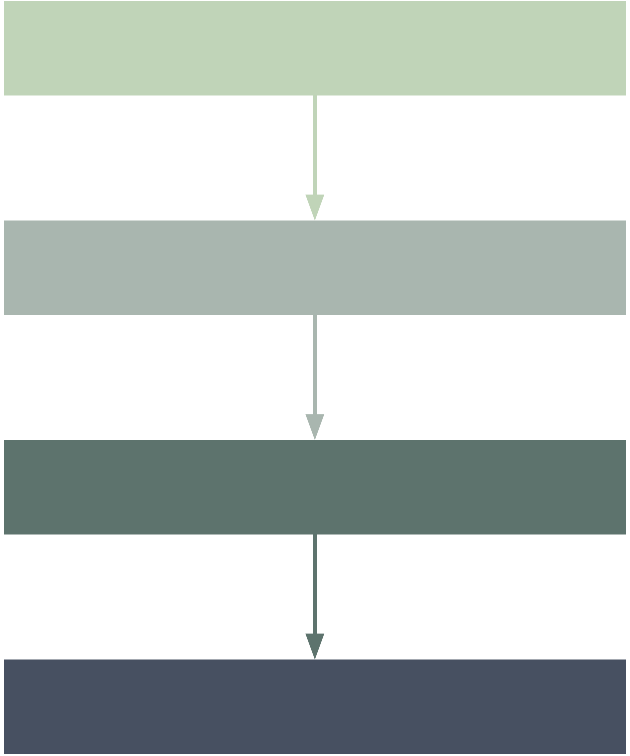


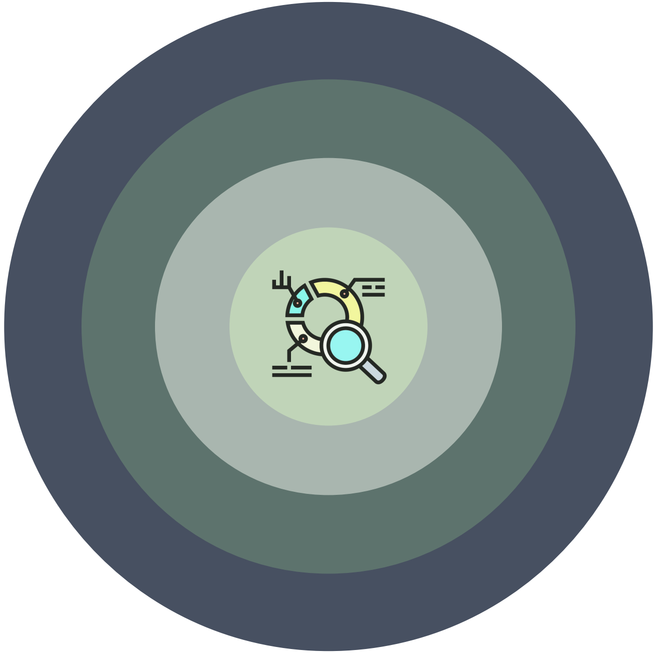
Run-time Efficiency Assessment for the Selection of Optimized Data Structures for Java Programs

- Janitha Nawarathna | S/16/417

Introduction

01 **Purpose** and objectives

02 **Literature** and methods



03 **Benchmarking** the selected data structures

04 **Results** and analysis

Purpose(aim)

Analysing the run-time behaviour of Java applications that use data structures in the implementation, and suggest if there are better, available data structures to use instead.

Objectives

- Develop a program solution to store information of run-time behaviour of Java data structures from Java List interface in the Java Collection interface.
- Identify the best possible alternative data structures for a Java program, within the chosen data structures using machine learning.
- Propose a system to Integrate the information of run-time behaviour and generate an assisting report.

