

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ  
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №8

Специальность ПО5

Выполнил:  
А.А. Игнатюк,  
студент группы ПО-5

Проверил:  
А.А. Крощенко,  
ст. преп. кафедры ИИТ,  
«\_\_» \_\_\_\_\_ 2022 г.

Брест 2022

**Цель работы:** Приобрести навыки написания простого оконного многопоточного приложения с использованием Java API.

## Вариант 5.

### Задание.

Разработать оконное приложение с использованием Java API, использующее один вспомогательный поток, вычисляющий заданную сумму и выполняющий вывод результата вычисления (как конечный, так и промежуточные) в любой визуальный компонент. Все исходные данные вводятся в соответствующие визуальные компоненты.

В программе должны быть предусмотрены функции приостановки, возобновления и полной остановки выполнения потока с выводом соответствующего сообщения. В случае быстрого выполнения потока и, как следствие, невозможности демонстрации функций приостановки, продумать искусственное торможение потока для достижения заданных целей. Обработать исключения.

$$5) \quad \sum_{k=1}^n \frac{1}{(2k-1)(2k+1)} = \frac{1}{1 \cdot 3} + \frac{1}{3 \cdot 5} + \dots + \frac{1}{(2n-1)(2n+1)}$$

Рисунок 1 - Исходная формула.

**Спецификация ввода:**

**Спецификация вывода:**

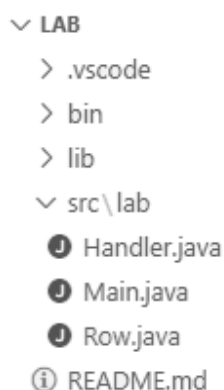


Рисунок 2 - Структура проекта.

**Код программы и результаты тестирования:**

Row.java 1 X

src > lab > Row.java > ...

```
1  package lab;
2
3  import javafx.concurrent.Task;
4
5  public final class Row extends Task {
6      private volatile boolean running;
7      private final int n;
8
9      public Row(final int n) {
10         this.running = true;
11         this.n = n;
12     }
13
14     public final void start() {
15         this.running = true;
16     }
17
18     public final void stop() {
19         this.running = false;
20     }
21
22     @Override
23     protected final Object call() {
24         try {
25             int previousOdd = 1, nextOdd = previousOdd + 2;
26             double result = 0.0;
27
28             for (int i = 0; i < n; ++i) {
29                 while (!running) {
30                     ;
31                 }
32
33                 result += 1.0 / (previousOdd * nextOdd);
34                 previousOdd = nextOdd;
35                 nextOdd += 2;
36
37                 updateMessage("Result for n = " + String.valueOf(i + 1) + " : " + String.valueOf(result));
38                 Thread.sleep(100);
39             }
40         } catch (final InterruptedException exception) {
41             System.out.println(exception.getMessage());
42         }
43
44         return null;
45     }
46 }
```

Рисунок 3 - Исходный код файла Row.java.

① Handler.java ×

src > lab > ① Handler.java > ...

```
1  package lab;
2
3  public final class Handler {
4      private Row row;
5      private Thread thread;
6
7      private final boolean isSet() {
8          return this.row != null && this.thread != null;
9      }
10
11     public Handler() {
12         this.row = null;
13         this.thread = null;
14     }
15
16     public final void start() {
17         if (this.isSet()) {
18             this.thread.start();
19         }
20     }
21
22     public final void stop() {
23         if (this.isSet()) {
24             this.row.stop();
25         }
26     }
27
28     public final void resume() {
29         if (this.isSet()) {
30             this.row.start();
31         }
32     }
33
34     public final void interrupt() {
35         if (this.isSet()) {
36             this.thread.interrupt();
37         }
38     }
39
40     public final void setRow(final Row row) {
41         if (this.isSet()) {
42             this.interrupt();
43         }
44
45         this.row = row;
46         this.thread = new Thread(this.row);
47     }
48 }
49
```

Рисунок 4 - Исходный код файла Handler.java.

