



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 **09.04.01/07 Интеллектуальные системы анализа,
обработки и интерпретации больших данных**

О Т Ч Е Т

по лабораторной работе № 8

Вариант 12

Название: Потоки в Java

Дисциплина: Языки программирования для работы с большими данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

П.А. Мартынюк

(И.О. Фамилия)

Преподаватель

П.В. Степанов

(Подпись, дата)

(И.О. Фамилия)

Москва, 2022

Цель работы:

Получение навыков работы с потоками в Java.

Выполнение:

Задание:

1. Реализовать многопоточное приложение “Банк”. Имеется банковский счет. Сделать синхронным пополнение и снятие денежных средств на счет/со счет случайной суммой. При каждой операции (пополнения или снятие) вывести текущий баланс счета. В том случае, если денежных средств недостаточно – вывести сообщение.
2. 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 выполнения подзадачи 1 (файл MainBank.java)

```
package task1;

import java.util.Random;

public class MainBank {

    public volatile static int account;
    public volatile static Random random;

    public static void main(String[] args) throws InterruptedException {
        random = new Random();
        account = 0;

        new Thread(income).start();
        Thread.sleep(2000);
        new Thread(outcome).start();
    }

    static Runnable income = new Runnable() {
        @Override
        public void run() {
            while (true) {
                try {
                    Thread.sleep(5000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                int receipt = random.nextInt(100, 500);
                account += receipt;
                System.out.print("New income: ");
                System.out.println(receipt);
                System.out.print("Account: ");
            }
        }
    }
}
```

```

        System.out.println(account);
        System.out.println("-----");
    }
}

};

static Runnable outcome = new Runnable() {
    @Override
    public void run() {
        while (true) {
            try {
                Thread.sleep(5000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            int withdraw = random.nextInt(200, 400);
            if (withdraw <= account) {
                account -= withdraw;
                System.out.print("New outcome: ");
                System.out.println(withdraw);
                System.out.print("Account: ");
                System.out.println(account);
                System.out.println("-----");
            } else {
                System.out.print("Tried to write-off: ");
                System.out.println(withdraw);
                System.out.print("Account: ");
                System.out.println(account);
                System.out.println("Not enough money!");
                System.out.println("-----");
            }
        }
    }
};
}

```

```

MainBank x
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
2021.3.2\lib\idea_rt.jar=65032:C:\Program Files\JetBrains\IntelliJ IDEA 2021.3.2\bin" -Dfile
.encoding=UTF-8 -classpath C:\Users\polin\IdeaProjects\JavaMagisterCourse\out\production\Lab8
task1.MainBank
New income: 257
Account: 257
-----
New outcome: 231
Account: 26
-----
New income: 329
Account: 355
-----
Tried to write-off: 359
Account: 355
Not enough money!
-----
New income: 394
Account: 749
-----
New outcome: 339
Account: 410
-----
|