Literature and Methods

Method 1

Method 2

Literature and Methods

Method 1

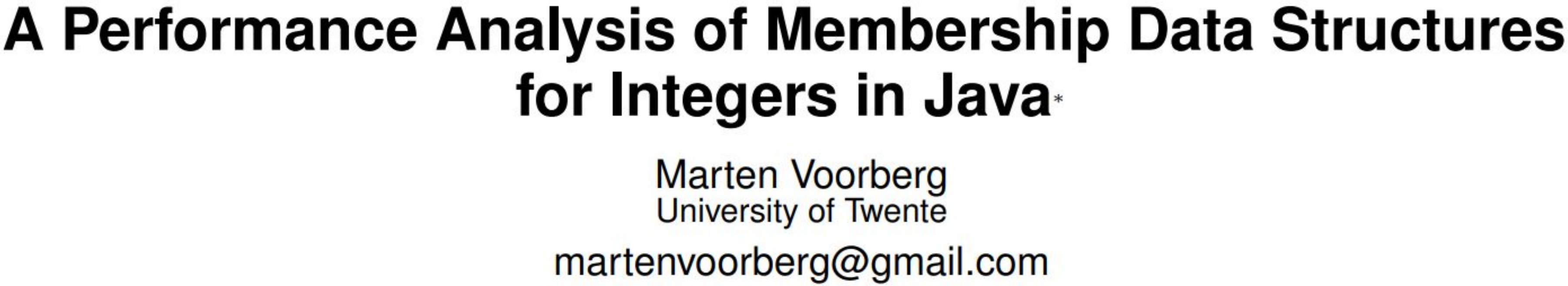
Benchmarking Java Applications from GitHub