Main.java X

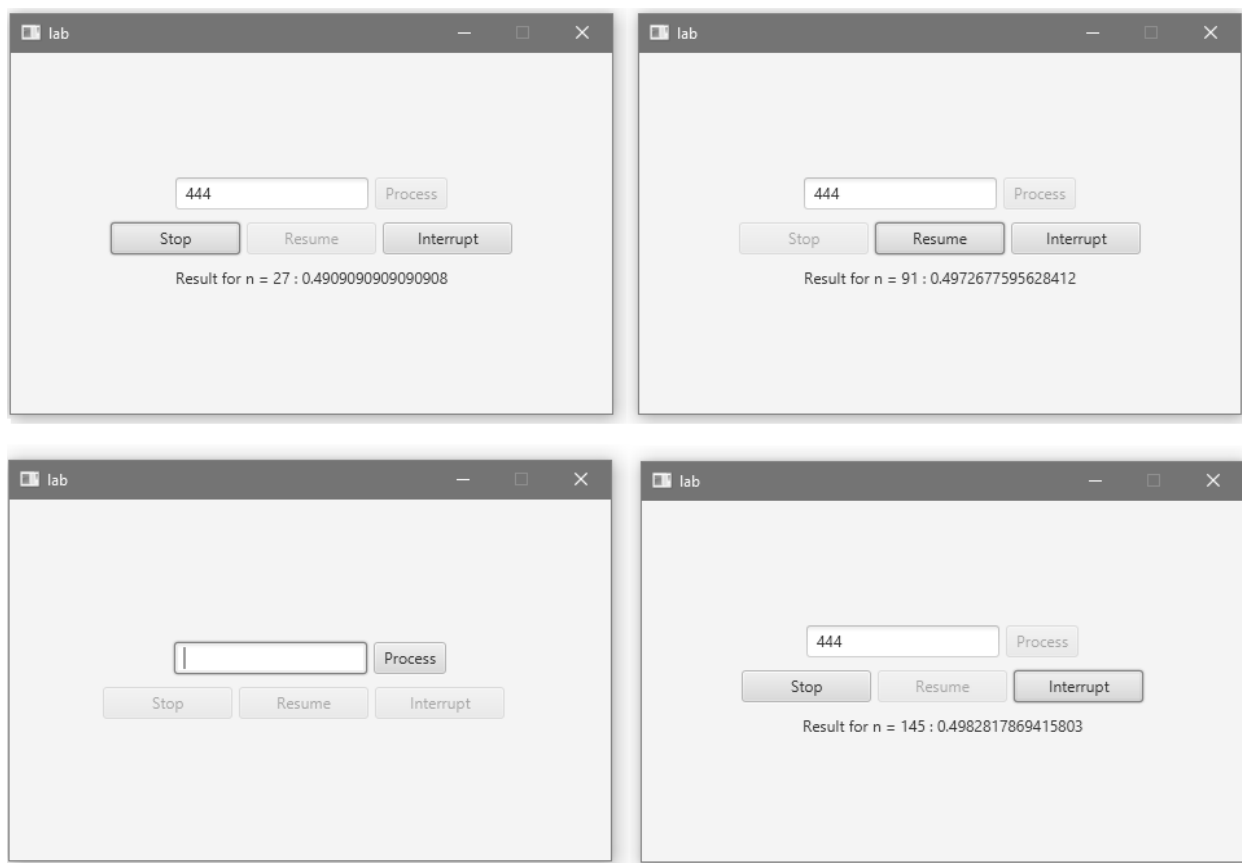
src > lab > Main.java > ...

```
1  package lab;
2
3  import javafx.scene.layout.FlowPane;
4  import javafx.scene.Scene;
5  import javafx.scene.control.TextField;
6  import javafx.scene.control.Label;
7  import javafx.scene.layout.HBox;
8  import javafx.scene.control.Button;
9  import javafx.application.Application;
10 import javafx.geometry.Pos;
11 import javafx.geometry.Orientation;
12 import javafx.stage.Stage;
13
14 public final class Main extends Application {
15     private FlowPane pane;
16     private Scene scene;
17
18     private TextField input;
19     private Label output;
20
21     private HBox threadControls;
22     private HBox processControls;
23     private HBox outputBox;
24
25     private Button stopThread;
26     private Button resumeThread;
27     private Button interruptThread;
28     private Button process;
29
30     private Row row;
31     private Handler handler;
32
33     public final static void main(final String[] args) {
34         Application.launch(args);
35     }
36
37     @Override
38     public final void start(final Stage stage) {
39         initThreadControls();
40         initProcessControls();
41
42         this.pane = new FlowPane(this.processControls, this.threadControls, this.outputBox);
43         this.pane.setAlignment(Pos.CENTER);
44         this.pane.setOrientation(Orientation.HORIZONTAL);
45         this.pane.setVgap(10);
46         this.pane.setHgap(10);
47
48         this.scene = new Scene(this.pane);
49         this.handler = new Handler();
50
51         stage.setTitle("lab");
52         stage.setResizable(false);
53         stage.setMinHeight(320);
54         stage.setMinWidth(480);
55         stage.setScene(scene);
56         stage.show();
57     }
58
```

Рисунок 5 - Исходный код файла Main.java.

## Продолжение рисунка 5.

```
59 private final void initThreadControls() {
60     this.stopThread = new Button("Stop");
61     this.stopThread.setPrefWidth(100);
62     this.stopThread.setDisable(true);
63
64     this.stopThread.setOnAction(event -> {
65         this.handler.stop();
66         this.stopThread.setDisable(true);
67         this.resumeThread.setDisable(false);
68     });
69
70     this.resumeThread = new Button("Resume");
71     this.resumeThread.setPrefWidth(100);
72     this.resumeThread.setDisable(true);
73
74     this.resumeThread.setOnAction(event -> {
75         this.handler.resume();
76         this.stopThread.setDisable(false);
77         this.resumeThread.setDisable(true);
78     });
79
80     this.interruptThread = new Button("Interrupt");
81     this.interruptThread.setPrefWidth(100);
82     this.interruptThread.setDisable(true);
83
84     this.interruptThread.setOnAction(event -> {
85         this.handler.interrupt();
86         this.stopThread.setDisable(true);
87         this.resumeThread.setDisable(true);
88         this.interruptThread.setDisable(true);
89         this.process.setDisable(false);
90     });
91
92     this.threadControls = new HBox(5, this.stopThread, this.resumeThread, this.interruptThread);
93 }
94
95 private final void initProcessControls() {
96     this.input = new TextField();
97     this.output = new Label();
98
99     this.process = new Button("Process");
100    this.process.setOnAction(event -> {
101        try {
102            int n = Integer.parseInt(this.input.getText());
103
104            this.row = new Row(n);
105            this.handler.setRow(this.row);
106            this.output.textProperty().bind(this.row.messageProperty());
107            this.handler.start();
108
109            this.stopThread.setDisable(false);
110            this.resumeThread.setDisable(true);
111            this.interruptThread.setDisable(false);
112            this.process.setDisable(true);
113        } catch (final NumberFormatException exception) {
114            System.out.println(exception.getMessage());
115        }
116    });
117
118    this.processControls = new HBox(5, this.input, this.process);
119    this.outputBox = new HBox(5, this.output);
120 }
121 }
122
```



**Рисунок 6 - Результат выполнения программы.**

**Вывод:** Приобрел навыки написания простого оконного многопоточного приложения с использованием Java API.