Федеральное государственное автономное образовательное учреждение высшего образования

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Факультет: ПИиКТ

Направление 09.03.04 «Системное и прикладное программное обеспечение»

Мегафакультет: КТиУ

**ЛАБОРАТОРНАЯ РАБОТА №4**

по дисциплине:

«МЕТОДЫ И СРЕДСТВА ПРОГРАММНОЙ ИНЖЕНЕРИИ»

**Выполнили:**

Студенты 2 курса

группы P3215

Вариант 2240

Девяткин А. Ю.

Ибадулаев А. Э. о.

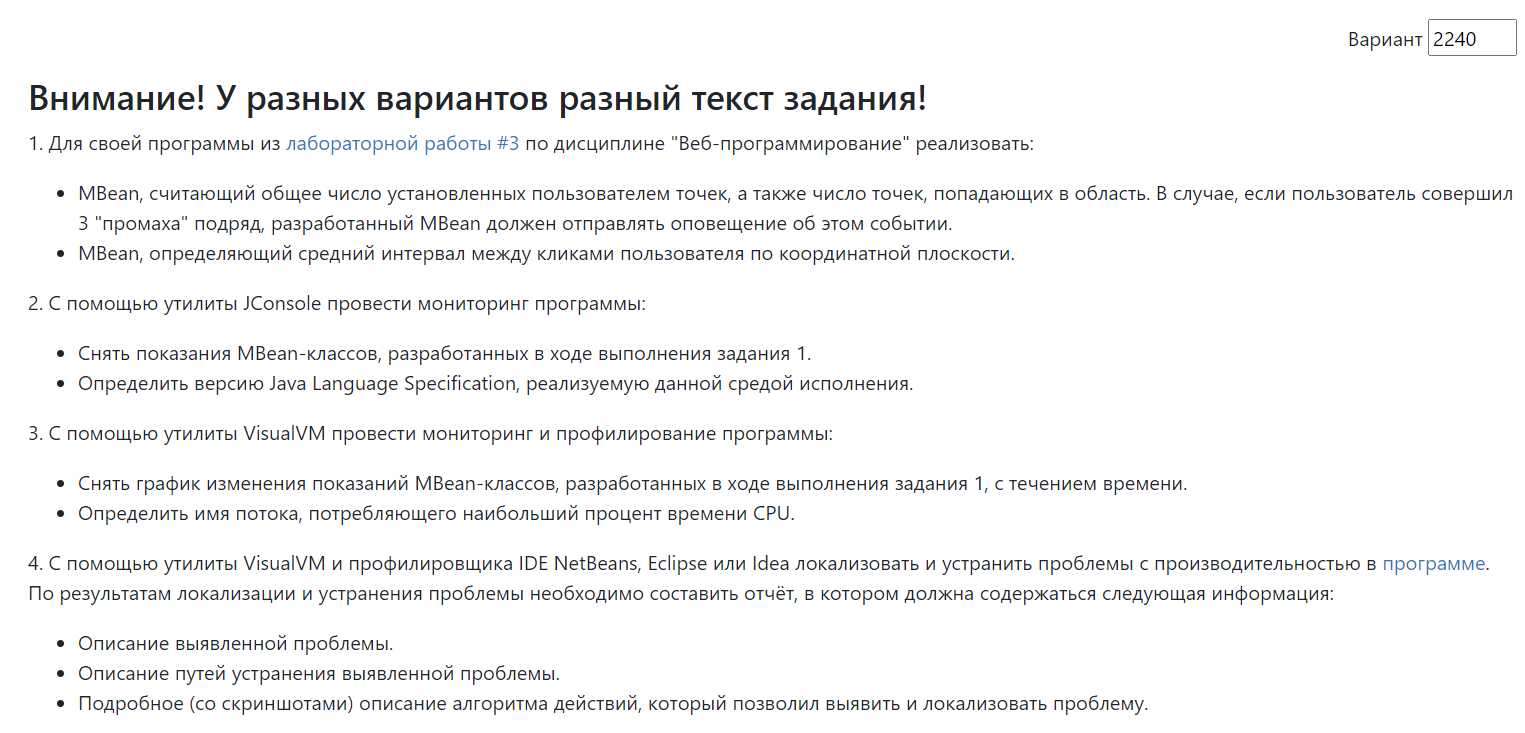
**Преподаватель:**

Покид А.В.