Literature and Methods

Method 2

Implementing a Java Application to Benchmark with a Larger Dataset

Literature and Methods



A Performance Analysis of Membership Data Structures for Integers in Java*

Marten Voorberg
University of Twente

martenvoorberg@gmail.com

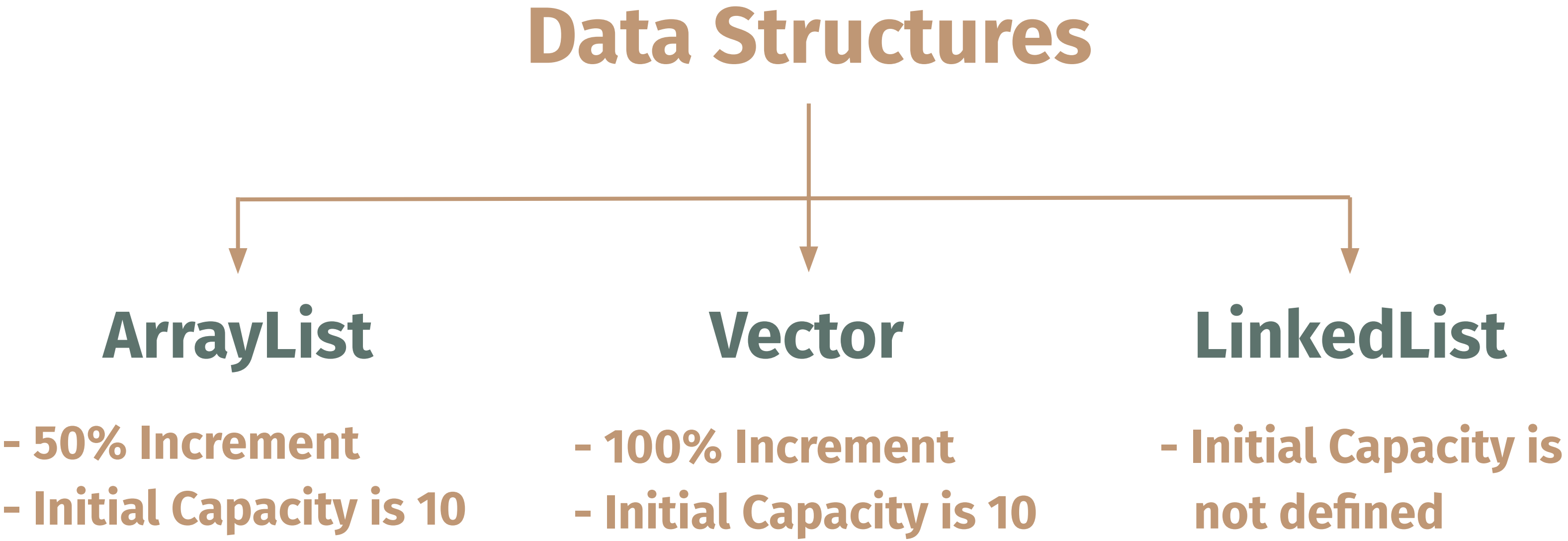
Benchmarking

Performance Measurements

Runtime

Memory Usage

Benchmarking



Data Structures

ArrayList

- 50% Increment
