**Лабораторная работа 1**

**08.02.2024**

**Разработка многопоточных приложений на языке Java**

**Источники**

1. Хорстманн, Кей С. Java. Библиотека профессионала, том 1. Основы. 11-е изд.: Пер. с англ. СПб.: ООО "Диалектика", 2019.
2. Блинов, И. Н., Романчик, В. С. Java from EPAM : учеб.-метод. пособие / И. Н. Блинов, В. С. Романчик. —2-е издание. Минск: Четыре четверти, 2021. —560 с.
3. <https://math.hws.edu/javanotes/>
4. Paul Deitel, Harvey Deitel - Java How to Program, Early Objects-Pearson Education (2015)

**Правила оформления отчета**

* Берем файл с условием.
* Переименовываем – добавляем свою фамилию к имени файла.
* Вставляем в «***Отчет***» результаты выполнения.

***Задание 1.***

Изучить (вспомнить) способы создания и запуска потока в Java.

[2, 369-370]

[1,12.1, 662-667]

***Задание 2.***

***2.1.***

Скомпилировать и запустить программу:

* class ThreadTest1

[3]

<https://math.hws.edu/javanotes/c12/s1.html>

Чтобы помочь вам понять, как несколько потоков выполняются параллельно, мы рассмотрим пример программы ThreadTest1.java. Эта программа создает несколько потоков. Каждый поток выполняет одну и ту же задачу. Задача — посчитать количество простых чисел меньше 10 000 000. (Конкретная выполняемая задача не важна для наших целей, поскольку она занимает нетривиальное количество времени. Это демонстрационная программа; в реальной программе было бы глупо иметь несколько потоков, которые сделайте то же самое, и метод, используемый для подсчета простых чисел, очень неэффективен.)

Главный поток не ожидает завершения дочерних потоков.

***Знать:***

* как замерять время выполнения вычислений

long startTime = System.currentTimeMillis();

int count = countPrimes(2,MAX);

long elapsedTime = System.currentTimeMillis() - startTime;

System.out.println("Thread " + id + " counted " +

count + " primes in " + (elapsedTime/1000.0) + " seconds.");

***2.2.***

Жизненный цикл потока [2, 371-372].

***2.3.***

[3]

<https://math.hws.edu/javanotes/c12/s1.html>

Добавить, чтобы главный поток ожидал завершения дочерних потоков.

***Знать:***

* Определение состояния потока.
* Ожидание завершения потока.

***2.4.***

1. Последовательная программа.

Реализовать решение задачи «Посчитать количество простых чисел меньше 10 000 000» в одном потоке.

2. Многопоточная (параллельная) программа

class ThreadTest1 – разделить работу между потоками.

3. Измерить время работы Последовательной и Многопоточной программ.

4. Провести эксперименты с многопоточной реализацией с разным количеством потоков.

5. Результата экспериментов сохранить.

***Отчет:***

* Текст программы.
* Результаты экспериментов.

***2.5.***

Многопоточная программа 2.

Creating and Executing Threads with the Executor Framework [4, 23.3].

***Отчет:***

* Текст программы.
* Результаты экспериментов.

***Задание 3.***

Задача: Calculate() выполняет трудоемкие вычисления

Цель: Улучшить отзывчивость программы

Средство: Разделить по потокам пользовательский интерфейс (консольное приложение) и трудоемкие вычисления

После запуска вычислений в отдельном потоке предоставить пользователю выбор: выполнить все вычисления или преждевременно завершить вычисления. Протестировать оба варианта.

***Отчет:***

* Текст программы.