```

Рисунок 1 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл MainShop.java)

```
package task1;

import java.util.Random;

public class MainShop {

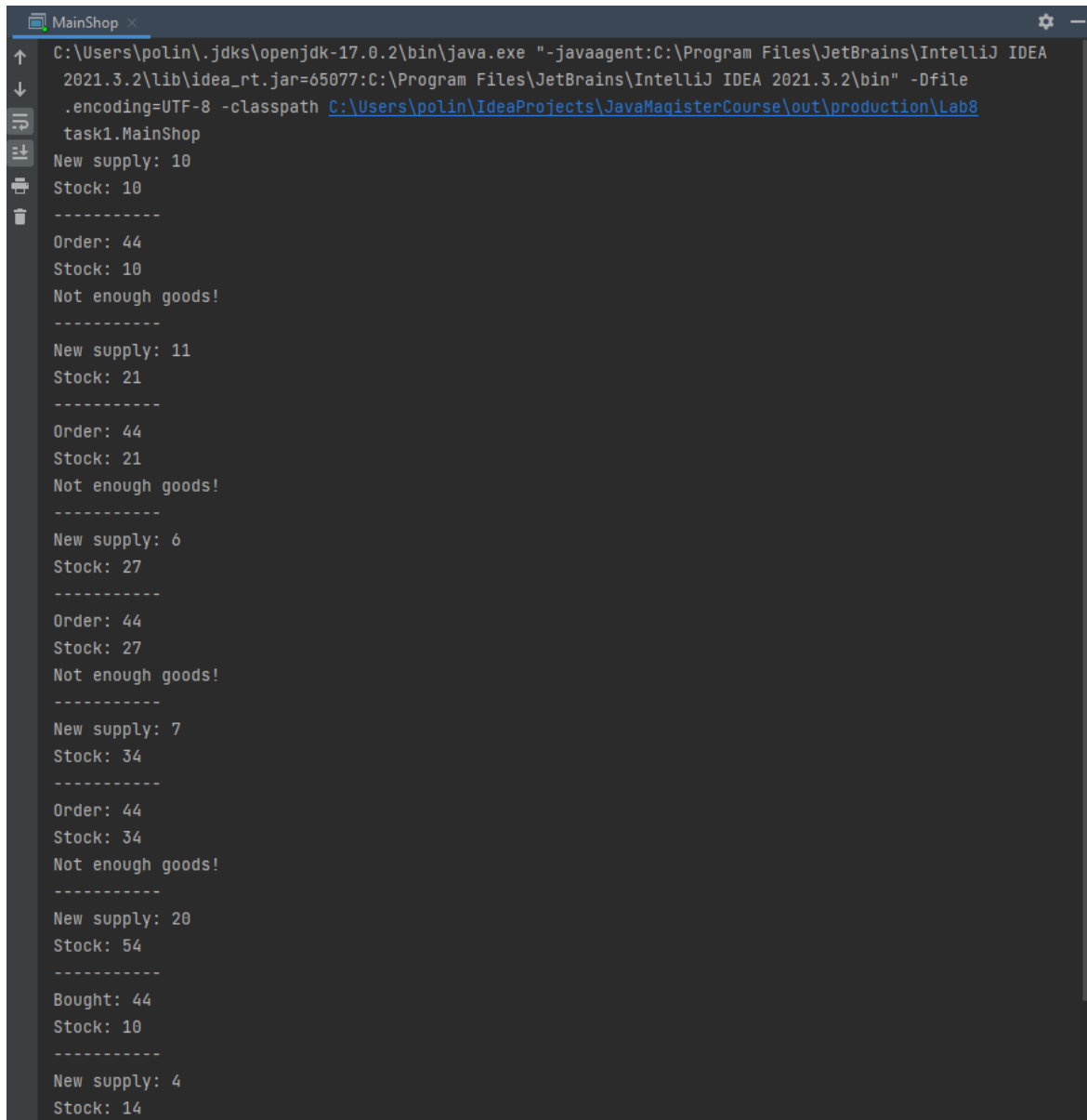
    public volatile static boolean supply_done;
    public volatile static int stock;
    public volatile static Random random;

    public static void main(String[] args) {
        random = new Random();
        stock = 0;
        supply_done = false;

        new Thread(supplier).start();
        new Thread(customer).start();
    }

    static Runnable supplier = new Runnable() {
        @Override
        public void run() {
            while (true) {
                try {
                    Thread.sleep(5000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                int goods = random.nextInt(1, 25);
                stock += goods;
                System.out.print("New supply: ");
                System.out.println(goods);
                System.out.print("Stock: ");
                System.out.println(stock);
                System.out.println("-----");
                supply_done = true;
            }
        }
    };

    static Runnable customer = new Runnable() {
        @Override
        public void run() {
            int order = random.nextInt(25, 50);
            while (true) {
                while (!supply_done) {
                    // беск цикл
                }
                if (order <= stock) {
                    stock -= order;
                    System.out.print("Bought: ");
                    System.out.println(order);
                    System.out.print("Stock: ");
                    System.out.println(stock);
                    System.out.println("-----");
                    order = random.nextInt(25, 50);
                } else {
                    System.out.print("Order: ");
                    System.out.println(order);
                    System.out.print("Stock: ");
                    System.out.println(stock);
                    System.out.println("Not enough goods!");
                    System.out.println("-----");
                }
                supply_done = false;
            }
        }
    };
}
```



```
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.3.2\lib\idea_rt.jar=65077:C:\Program Files\JetBrains\IntelliJ IDEA 2021.3.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\polin\IdeaProjects\JavaMagisterCourse\out\production\Lab8 task1.MainShop
New supply: 10
Stock: 10
-----
Order: 44
Stock: 10
Not enough goods!
-----
New supply: 11
Stock: 21
-----
Order: 44
Stock: 21
Not enough goods!
-----
New supply: 6
Stock: 27
-----
Order: 44
Stock: 27
Not enough goods!
-----
New supply: 7
Stock: 34
-----
Order: 44
Stock: 34
Not enough goods!
-----
New supply: 20
Stock: 54
-----
Bought: 44
Stock: 10
-----
New supply: 4
Stock: 14
```

Рисунок 2 - Результат выполнения кода решения подзадачи 2

Ссылка на программное решение:

Программное решение представлено в репозитории распределённой системы управления версиями Git:

<https://github.com/Owlfeather/JavaMagisterCourse/tree/main/Lab8/src>

Вывод:

При выполнении лабораторной работы были получены навыки работы с потоками в Java.