- Initial Capacity is 10

Vector

- 100% Increment
- Initial Capacity is 10

LinkedList

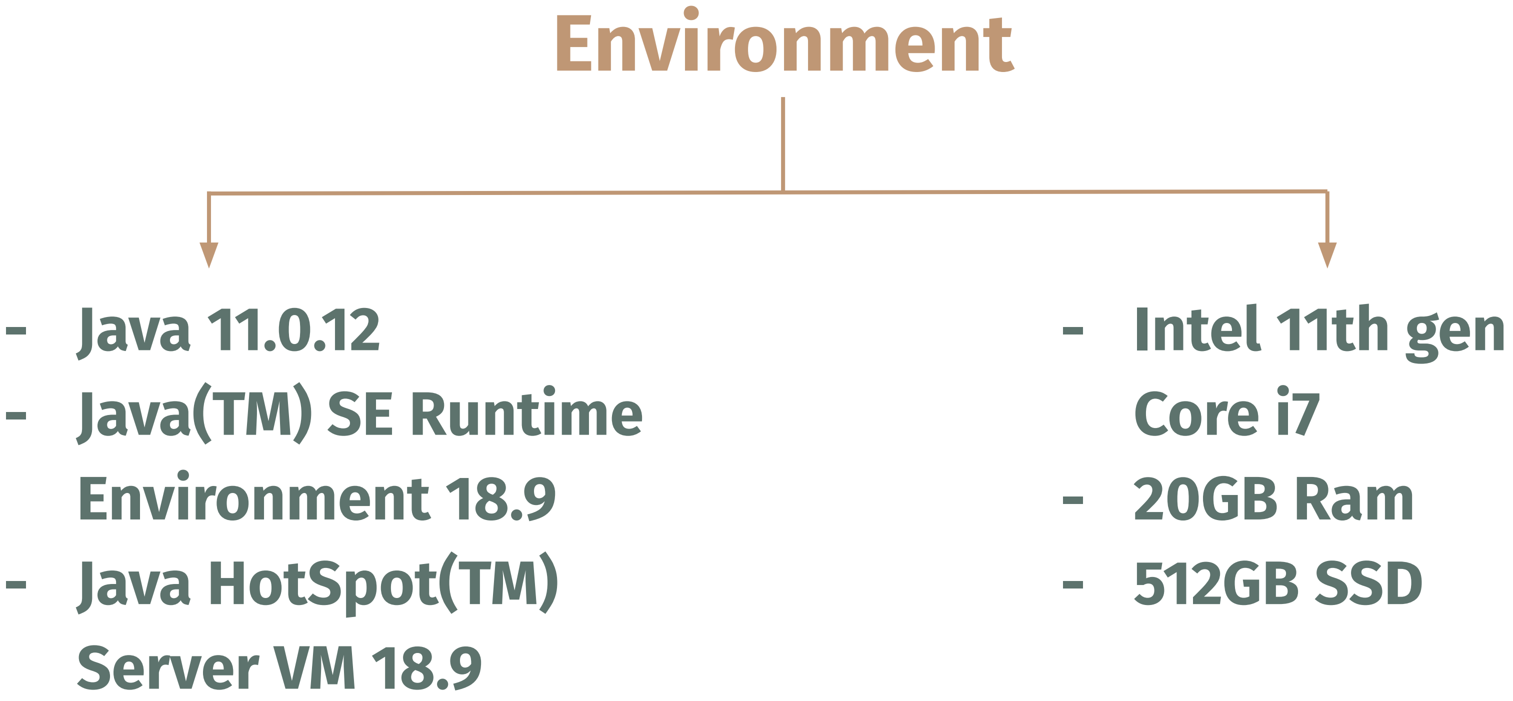
- Initial Capacity is not defined

Benchmarking

Functions

- `add()`
- `addAt()`
- `contains()`
- `indexOf()`
- `get()`
- `delete()`

Benchmarking



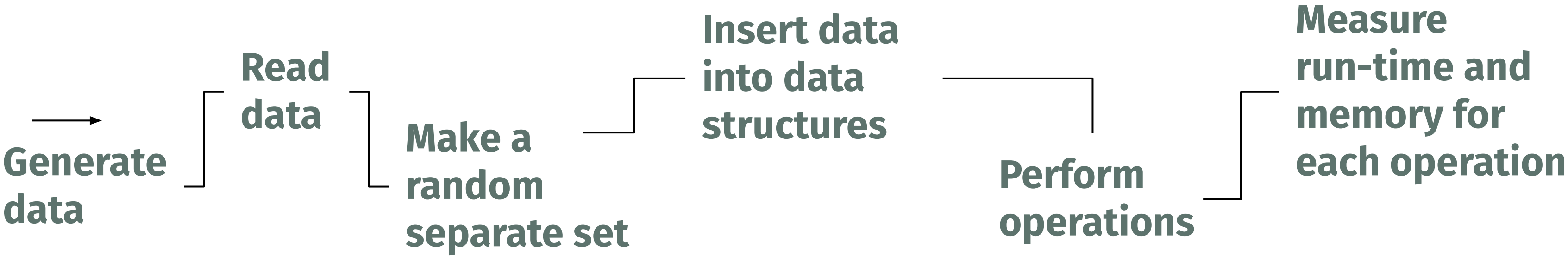
Environment

- Java 11.0.12
- Java(TM) SE Runtime Environment 18.9
- Java HotSpot(TM) Server VM 18.9

