Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСТИТЕТ

Институт информационных технологий и анализа данных

ОТЧЕТ

к лабораторной работе по дисциплине:

|  |
| --- |
| **«Технология разработки программных комплексов»** |
| Распараллеливание потоков |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | ИСТб-19-1 |  |  |  | Путинцев О.Р. |
|  | шифр группы |  | подпись |  | фамилия И.О. |
| Проверил |  |  | 13 |  | Бахвалова З.А. |
|  | должность |  | подпись |  | фамилия И.О. |

г. Иркутск 2022 г.

**Оглавление**

[1. Задание к лабораторной работе: 3](#_Toc98446182)

[2. Диаграмма классов: 4](#_Toc98446183)

[3. Исходный код классов 5](#_Toc98446184)

[3.1. Main 5](#_Toc98446185)

[3.2. Generator 5](#_Toc98446186)

[3.3. NotGenerator 6](#_Toc98446187)

[4. Результат работы программы: 7](#_Toc98446188)

1. Задание к лабораторной работе:

Все варианты заданий подразумевают использование промежуточных буферов, представляющих собой динамические массивы. Максимальный размер буферов - N чисел. N определяется для каждого варианта. Потоки, помещающие числа в буферы, следят за переполнение буферов. Потоки, извлекающие числа из буферов, могут производить данную операцию в произвольный момент времени вне зависимости от того, заполнен ли буфер полностью или нет.

**Вариант 18.** Значения констант и реализуемые потоками функции:

**N=4**

**Первый поток** - генерирует в буфер 800 случайных чисел из интервала от 0 до 1.

**Второй поток** - переводит получившееся в буфере число из двоичной системы исчисления в восьмеричную, выводит полученный результат на экран и очищает буфер.

1. Диаграмма классов:

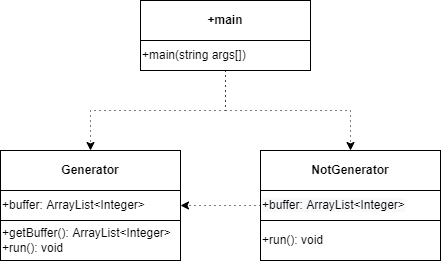


Рисунок 1 – Диаграмма классов

1. Исходный код классов
   1. Main

public class Main {  
 public Main(){  
  
 }  
 public static void main(String[] args) throws InterruptedException {  
 Generator generator= new Generator();  
 NotGenerator ng = new NotGenerator(generator);  
 generator.start();  
 ng.start();  
 }  
  
}

}

* 1. Generator

public class Generator extends Thread {  
 private volatile ArrayList<Integer> buffer = new ArrayList();  
  
 public Generator() {  
 }  
  
 public ArrayList<Integer> getBuffer() {  
 return this.buffer;  
 }  
  
 public void run() {  
 int count=0;  
 while(count < 800) {  
 synchronized(this.buffer) {  
 while(this.buffer.size() == 4) {  
 try {  
 this.buffer.wait();  
 } catch (InterruptedException var8) {  
 var8.printStackTrace();  
 }  
 }  
 int result =(int) (Math.random() \* 8);  
 this.buffer.add(result);  
 System.out.println("Первый поток. Число № " + count + " значение: " + result);  
 ++count;  
 this.buffer.notifyAll();  
 }  
 }  
  
 }

* 1. NotGenerator

public class NotGenerator extends Thread{  
 private volatile ArrayList<Integer> buffer;  
  
 public NotGenerator(Generator generator) {  
 this.buffer = generator.getBuffer();  
 }  
  
  
 public void run() {  
 int count = 0;  
  
 while(count < 800) {  
 synchronized(this.buffer) {  
 while(this.buffer.isEmpty()) {  
 try {  
 this.buffer.wait();  
 } catch (InterruptedException var10) {  
 var10.printStackTrace();  
 }  
 }  
 int num = (Integer)this.buffer.get(0);  
 this.buffer.remove(0);  
 String result1 = Integer.toBinaryString(num);  
 System.out.println("Второй поток. Число № " + count + " значение: " + result1);  
 ++count;  
 this.buffer.notifyAll();  
  
 }  
 }

1. Результат работы программы:

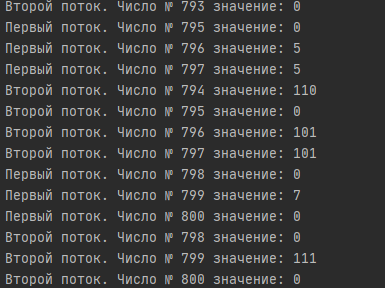


Рисунок 2 – Результат работы программы