Санкт-Петербург

2022

# Задание



# Ссылка на исходный код

# <https://github.com/AliIbadullayev/Basics-of-Software-Engineering/tree/main/lab4>

# Скриншоты программы VisualVM с комментариями по ходу поиска утечки памяти

1. Чтобы определить проблему, мы начали тестировать, где и как быстро росла куча. Мы видим, что byte[] в VisualVM | | Sampler | Memory растет очень быстро. Изображение выглядит как текст

   Автоматически созданное описание
2. Это может быть из-за утечек памяти. (Когда указатель на неиспользуемый объект остается с этим объектом) и поэтому сборщик мусора не может работать так, как нужно. Чтобы решить утечку памяти, мы увидели возможные статические поля (возможно, это может вызвать эту проблему). И через несколько раз мы увидели, что наша программа зависала на месте, когда ей нужно было выполнить в классе JavaScript.java метод

private void handleScriptException( Exception e, String badScript )

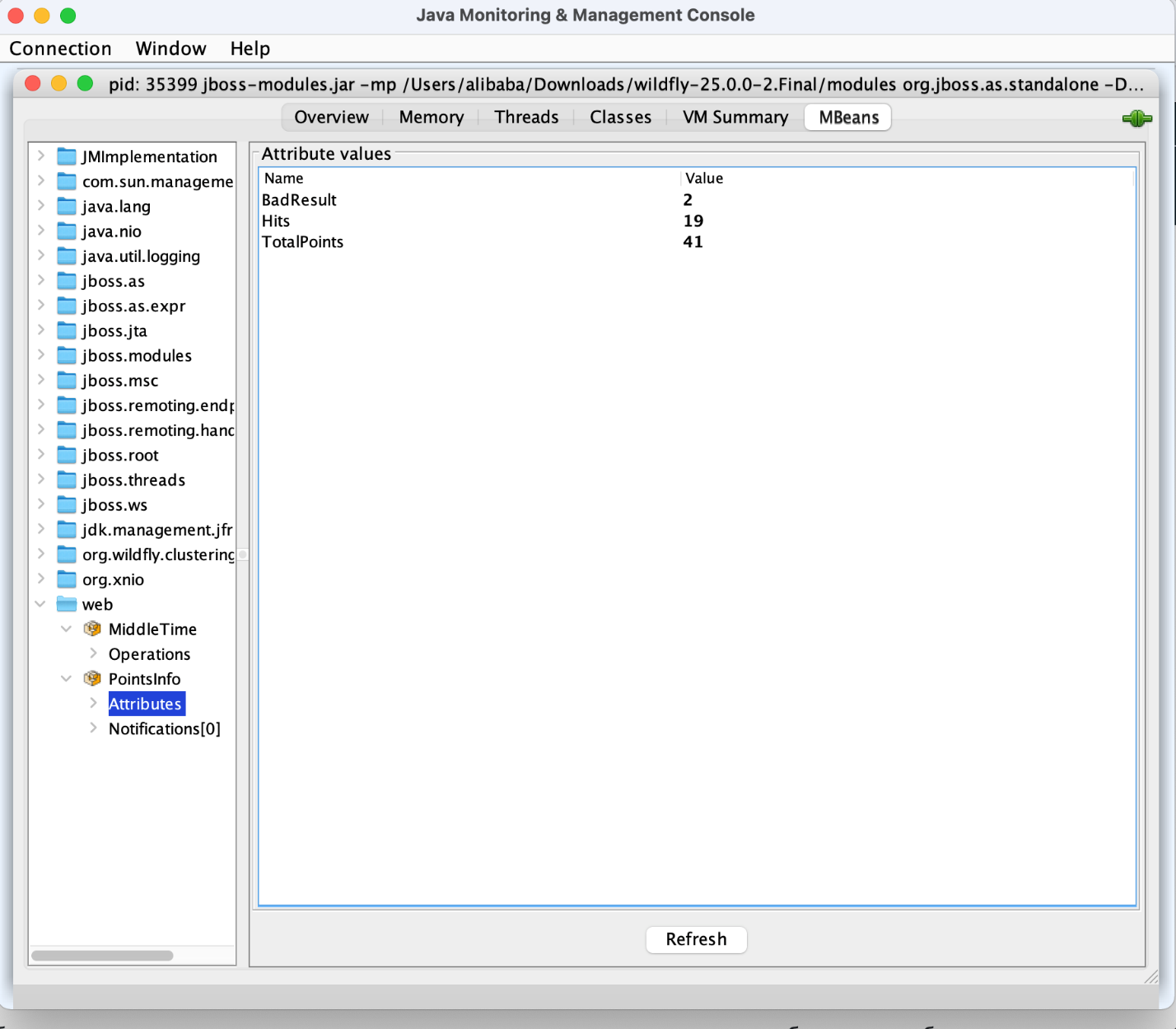
Изображение выглядит как текст, щуп, снимок экрана

