|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **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**

**Отчет**

**по лабораторной работе № 8**

**Название:** Потоки

**Дисциплина:** Языки программирования для работы с большими данными

Студент гр. ИУ6-23М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_** Г. Е. Горский **\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**П.В. Степанов **\_\_**

(Подпись, дата) (И.О. Фамилия)

Москва, 2023

**Цель:** ознакомиться с базовыми принципами языка Java для работы с большими данными

**Вариант 1:**

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

**Решение:**

public class Main {  
public static void main(String[] args) throws InterruptedException {  
 final Lock lock = new ReentrantLock(true);  
 Bank bank = new Bank();  
 System.out.println("счет создан");  
 boolean active = true;  
 Random random = new Random();  
 Thread first = new Thread(new Runnable()  
 {  
 public void run()  
 {  
 while (active) {  
 lock.lock();  
 try {  
 Thread.sleep(1000);  
 bank.put(random.nextInt(0,100));  
 } catch (InterruptedException e) {  
 }  
 lock.unlock();  
 }  
 }  
 });  
 Thread second = new Thread(new Runnable()  
 {  
 public void run()  
 { while (active) {  
 lock.lock();  
 try {  
 Thread.sleep(1000);  
 bank.output(random.nextInt(0,100));  
 } catch (InterruptedException e) {  
 }  
 lock.unlock();  
 }  
 }  
 });  
 first.start();  
 second.start();  
 //first.join();  
 second.join();  
 }  
}  
class Bank{  
 private float summ;  
 public Bank(float summ) {  
 this.summ = summ;  
 }  
 public Bank() {  
 this.summ = 0;  
 }  
 public float getSumm() {  
 return summ;  
 }  
 public void setSumm(float summ) {  
 this.summ += summ;  
 }  
 public void put(float summ){  
 setSumm(summ);  
 System.out.println("summ=" + this.getSumm());  
 }  
 public void output(float summ){  
 if (this.getSumm()-summ>=0){  
 setSumm(-summ);  
 System.out.println("summ=" + this.getSumm());  
 }  
 else System.out.println("недостаточно денег");  
 }  
}

Результат выполнения представлен на рисунке 1.

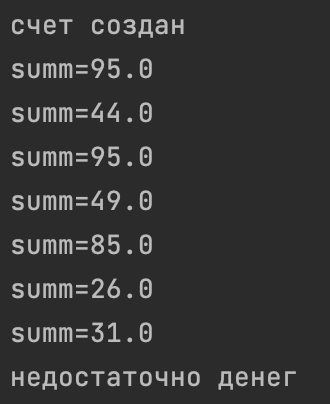
****

Рисунок 1 — пример выполнения

public class Main {  
 public static void main(String[] args) {  
 System.out.println("Hello world!");  
 Object lock = new Object();  
 Robot robot =new Robot("Left",lock);  
 Robot robot1 = new Robot("Right",lock);  
 Thread leftLegThread = new Thread(robot);  
 Thread rightLegThread = new Thread(robot1);  
 leftLegThread.start();  
 rightLegThread.start();  
 }  
}  
class Robot implements Runnable{  
 private String Leg;  
 private final Object lock;  
 public void setLeg(String leg) {  
 Leg = leg;  
 }  
 public String getLeg() {  
 return Leg;  
 }  
 public Robot(String leg, Object lock) {  
 this.lock = lock;  
 this.setLeg(leg);  
 }  
 public void run() {  
 while (true) synchronized (lock) {  
 System.out.println(this.getLeg() + " step");  
 lock.notify();  
 try {  
 lock.wait(10000);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

Результат выполнения представлен на рисунке 2.

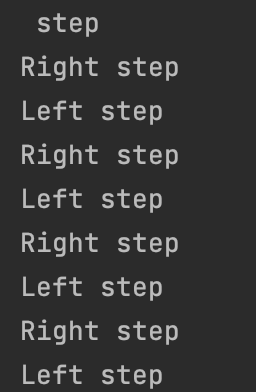


Рисунок 2 — пример выполнения

**Вывод:** были разработаны классы и методы согласно вариантам.