Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа № 4-5

По дисциплине ГИИС

Тема: «Разработка приложения «Пятнашки»»

Выполнил:

Студент 4 курса

Группы ИИ-16 (2)

Пешко А.С.

Проверил:

Михно Е.В.

Брест, 2021

**Цель работы:**  Изучить базовые компоненты средства разработки графического интерфейса.

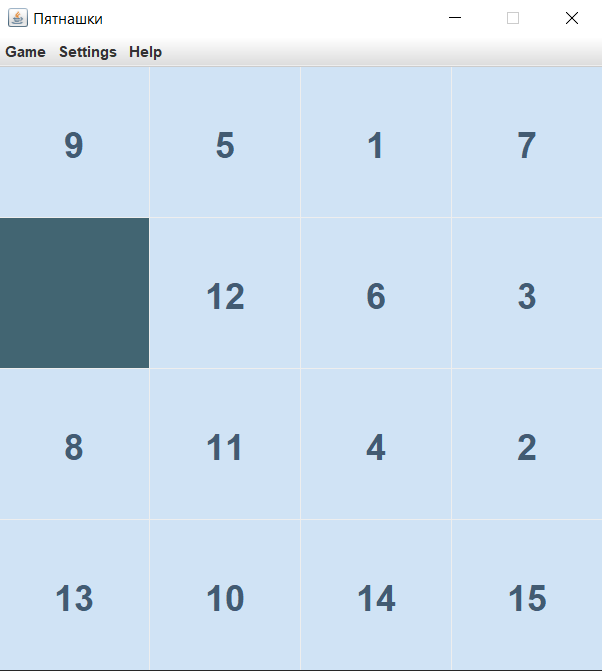
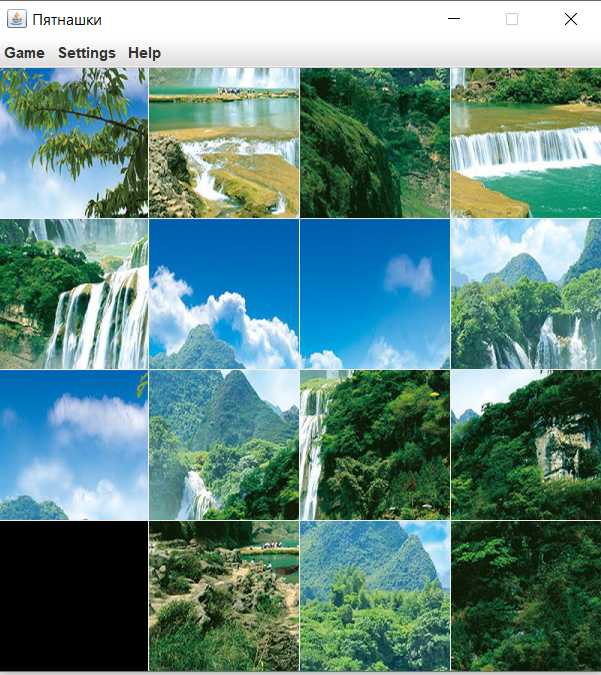
**Ход работы:**

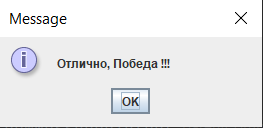
**Задание:** Выполнить последовательно разработку приложения «Пятнашки».

В ходе лабораторной работы разрабатывается функциональность для реализации логики игры. Конечным результатом работы является приложение, которое имеет следующую функциональность:

1. Графическая форма с костяшками обозначенными цифрами.
2. Управление движением костяшек при помощи мыши и клавиатуры.
3. Сообщение о победе, если костяшки упорядочены.  
     
   public class Puzzle extends JPanel {  
     
    private JLabel[][] squares;  
    private JLabel[][] squaresEtalon;  
    private int row;  
    private int col;  
    public int sizeGame = 4;  
    public int sizeBlock = 150;  
    MouseHandler handler = new MouseHandler();  
    public boolean win = false;  
    public BufferedImage img;  
    public BufferedImage[][] images;  
     
    private void initialize() {  
    setSize(sizeBlock \* sizeBlock, sizeBlock \* sizeBlock);  
    row = sizeGame - 1;  
    col = sizeGame - 1;  
    squaresEtalon = new JLabel[sizeGame][sizeGame];  
    squares = new JLabel[sizeGame][sizeGame];  
    setLayout(new GridLayout(sizeGame, sizeGame, 1, 1));  
    }  
     
    private void createPuzzle() {  
    for (int i = 0; i < sizeGame; i++) {  
    for (int j = 0; j < sizeGame; j++) {  
    add(squares[i][j]);  
    }  
    }  
    }  
     
    public Puzzle() {  
    initialize();  
    createNumberGame();  
    shuffle();  
    createPuzzle();  
    }  
     
    public Puzzle(String src) {  
    initialize();  
    createImageGame(src);  
    shuffle();  
    createPuzzle();  
    }  
     
    private void createNumberGame() {  
    JLabel sq;  
    for (int i = 0; i < sizeGame; i++) {  
    for (int j = 0; j < sizeGame; j++) {  
    if (i != sizeGame - 1 || j != sizeGame - 1) {  
    sq = new JLabel("<html>" +  
    "<font size='18' color='#425B72'><strong>" + (i \* sizeGame + j + 1) + "</strong></font>" +  
    "</html>");  
    sq.setVerticalAlignment(JLabel.CENTER);  
    sq.setHorizontalAlignment(JLabel.CENTER);  
    sq.setBackground(new Color(208,227,245));  
    } else {  
    sq = new JLabel();  
    sq.setBackground(new Color(66, 101, 114));  
    }  
    sq.setPreferredSize(new Dimension(sizeBlock, sizeBlock));  
    sq.setOpaque(true);  
    squaresEtalon[i][j] = sq;  
    sq.addMouseListener(handler);  
    squares[i][j] = sq;  
    }  
    }  
    }  
     