- Intel 11th gen Core i7
- 20GB Ram
- 512GB SSD

Benchmarking

Process



Generate data

Read data

Make a random separate set

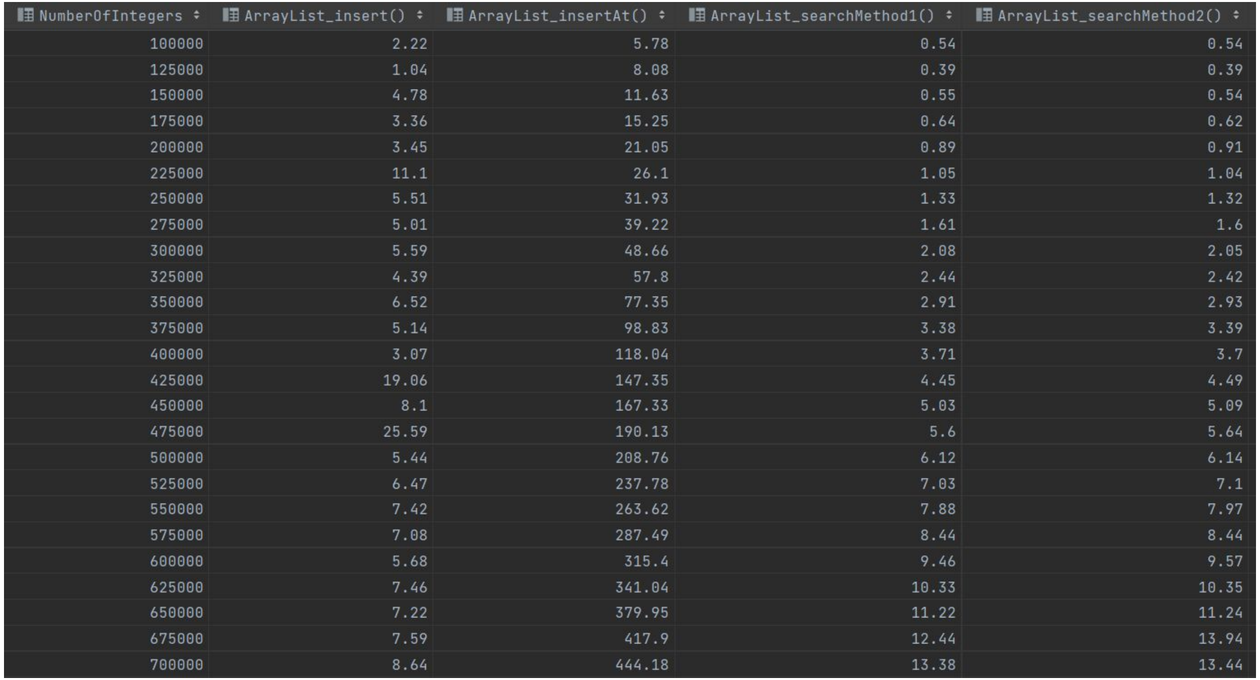
Insert data into data structures

Perform operations

Measure run-time and memory for each operation

Analysis

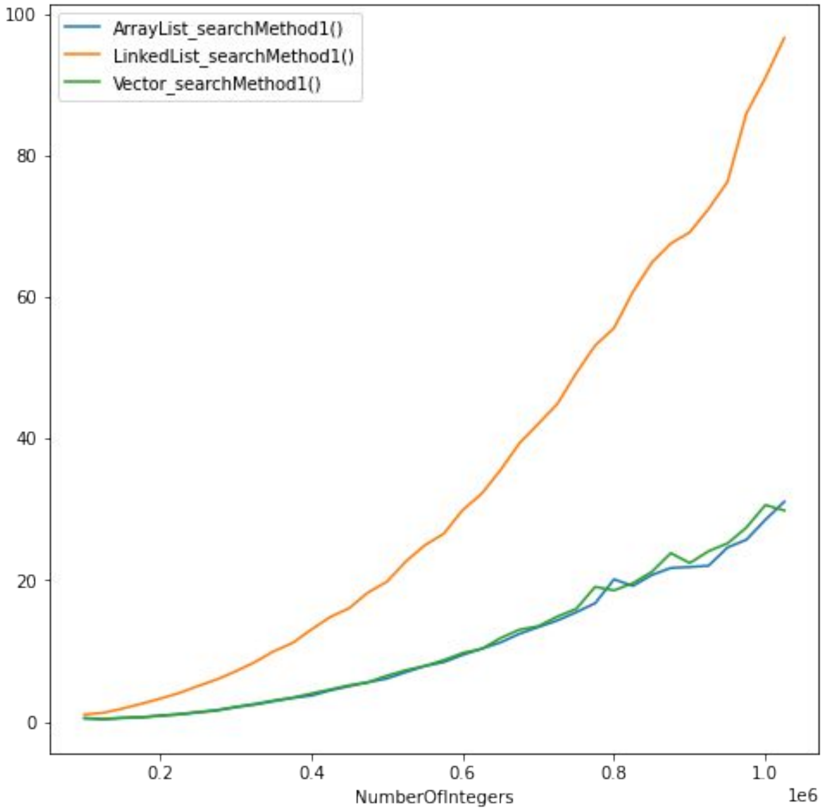
Collected Data - Runtime(ms)



