6 Лабораторная работа №6 Исследование способов профилирования программного обеспечения

6.1. Цель работы

Исследовать критические по времени выполнения участки программного кода и возможности их устранения. Приобрести практические навыки анализа программ с помощью профайлера VisualVM.

6.2. Постановка задачи

Разработать программу на основе библиотеки классов, реализованной и протестированной в предыдущей работе. Программа должна как можно более полно использовать функциональность класса. При необходимости для наглядности профилирования в методы класса следует искусственно внести задержку выполнения. Выполнить профилирование разработанной программы, выявить функции, на выполнение которых тратится наибольшее время. Модифицировать программу с целью оптимизации времени выполнения. Выполнить повторное профилирование программы, сравнить новые результаты и полученные ранее, сделать выводы.

6.3. Ход выполнения работы

Для выполнения данной лабораторной работы была разработана программа, состоящая из основного класса, представленного в листинге 1, и библиотечных классов, составленных в предыдущих лабораторных работах. Для наглядности

профилирования в метод Matrix#columnPositionWithLongestSeries была добавлена задержка выполнения с помощью Thread#sleep с аргументом 100 миллисекунд.

Листинг 1 — Класс Арр

```
package org.cory7666.softwaretestingexample;
import java.io.File;
import org.cory7666.softwaretestingexample.task1.FileWithMatrix;
import org.cory7666.softwaretestingexample.task1.Matrix;
  public static void main (String[] args) throws Exception
    if (args.length != 2)
      System.err.println("Usage: program [file-with-matrix]"); System.exit(1);
    else
      final File file = new File(args[0]);
      System.out.printf("Получен файл \"%s\".%n", file.toPath());
      final Matrix matrix = new FileWithMatrix(file).parse();
      System.out.println("Данные из файла считаны.");
      int result = matrix.columnPositionWithLongestSeries();
      System.out.printf("Ответ: %d.%n", result);
      System.out.println("Выполнения програмы завершено.");
      System.exit(0);
 }
}
```

Программа была скомпилирована. Для этой лабораторной работы была составлена матрица, имеющая 20 столбцов и 1 000 000 строчек.

Снимок выполнения программы с задержками представлен на рисунке, без задержек — на рисунке, результат сравнения двух снимков — на рисунке. Каждый из снимков содержит время выполнения функции по реальному времени/по времени процессора, количество вызовов.

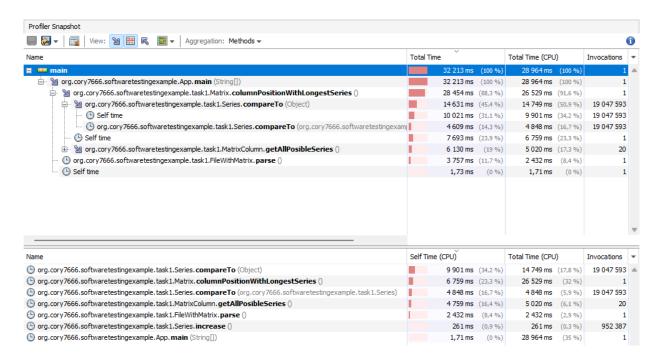


Рисунок 6.1 — Выполнение программы с задержками

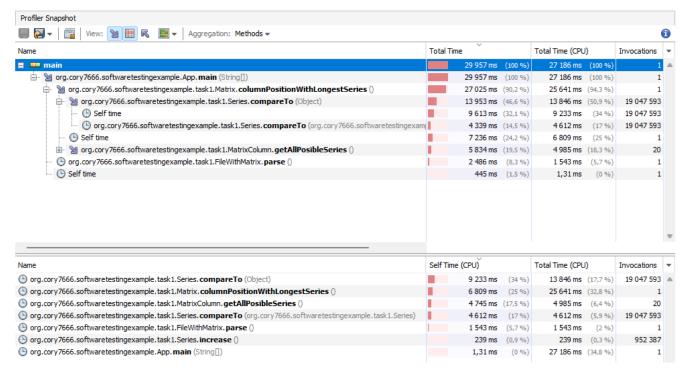


Рисунок 6.2 — Выполнение программы без задержек

		■ 🔄 🔻 📦 View: 💟 🔡 🖟 🗮 🕶 Aggregation: Methods 🕶			
· main	Total	Time	Total Time (CPU)	Invocations	
		-2 256 ms	-1 778 ms	+0	
⊕ Morg.cory7666.softwaretestingexample.App. main (String[])		-2 256 ms	-1 778 ms	+0	
··· 🕒 Self time		+444 ms	-0,398 ms	+0	
		-1 271 ms	-888 ms	+0	
🖮 🤰 org.cory7666.softwaretestingexample.task1.Matrix.columnPositionWithLongestSeries ()		-1 429 ms	-888 ms	+0	
	- 1	-295 ms	-35,3 ms	+0	
(org.cory7666.softwaretestingexample.task1.Series.increase ()		-38,0 ms	-22,1 ms	+0	
	1	-257 ms	-13,2 ms	+0	
···· (5) Self time		-456 ms	+49,6 ms	+0	
⊕ 🕍 org.cory7666.softwaretestingexample.task1.Series.compareTo (Object)	1	-677 ms	-903 ms	+0	
🕒 org.cory7666.softwaretestingexample.task1.Series.compareTo (org.cory7666.softwaretestingexam		-269 ms	-235 ms	+0	
(<u>Self time</u>	1	-407 ms	-667 ms	+0	
me	Self1	Time (CPU)	Total Time (CPU)	Invocations	
org.cory7666.softwaretestingexample.task1.Matrix.columnPositionWithLongestSeries ()	JCII I	+49,6 ms	. ,		
org.cory7666.softwaretestingexample.App. main (String[])		-0,398 ms			
org.cory7666.softwaretestingexample.task1.MatrixColumn.getAllPosibleSeries ()		-13.2 ms			
org.cory7666.softwaretestingexample.task1.Series.increase ()		-22,1 ms	,		
org.cory7666.softwaretestingexample.task1.Series.compareTo (org.cory7666.softwaretestingexample.task1.Series)		-22,1 ms			
org.cory7666.softwaretestingexample.task1.Series.compareTo (org.cory7666.softwaretestingexample.task1.Series)		-255 ms			
org.cory7666.softwaretestingexample.task1.FileWithMatrix.parse ()		-888 ms			

Рисунок 6.3 — Сравнение двух снимков

Профайлер показывает, что горячей точкой программы является функция Series#compareTo. Если убрать задержку из функции Matrix# columnPositionWithLongestSeries, время выполнения функции уменьшится на 1429мс.

Выводы по результатам работы

При выполнении данной лабораторной работы были получены навыки работы с профайлером VisualVM. Определено, что профайлер следует использовать для определения «узких мест» программы, времени выполнения каждой из функций.