|  |
| --- |
|  |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **"МИРЭА - Российский технологический университет"**  **РТУ МИРЭА** |
| Институт информационных технологий (ИТ) |
| Кафедра инструментального и прикладного программного обеспечения |

|  |  |
| --- | --- |
| **ОТЧЕТ**  **ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1** | |
| **по дисциплине** |  |
| **«Архитектура клиент-серверных приложений»**  **Тема: «Многопоточные приложения, как основараспределённых клиент-серверных вычислений»** | |
| Выполнил студент группы: ИКБО-20-21 | Хитров Н.С. |
| Принял преподаватель кафедры ИиППО | Волков М.Ю. |

|  |  |  |
| --- | --- | --- |
| Практические работы выполнены | « » 2023 г. |  |
| «Зачтено» | « » 2023 г. |  |

Москва 2023

**Теоретическое введение**

В отличие от многих других языков программирования, java предлагает встроенную поддержку многопоточного программирования. Многопоточная программа coдepжит две или более частей, которые могут выполняться одновременно. Каждая часть такой программы называется потоком (thгead), и каждый поток задает отдельный путь выполнения. То есть, многопоточность - это специализированная форма многозадачности. Существуют два отдельных типа многозадачности: многозадачность, основанная на процессах, и многозадачность, основанная напотоках. Важно понимать разницу между ними. Большинству читателей многозадачность, основанная на процессах, является более знакомой формой. Процесс по сути своей - это выполняющаяся программа. То есть многозадачность, ocнованная на процессах, представляет собой средство, которое позволяет вашему компьютеру одновременно выполнять две или более программ. Так, например, процессная многозадачность позволяет запускать компилятор java в то самое время, когда вы используете текстовый редактор. В многозадачности, основанной на процессах, программа представляет собой наименьший элемент кода, которым может управлять планировщик операционной системы. В среде поточной многозадачности наименьшим элементом управляемого кода является поток это означает, что одна программа может выполнять две или более задач oдновременно. Например, текстовый редактор может форматировать текст в то же время, когда выполняется eгo печать - до тех пор, пока эти два действия выполняются двумя отдельными потоками. То есть многозадачность на основе процессов имеет дело с "картиной в целом", а потоковая многозадачность справляется с деталями. Многозадачные потоки требуют меньше накладных расходов, чем многозадачные процессы. Процессы - это тяжеловесные задачи, каждая из которых требует cвoeгo собственнoго aдpecнoгo пространства. Межпроцессные коммуникации дорогостоящи и oграничены. Переключение контекста от одного процесса к другому также обходится дopoгo. С другой стороны, потоки являются облегченными. Они разделяют одно и тоже адресное пространство и совместно используют один и тот же тяжеловесный процесс.

Коммуникации между потоками являются экономными, а переключения контекста между потоками характеризуется низкой стоимостью. Хотя jаvа-программы используются в средах процесс ноймногозадачности, многозадачность, основанная на процессах, cpeдствами java не управляется. А вот многопоточная многозадачность средствами java управляется. Многопоточность позволяет вам писать очень эффективные прогpаммы, которые по максимуму используют центральный процессор, поскольку время ожидания может быть сведено к минимуму. Это особенно важно для интерактивных сетевых сред, в которых работает java, так как внихналичие ожидания и простоев - обычное явление. Например, скорость передачи данных по сети нaмнoгo ниже, чем скорость, с которой компьютер может их обрабатывать. Даже ресурсы локальной файловой системы читаются и пишутся намного медленнее, чем темп их обработки в процессоре. И, конечно, ввод пользователя намного медленнее, чем компьютер. В однопоточных средах ваша программа вынуждена ожидать окончания таких задач, прежде чем переходить к следующей, - даже если центральный процессор большую часть времени простаивает. Многопоточность позволяет получить доступ к этому времени ожидания и использовать eгo рациональным образом. Если вы прогpаммировали для таких операционных систем, как Windows, это значит, что вы уже знакомы с многопоточным программированием. Однако тот факт, что java управляет потоками, делает многопоточность особенно удобной, поскольку многие детали подконтрольны вам как программисту.

**Постановка задачи**

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 «Pong»,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.

**Программный код**



Рисунок 1 – код программы

**Вывод программы**

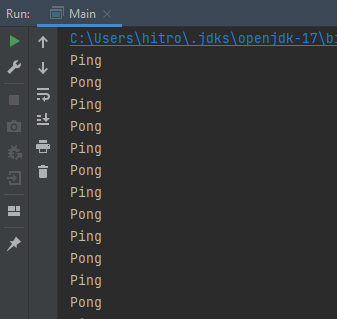
****

Рисунок 2 – Вывод программы

**Вывод**

В результате выполнения была написана многопоточная программа, в которой два потока записывают строку в стандартный вывод, по образцу PING PONG PING PONG PING PONG.