NumberOfIntegers	ArrayList_insert()	ArrayList_insertAt()	ArrayList_searchMethod1()	ArrayList_searchMethod2()
100000	2.22	5.78	0.54	0.54
125000	1.04	8.08	0.39	0.39
150000	4.78	11.63	0.55	0.54
175000	3.36	15.25	0.64	0.62
200000	3.45	21.05	0.89	0.91
225000	11.1	26.1	1.05	1.04
250000	5.51	31.93	1.33	1.32
275000	5.01	39.22	1.61	1.6
300000	5.59	48.66	2.08	2.05
325000	4.39	57.8	2.44	2.42
350000	6.52	77.35	2.91	2.93
375000	5.14	98.83	3.38	3.39
400000	3.07	118.04	3.71	3.7
425000	19.06	147.35	4.45	4.49
450000	8.1	167.33	5.03	5.09
475000	25.59	190.13	5.6	5.64
500000	5.44	208.76	6.12	6.14
525000	6.47	237.78	7.03	7.1
550000	7.42	263.62	7.88	7.97
575000	7.08	287.49	8.44	8.44
600000	5.68	315.4	9.46	9.57
625000	7.46	341.04	10.33	10.35
650000	7.22	379.95	11.22	11.24
675000	7.59	417.9	12.44	13.94
700000	8.64	444.18	13.38	13.44

