integer.parseInt(cell)]++;  
 }  
 }  
  
 int maxCount = 0;  
 int secondMaxCount = 0;  
 for (int count : counts) {  
 if (count > maxCount) {  
 secondMaxCount = maxCount;  
 maxCount = count;  
 } else if (count > secondMaxCount) {  
 secondMaxCount = count;  
 }  
 }  
  
 if (maxCount == 5) {  
 return isDiagonal ? 60 : 50;  
 } else if (maxCount == 4) {  
 return isDiagonal ? 170 : 160;  
 } else if (maxCount == 3 && secondMaxCount == 2) {  
 return isDiagonal ? 90 : 80;  
 } else if (maxCount == 3) {  
 return isDiagonal ? 50 : 40;  
 } else if (maxCount == 2 && secondMaxCount == 2) {  
 return isDiagonal ? 30 : 20;  
 } else if (maxCount == 2) {  
 return isDiagonal ? 20 : 10;  
 }  
  
 return 0;  
 }  
  
 public static int calculateTotalScore(String[][] board) {  
 int totalScore = 0;  
  
 // Проверка рядов  
 for (String[] row : board) {  
 totalScore += calculateScore(row, false);  
 }  
  
 // Проверка столбцов  
 for (int col = 0; col < 5; col++) {  
 String[] column = new String[5];  
 for (int row = 0; row < 5; row++) {  
 column[row] = board[row][col];  
 }  
 totalScore += calculateScore(column, false);  
 }  
  
 // Проверка диагоналей  
 String[] mainDiagonal = new String[5];  
 String[] antiDiagonal = new String[5];  
 for (int i = 0; i < 5; i++) {  
 mainDiagonal[i] = board[i][i];  
 antiDiagonal[i] = board[i][4-i];  
 }  
 totalScore += calculateScore(mainDiagonal, true);  
 totalScore += calculateScore(antiDiagonal, true);  
  
 return totalScore;  
 }  
}

Класс ComputerPlayer

class ComputerPlayer {  
 private Random random = new Random();  
  
 public int[] makeMove(String[][] board, int currentNumber) {  
 int bestScore = -1;  
 int[] bestMove = new int[2];  
  
 for (int i = 0; i < 5; i++) {  
 for (int j = 0; j < 5; j++) {  
 if (board[i][j].isEmpty()) {  
 // Пробуем сделать ход  
 board[i][j] = String.valueOf(currentNumber);  
 int score = ScoringLogic.calculateTotalScore(board);  
 // Отменяем ход  
 board[i][j] = "";  
  
 // Если это лучший ход, запоминаем его  
 if (score > bestScore) {  
 bestScore = score;  
 bestMove[0] = i;  
 bestMove[1] = j;  
 }  
 }  
 }  
 }  
  
 // Если все клетки одинаково хороши, выбираем случайную  
 if (bestScore == -1) {  
 do {  
 bestMove[0] = random.nextInt(5);  
 bestMove[1] = random.nextInt(5);  
 } while (!board[bestMove[0]][bestMove[1]].isEmpty());  
 }  
  
 return bestMove;  
 }  
}

Задача 10

section .data

prompt db "Введите число: ", 0

result db "Результат: ", 0

newline db 10, 0

section .bss

num resb 8

output resb 8

section .text

global \_start

\_start:

mov rax, 1

mov rdi, 1

mov rsi, prompt

mov rdx, 16

syscall

mov rax, 0

mov rdi, 0

mov rsi, num

mov rdx, 8

syscall

mov rcx, 0

mov rsi, num

parse\_loop:

movzx rax, byte [rsi]

cmp al, 10

je parse\_done

sub al, '0'

imul rcx, 10

add rcx, rax

inc rsi

jmp parse\_loop

parse\_done:

shr rcx, 1

mov rax, rcx

mov rsi, output

add rsi, 7

mov byte [rsi], 0

convert\_loop:

dec rsi

mov rdx, 0

mov rbx, 10

div rbx

add dl, '0'

mov [rsi], dl

test rax, rax

jnz convert\_loop

mov rax, 1

mov rdi, 1

mov rsi, result

mov rdx, 10

syscall

mov rax, 1

mov rdi, 1

mov rsi, output

mov rdx, 8

syscall

mov rax, 1

mov rdi, 1

mov rsi, newline

mov rdx, 1

syscall

mov rax, 60

xor rdi, rdi

syscall

Задача 11

section .data

prompt\_x db "Введите x: ", 0

prompt\_a db "Введите a: ", 0

result\_msg db "Результат x - a = ", 0

newline db 10, 0

section .bss

x resq 1

a resq 1

input\_buffer resb 16

section .text

global \_start

Psub:

mov rax, [rdi] ; загружаем значение x

sub rax, rsi ; вычитаем a

mov [rdi], rax ; сохраняем результат обратно в x

ret

\_start:

; Запрашиваем x

mov rax, 1

mov rdi, 1

mov rsi, prompt\_x

mov rdx, 13

syscall

; Считываем x

mov rax, 0

mov rdi, 0

mov rsi, input\_buffer

mov rdx, 16

syscall

; Преобразуем строку в число

mov rdi, input\_buffer

call atoi

mov [x], rax

; Запрашиваем a

mov rax, 1

mov rdi, 1

mov rsi, prompt\_a

mov rdx, 13

syscall

; Считываем a

mov rax, 0

mov rdi, 0

mov rsi, input\_buffer

mov rdx, 16

syscall

; Преобразуем строку в число

mov rdi, input\_buffer

call atoi

mov [a], rax

; Вызываем Psub

mov rdi, x

mov rsi, [a]

call Psub

; Выводим результат

mov rax, 1

mov rdi, 1

mov rsi, result\_msg

mov rdx, 21

syscall

mov rdi, [x]

call itoa

mov rax, 1

mov rdi, 1

mov rsi, input\_buffer

mov rdx, 16

syscall

; Выводим новую строку

mov rax, 1

mov rdi, 1

mov rsi, newline

mov rdx, 1

syscall

; Выход из программы

mov rax, 60

xor rdi, rdi

syscall

atoi:

xor rax, rax

.loop:

movzx rcx, byte [rdi]

test rcx, rcx

jz .done

cmp rcx, '0'

jl .done

cmp rcx, '9'

jg .done

sub rcx, '0'

imul rax, 10

add rax, rcx

inc rdi

jmp .loop

.done:

ret

itoa:

mov rax, rdi

mov rdi, input\_buffer

add rdi, 15

mov byte [rdi], 0

mov rbx, 10

.loop:

xor rdx, rdx

div rbx

add dl, '0'

dec rdi

mov [rdi], dl

test rax, rax

jnz .loop

mov rax, rdi

ret