    private void createImageGame(String src) {  
    try {  
    img = ImageIO.read(new File(src));  
    int width = img.getWidth() / sizeGame;  
    int height = img.getHeight() / sizeGame;  
    images = new BufferedImage[sizeGame][sizeGame];  
    JLabel sq;  
    for (int i = 0; i < sizeGame; i++) {  
    for (int j = 0; j < sizeGame; j++) {  
    sq = new JLabel();  
    images[i][j] = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_ARGB);  
    for (int k = 0 + i \* height, q = 0; k < height + i \* height; k++, q++) {  
    for (int l = 0 + j \* width, w = 0; l < width + j \* width; l++, w++) {  
    if (i != sizeGame - 1 || j != sizeGame - 1) {  
    images[i][j].setRGB(w, q, img.getRGB(l, k));  
    }  
    else {  
    images[i][j].setRGB(w, q, Color.BLACK.getRGB());  
    }  
    }  
    }  
    sq.setSize(new Dimension(sizeBlock, sizeBlock));  
    sq.setIcon(new ImageIcon(resize(images[i][j], sq)));  
     
    squaresEtalon[i][j] = sq;  
    sq.addMouseListener(handler);  
    squares[i][j] = sq;  
    }  
    }  
    } catch (IOException e) {  
    e.printStackTrace();  
    }  
    }  
     
    private void shuffle() {  
    Random rnd = new Random();  
    String[] moves = {"up", "down", "left", "right"};  
    for (int i = 0; i < 100; i++) {  
    move(moves[rnd.nextInt(4)]);  
    }  
    }  
     
    public void checkWin() {  
    for (int i = 0; i < sizeGame; i++) {  
    for (int j = 0; j < sizeGame; j++) {  
    if (!squaresEtalon[i][j].equals(squares[i][j]))  
    return;  
    }  
    }  
    win = true;  
    for (int i = 0; i < sizeGame; i++)  
    for (int j = 0; j < sizeGame; j++) {  
    squares[i][j].removeMouseListener(handler);  
    }  
    JOptionPane.showMessageDialog(Puzzle.this, "You Win!!!");  
     
    }  
     
    public BufferedImage resize(BufferedImage img, JLabel panel) {  
    Image tmp = img.getScaledInstance(panel.getWidth(), panel.getHeight(), Image.SCALE\_SMOOTH);  
    img = new BufferedImage(panel.getWidth(), panel.getHeight(), BufferedImage.TYPE\_INT\_ARGB);  
     
    Graphics2D g2d = img.createGraphics();  
    g2d.drawImage(tmp, 0, 0, null);  
    g2d.dispose();  
    return img;  
    }  
     
    public void move(String key) {  
    switch (key) {  
    case "up": {  
    if (row == sizeGame - 1)  
    return;  
    else {  
    row++;  
    swap(row, col, row - 1, col);  
    break;  
    }  
    }  
    case "down": {  
    if (row == 0)  
    return;  
    else {  
    row--;  
    swap(row, col, row + 1, col);  
    break;  
    }  
    }  
    case "left": {  
    if (col == sizeGame - 1)  
    return;  
    else {  
    col++;  
    swap(row, col, row, col - 1);  
    break;  
    }  
    }  
    case "right": {  
    if (col == 0)  
    return;  
    else {  
    col--;  
    swap(row, col, row, col + 1);  
    break;  
    }  
    }  
    default:  
    return;  
    }  
     
    paint();  
    }  
     
    private void swap(Integer rowNew, Integer colNew, Integer rowOld, Integer colOld) {  
    JLabel t = squares[rowNew][colNew];  
    squares[rowNew][colNew] = squares[rowOld][colOld];  
    squares[rowOld][colOld] = t;  
    }  
     
    private void paint() {  
    removeAll();  
    for (int i = 0; i < sizeGame; i++)  
    for (int j = 0; j < sizeGame; j++)  
    add(squares[i][j]);  
    revalidate();  
    repaint();  
    }  
     
    private class MouseHandler implements MouseListener {  
     
    private void handleClick(Integer i, Integer j) {  
    if (row == i && col == j + 1) {  
    col--;  
    swap(row, col + 1, i, j);  
    }  
    if (row == i && col == j - 1) {  
    col++;  
    swap(row, col - 1, i, j);  
    } else if (row == i + 1 && col == j) {  
    row--;  
    swap(row + 1, col, i, j);  
    } else if (row == i - 1 && col == j) {  
    row++;  
    swap(row - 1, col, i, j);  
    } else return;  
    checkWin();  
    }  
     
    @Override  
    public void mouseClicked(MouseEvent e) {  
    Object source = e.getSource();  
    for (int i = 0; i < sizeGame; i++)  
    for (int j = 0; j < sizeGame; j++)  
    if (source.equals(squares[i][j])) {  
    handleClick(i, j);  
    paint();  
    return;  
    }  
    }  
    }  
   }

**Результат работы:**



**Вывод:** изучили базовые компоненты средства разработки графического интерфейса.