Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших данных в системах поддержки принятия решений.**

О Т Ч Е Т

по лабораторной работе № 8

Вариант № 11

Название: Потоки

Дисциплина: Языки программирования для работы с большими данными

Студент ИУ6-23М (Группа) _____ (Подпись, дата) С.В.Мельников (И.О. Фамилия)

Преподаватель _____ (Подпись, дата) П.В. Степанов (И.О. Фамилия)

Москва, 2023

Цель работы

Изучить работу потоков в языке программирования Java.

Задание 1 (Вариант 1, Задание 2):

Реализовать многопоточное приложение “Робот”. Надо написать робота, который умеет ходить. За движение каждой его ноги отвечает отдельный поток. Шаг выражается в выводе в консоль LEFT или RIGHT

Листинг программы:

Код класса Robot:

```
public class Robot {

    private final Thread left_thread;
    private final Thread right_thread;
    class Leg implements Runnable {
        private final String name;
        Leg(String name) {
            this.name = name;
        }
        @Override
        public void run() {
            while(true) {
                step();
                try {
                    Thread.sleep(1000);
                } catch (InterruptedException e) {
                    System.out.println("Leg " + this.name + " stopped");
                    break;
                }
            }
        }
        private void step() {
            System.out.println(name);
        }
    }

    public Robot() {
        Leg left_leg = new Leg("LEFT");
        Leg right_leg = new Leg("RIGHT");
        left_thread = new Thread(left_leg);
        right_thread = new Thread(right_leg);
    }

    void run() {
        left_thread.start();
        try {
            Thread.sleep(500);
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
        right_thread.start();
    }

    void stop() {
```

```
        left_thread.interrupt();
        right_thread.interrupt();
    }

    public static void main(String[] args) {
        Robot robot = new Robot();
        robot.run();
        try {
            Thread.sleep(5000);
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
        robot.stop();
    }
}
```

Работа программы представлена на рисунке 1.

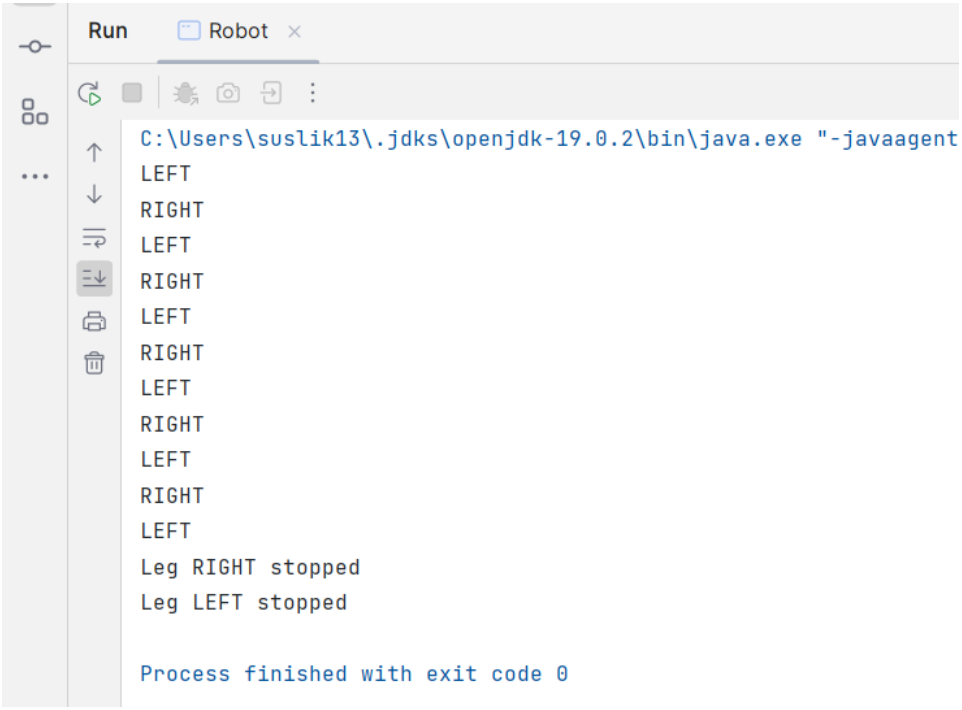

Рисунок 1 – Работа программы Robot.java

Задание 2 (Вариант 1, Задание 3):

Реализовать многопоточное приложение “Магазин”. Вся цепочка: производитель-магазин-покупатель. Пока производитель не поставит на склад продукт, покупатель не может его забрать. Реализовать приход товара от производителя в магазин случайным числом. В том случае, если товара в магазине не хватает– вывести сообщение.

Листинг программы:

Код класса ShopSystem:

```
public class ShopSystem {
    private final Producer producer;
    private final Shop shop;
    private final Customer customer;
    private static final Queue<Integer> store = new PriorityQueue<>();
    private static final Queue<Integer> shelving = new PriorityQueue<>();


    class Producer implements Runnable {
        private final String name;

        public Producer(String name) {
            this.name = name;
        }

        @Override
        public void run() {
            double probability;
            for (int i = 0; i < 10; i++) {
                try {
                    long secs = (long)(Math.random() * 3) + 1;
                    Thread.sleep(500 * secs);
                } catch (InterruptedException e) {
                    throw new RuntimeException(e);
                }
                probability = Math.random();
                if (probability > 0.5) {
                    int product_id = (int)(Math.random() * 90) + 10;
                    System.out.printf("%10s: Отправлен товар с id: %d\n",
this.name, product_id);
                    store.add(product_id);
                }
            }
            System.out.printf("%10s: Конец отправки\n", this.name);
            store.add(-1);
        }
    }
    class Shop implements Runnable {
        private final String name;

        public Shop(String name) {
            this.name = name;
        }

        @Override
        public void run() {
            while (true) {
                if (store.isEmpty()) {
                    try {
                        Thread.sleep(10);
                    } catch (InterruptedException e) {
                        throw new RuntimeException(e);
                    }
                } else {
                    int product_id = store.poll();
                    if (product_id == -1) {
                        break;
                    }
                    try {
                        long secs = (long)(Math.random() * 3) + 1;
                        Thread.sleep(500 * secs);
```

```
                    } catch (InterruptedException e) {
                        throw new RuntimeException(e);
                    }
                    System.out.printf("%10s: Получен товар с id: %d\n", this.name,
product_id);
                    shelving.add(product_id);
                }
            }
            System.out.printf("%10s: Магазин закрывается\n", this.name);
            shelving.add(-1);
        }
    }
    class Customer implements Runnable {
        private final String name;

        public Customer(String name) {
            this.name = name;
        }

        @Override
        public void run() {
            boolean waiting = true;
            while (true) {
                if (shelving.isEmpty()) {
                    try {
                        Thread.sleep(500);
                        if (waiting) {
                            System.out.printf("%10s: Ждет товар\n", this.name);
                            waiting = false;
                        }
                    } catch (InterruptedException e) {
                        throw new RuntimeException(e);
                    }
                } else {
                    int product_id = shelving.poll();
                    if (product_id == -1) {
                        break;
                    }
                    try {
                        long secs = (long)(Math.random() * 3) + 1;
                        Thread.sleep(500 * secs);
                    } catch (InterruptedException e) {
                        throw new RuntimeException(e);
                    }
                    System.out.printf("%10s: Купил товар с id: %d\n", this.name,
product_id);
                    waiting = true;
                }
            }
            System.out.printf("%10s: Клиент ушел\n", this.name);
        }
    }

    public ShopSystem() {
        producer = new Producer("Хлеб Завод");
        shop = new Shop("Пятерочка");
        customer = new Customer("Серёжа");
    }

    void run() {
        new Thread(producer).start();
        new Thread(shop).start();
        new Thread(customer).start();
    }
    public static void main(String []args) throws InterruptedException {
```

```
        ShopSystem shop_system = new ShopSystem();
        shop_system.run();
    }
}
```

Работа программы представлена на рисунке 2.

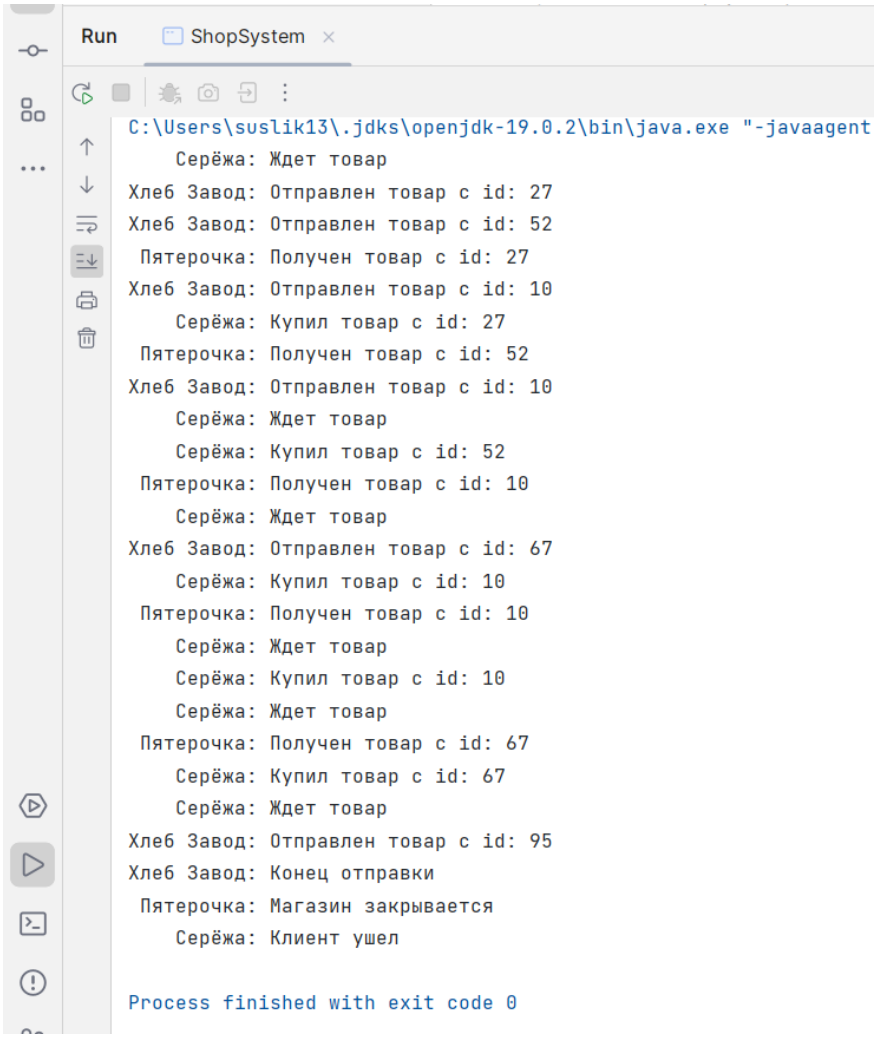

Рисунок 2 – Работа программы ShopSystem.java

Вывод

В ходе выполнения лабораторной работы были получены навыки для работы с потоками в языке программирования Java.