**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,**

**СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

25.11 2024 г**. №** \_\_\_\_\_\_\_\_\_\_

Факультет Кибербезопасность

Кафедра Защищенных систем связи

# Отчет по лабораторной работе №5

«Обработка исключений»

**по дисциплине «Разработка защищённого программного обеспечения»**

Выполнил студент гр. ИКБ-31  
Вавилин Сергей Максимович

Принял ст.преп.

Цветков А.Ю.

# Основное задание

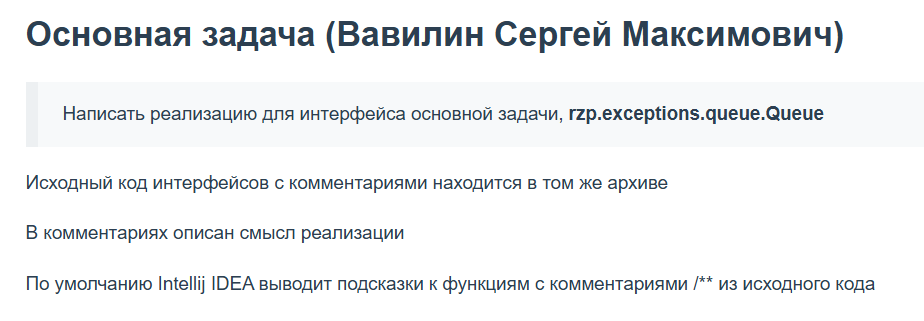
**Цель лабораторной работы**

1. Освоить механизмы обработки исключений в Java

**Используемое программное обеспечение**

1. JDK 1.5 и выше
2. Интегрированная среда разработки на Java

**Вариант задания**

****

**Листинг программы**

*package rzp.exceptions.queue;*

*public class queueimpl implements Queue {*

*public int[] items; // массив для хранения элементов*

*public int maxSize; // максимальный размер очереди*

*private int front; // указатель на начало очереди*

*public int rear; // указатель на конец очереди*

*public int count; // количество элементов в очереди*

*// Реализация методов будет здесь*

*@Override*

*public void init(int maxSize) throws QueueException {*

*if (items != null) {*

*throw new QueueException("queue is already initialized");*

*}*

*this.maxSize = maxSize;*

*this.items = new int[maxSize];*

*this.front = 0;*

*this.rear = -1;*

*this.count = 0;*

*}*

*@Override*

*public boolean isEmpty() {*

*return count == 0;*

*}*

*@Override*

*public boolean isFull() {*

*return count == maxSize;*

*}*

*@Override*

*public int size() {*

*return count;*

*}*

*@Override*

*public boolean contains(int item){*

*if (isEmpty()) {*

*return false;*

*}*

*for (int i = 0 ; i < count ; i++ ) {*

*if (items[(front + i) % maxSize] == item) {*

*return true;*

*}*

*}*

*return false;*

*}*

*// with exceptions*

*@Override*

*public int peek() throws QueueException {*

*if (isEmpty()) {*

*throw new QueueException("queue is empty");*

*}*

*return items[front];*

*}*

*@Override*

*public int poll() throws QueueException {*

*if (isEmpty()) {*

*throw new QueueException("queue is empty");*

*}*

*int item = items[front];*

*front = (front + 1) % maxSize;*

*count--;*

*return item;*

*}*

*@Override*

*public void offer(int item) throws QueueException {*

*if (isFull()) {*

*throw new QueueException("queue is full");*

*}*

*rear = (rear + 1) % maxSize;*

*items[rear] = item;*

*count++;*

*}*

*}*

**Вывод**

Были получены навыки работы с исключениями, вылавливать ошибки и логировать их.

# Дополнительное задание

**Цель лабораторной работы**

1. Освоить механизмы обработки исключений в Java

**Используемое программное обеспечение**

1. JDK 1.5 и выше
2. Интегрированная среда разработки на Java

**Вариант задания**

****

**Листинг программы**

*package rzp.exceptions.queue;*

*public class queueExtraImpl extends queueimpl implements QueueExtra {*

*@Override*

*public void offer(int... items) throws QueueException, IllegalStateException {*

*if (this.items == null ) {*

*throw new IllegalStateException("Queue is not initialized");*

*}*

*if (items == null || items.length == 0) {*

*throw new QueueException("no items to add");*

*}*

*if (count + items.length > maxSize) {*

*throw new QueueException("not enough space");*

*}*

*for (int item : items) {*

*if (isFull()){*

*throw new QueueException("queue is full");*

*}*

*rear = (rear + 1) % maxSize;*

*this.items[rear] = item;*

*count++;*

*}*

*}*

*}*

**Вывод**

Изучил работу с исключениями научился отлавливать исключения и писать лог для исключений.