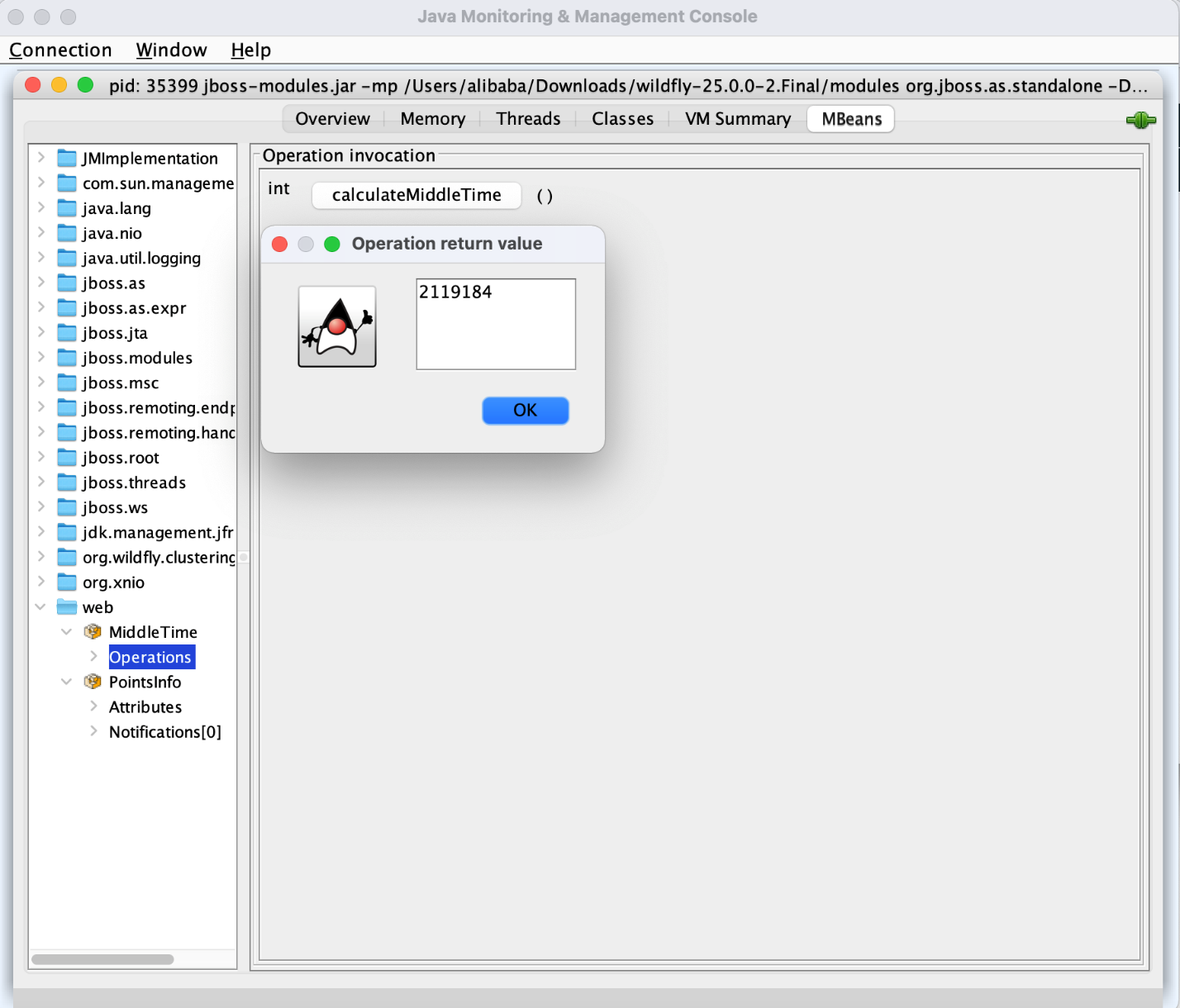
Автоматически созданное описание

1. Решение проблемы мы придумали следующее: вместо того, чтобы хранить все записи об ошибках, которые в ходе программы всё равно нигде не считываются, мы решили хранить только последние 100 (или любое другое количество, которое можно задать константой *MAX\_LENGTH\_OF\_ERRORS\_LIST*

if (*\_errorMessages*.size() > *MAX\_LENGTH\_OF\_ERRORS\_LIST*)  
 *\_errorMessages*.remove(0);  
*\_errorMessages*.add( errorMessage );

# Скриншоты программы JConcole со снятыми показаниями



 Изображение выглядит как текст

Автоматически созданное описание

# 

# 

# Вывод

В ходе выполнения лабораторной работы мы познакомились с утилитами профилирования и мониторинга VisualVM и JConsole, ознакомились с технологией MBean и реализовали 2 бина, а также с помощью выявили и устранили проблемы с производительностью выданной нам программы.