|  |  |
| --- | --- |
| 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**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 8 |

**Название:** Потоки Threads

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6И-21М |  |  | Ши Жуньфан |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П. В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

**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.RightFoot

|  |
| --- |
| package com.company;  public class Lab8\_1 {  public static final int *NACCOUNTS* = 10;  public static final double *INITIAL\_BALANCE* = 1000;  public static void main(String[] args) {  Bank b = new Bank(*NACCOUNTS*, *INITIAL\_BALANCE*);  int i;  for (i = 0; i < *NACCOUNTS*; i++) {  TransferRunnable r = new TransferRunnable(b, i, *INITIAL\_BALANCE*);  Thread t = new Thread(r);  t.start();  }  }  } class Bank{  private final double[] accounts;  public Bank(int n,double initialBalance) {  accounts = new double[n];  for (int i = 0; i < accounts.length; i++) {  accounts[i] = initialBalance;  }  }  public synchronized void transfer(int from ,int to ,double amount){  if(accounts[from] < amount){  System.*out*.println("Insufficient balance");  return ;  }  System.*out*.print(Thread.*currentThread*());  accounts[from] -= amount;  System.*out*.printf(" %10.2f from %d to %d%n",amount,from,to);  accounts[to] += amount;  System.*out*.printf("account %d balance: %10.2f,account %d balance:%10.2f%n",from,accounts[from],to,accounts[to]);  }  public int size(){  return accounts.length;  } }  class TransferRunnable implements Runnable{  private Bank bank;  private int fromAccount;  private double maxAmount;//  private int DELAY =10;   public TransferRunnable(Bank b, int from, double max) {  bank = b;  fromAccount = from;  maxAmount = max;  }  public void run() {  int toAccount = (int)(bank.size() \* Math.*random*());  double amount = maxAmount \* Math.*random*();  bank.transfer(fromAccount, toAccount, amount);  try {  Thread.*sleep*((int)(DELAY \* Math.*random*()));  } catch (InterruptedException e) {  e.printStackTrace();  }  }  } |