|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №1**

по дисциплине «Архитектура клиент-серверных приложений»

**Тема практической работы: Многопоточные приложения, как основа распределённых клиент-серверных вычислений**

**Студент группы** ИКБО-10-19 Харитонов Дмитрий Михайлович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель практической работы** преподаватель Степанов В. С.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Москва 2021

**Цель:**

Ознакомится с моделью многопоточного программирования в Java.

**Задание:**

Используя материалы данной практической работы необходимо написать многопоточную программу, в которой два потока записывают строку в стандартный вывод, по образцу PING PONG PING PONG PING PONG. Программа должна работать следующим образом:

1. 1-й поток печатает «Ping» и переходит в состояние ожидания.
2. 2-й поток выходит из состояния ожидания, печатает «Pоng», уведомляет 1-й поток, возвращается в состояние ожидания.
3. 1-й поток выходит из состояния ожидания, печатает «Pшng», уведомляет 2-й поток, возвращается в состояние ожидания.
4. Шаги 2 и 3 повторяются и печатают «Ping Pong».

Программа должна быть реализована только с использованием Wait Notify, либо ReentrantLock.

**Ход работы:**

В ходе выполнения практической работы был создан класс, реализующий работу с потоками, согласно заданию. Ниже, на рисунках, представлена реализация программы и результат её работы.

Исходный код программы:

PingPong.java

class PingPong implements Runnable{  
 Object obj;  
 String str;  
 PingPong(Object obj,String str) {  
 this.str=str;  
 this.obj=obj;  
 }  
 public void run() {  
 synchronized (obj) {  
 while (true) {  
 System.*out*.println(str);  
 try {  
 Thread.*sleep*(500);  
 } catch (InterruptedException e1) {  
 e1.printStackTrace();  
 }  
 obj.notify();  
 try {  
 obj.wait();  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 }  
 }  
 }  
}

Program.java

public class Program {  
 public static void main(String[] args) {  
 Object obj= new Object();  
 Thread ping = new Thread(new PingPong(obj,"Ping!"));  
 Thread pong = new Thread(new PingPong(obj,"Pong!"));  
 ping.start();  
 pong.start();  
 }  
}

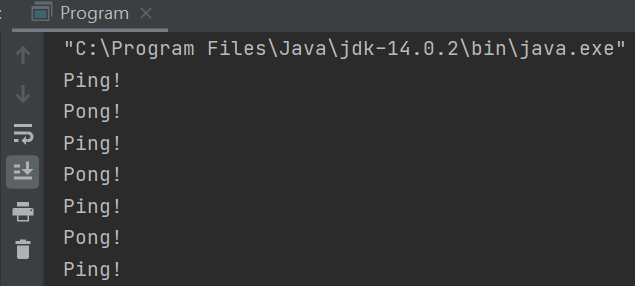


Рисунок 1 – результат выполнения программы

**Вывод:**

В результате выполнения практической работы были получены знания и навыки работы с многопоточной моделью программирования в JAVA.