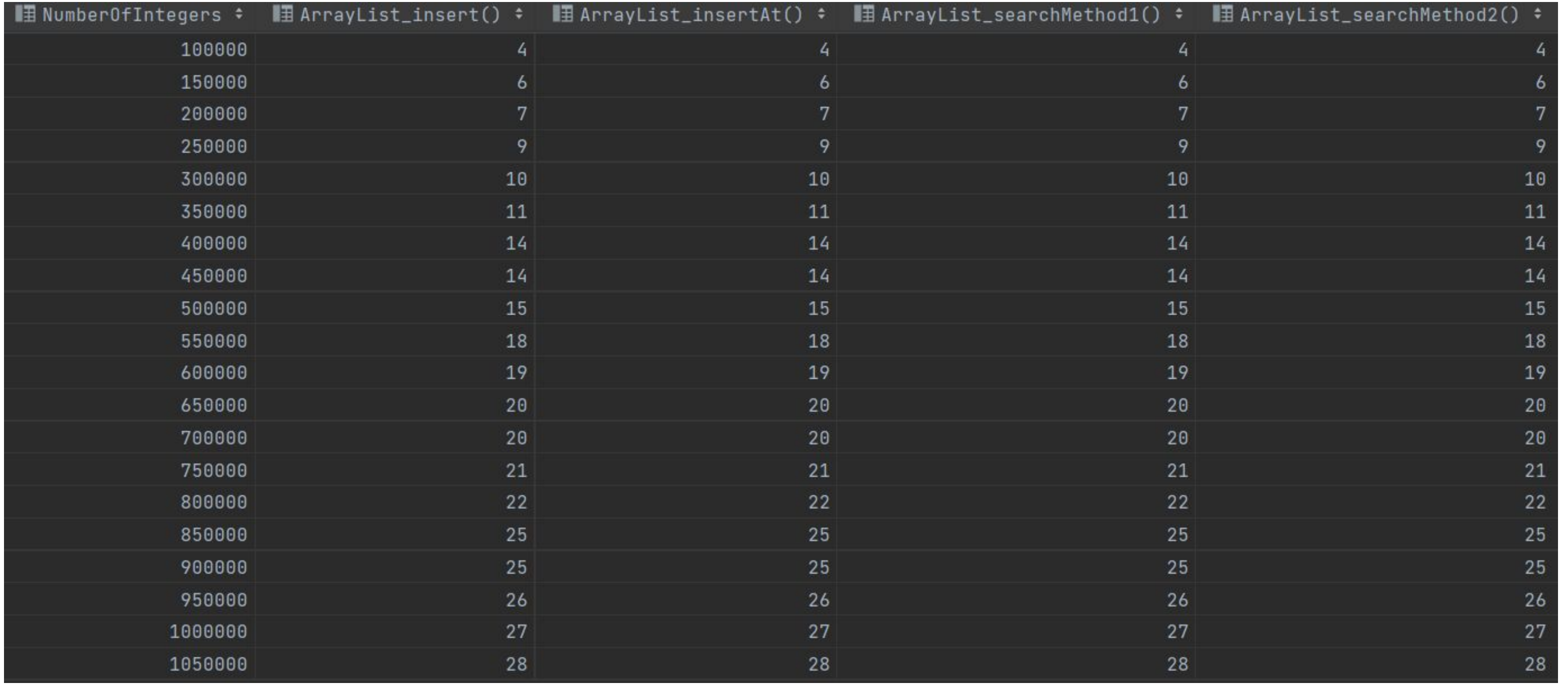
Analysis

Runtime Behaviour



Analysis

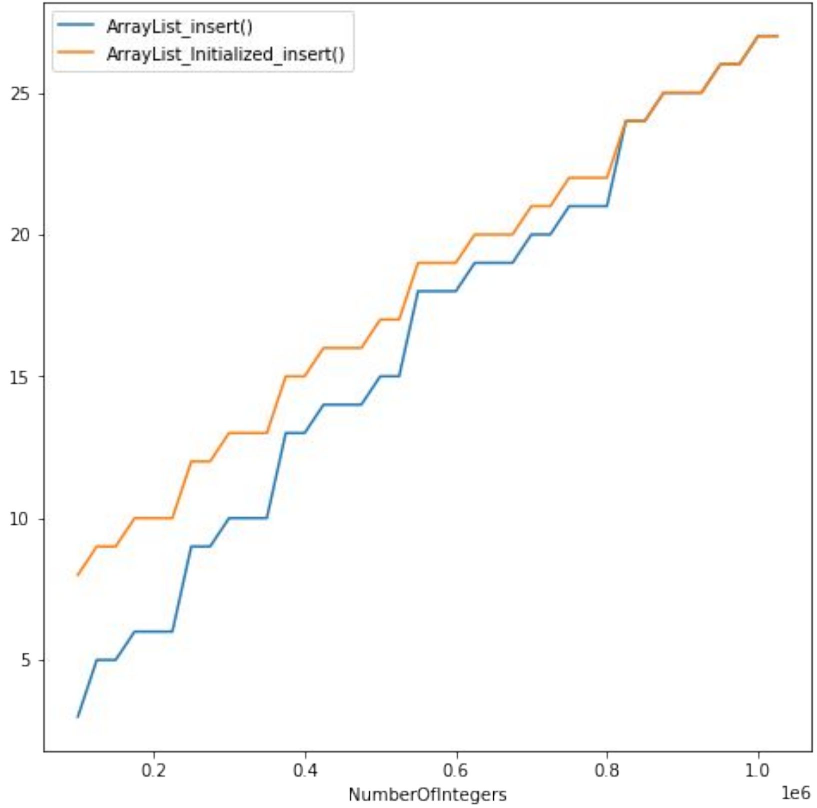
Collected Data - Memory(Mb)



NumberOfIntegers	ArrayList_insert()	ArrayList_insertAt()	ArrayList_searchMethod1()	ArrayList_searchMethod2()
100000	4	4	4	4
150000	6	6	6	6
200000	7	7	7	7
250000	9	9	9	9
300000	10	10	10	10
350000	11	11	11	11
400000	14	14	14	14
450000	14	14	14	14
500000	15	15	15	15
550000	18	18	18	18
600000	19	19	19	19
650000	20	20	20	20
700000	20	20	20	20
750000	21	21	21	21
800000	22	22	22	22
850000	25	25	25	25
900000	25	25	25	25
950000	26	26	26	26
1000000	27	27	27	27
1050000	28	28	28	28

Analysis

Memory Behaviour



Analysis

Building a Machine Learning Model

References

1. Marten Vooberg: A Performance Analysis of Membership Data Structures for Integers in Java(2021)
2. Woodside, M., Franks, G., Petriu, D.C.: The future of software performance engineering. In: Future of Software Engineering (FOSE) (2007)
3. Wu, X., Woodside, M.: Performance modeling from software components. SIGSOFT Softw. Eng. Notes 29(1), pp. 290–301 (2004)
4. M. Harkema, D. Quaetel, B.M.M. Gijsen, R.D. van der Mel: Performance Monitoring of Java Applications (2002)
5. Java Documentation. [Online]. Available from: https://docs.oracle.com/javase/tutorial/collections/interfaces/list.html

Thank You!

Any Questions?