**Linnéuniversitetet**
Kalmar Växjö

Report

# Time

*Author:* Xiaohe Zhu
*Semester*: Autumn 2019
*Course code*: 1DV507

**Linnéuniversitetet**
Kalmar Växjö

## Abstract

This is a short report of Exercises 6 and 7 in Assignment 4 in the course 1DV507 – Programming and Data Structures. The task of exercise 6 is to find how many strings can be concatenation using "+" and append using string builder in one second, and what is difference between short string and long string. The task of exercise 7 is to find how many integers and strings can be sorted in 1 second when using: a) insertion sort b) merge sort.

# 1 Exercises

The task of exercise 6 is to find how many strings can be concatenation using "+" and append using string builder in one second, and what is difference in time between short string and long string. The task of exercise 7 is to find how many integers and strings can be sorted in 1 second when using: a) insertion sort b) merge sort. The method of insertion sort and merge sort are defined in assignment 3.

# 2 Experimental Setup

All experiments was done on a MacBook Pro with an Intel Core i5 processor (2.3GHz) with 8GB of memory. We used JavaSE-1.8 and during the experi- ments we also allowed the Java Virtual Machine to use a heap space of 4GB us- ing the VM arguments: -Xmx4096m -Xms4096m. Furthermore, we closed all other applications while performing the experiments and made sure that neither input data generation nor print-outs was done during the time measurements. We used the clock System.currentTimeMillis() in all our measurements.

We used the same approach in all experiments. Each time measurement is an average of 10 consecutive runs (i.e., sorting an array or a list) and we always make sure to generate our input data (random arrays/lists) before the time measurements start.

# 3 String Concatenation

Short concatenation

| | times | length |
|---|---|---|
| Execute 1 | 40253 | 40253 |
| Execute 2 | 43574 | 43574 |
| Execute 3 | 44078 | 44078 |
| Execute 4 | 41085 | 41085 |
| Execute 5 | 44115 | 44115 |

Long concatenation

| | times | length |
|---|---|---|
| Execute 1 | 5916 | 473280 |
| Execute 2 | 5944 | 475520 |
| Execute 3 | 5966 | 477280 |
| Execute 4 | 5383 | 430640 |
| Execute 5 | 5730 | 458400 |

Short append

| | times | length |
|---|---|---|
| Execute 1 | 31068707 | 31068707 |
| Execute 2 | 31757842 | 31757842 |

| | | |
|---|---|---|
| Execute 3 | 30206849 | 30206849 |
| Execute 4 | 29080799 | 29080799 |
| Execute 5 | 29491905 | 29491905 |

Long concatenation

| | times | length |
|---|---|---|
| Execute 1 | 3441603 | 275328240 |
| Execute 2 | 4061225 | 324898000 |
| Execute 3 | 3798701 | 303896080 |
| Execute 4 | 3944985 | 315598800 |
| Execute 5 | 3415591 | 273247280 |

Average

| | times | length |
|---|---|---|
| Short concatenation | 42621 | 42621 |
| Long concatenation | 5787 | 463024 |
| Short append | 30321220 | 30321220 |
| Long append | 3732421 | 298593296 |

String builder is faster, since in java the length of java is unchangeable. Whenever using "+" to concatenate string, a new string object will be made by adding the old string and new string. The old string will be deleted by garbage collector.

On the other hand, string builder is changeable, therefore adding a new string will directly adding into the string builder. The only extra word for string builder is converting it to string.

# 4 Sort

Insertion sort int array

Size = 98733 ==> Time = 1.001,

Size = 97411 ==> Time = 0.982,

Size = 97216 ==> Time = 1.009,

Size = 96106 ==> Time = 0.99,

Size = 99203 ==> Time = 1.012

Average

Size = 97733 ==> Time = 0.998

Insertion sort string array

Size = 16875==> Time = 0.976,

Size = 15198 ==> Time = 1.02,

Size = 16666 ==> Time = 0.989,

Size = 15399 ==> Time = 0.975,

Size = 16881 ==> Time = 1.017,

Average

Size = 16203 ==> Time = 0.995

Merge sort int array

Size = 7014082==> Time = 0.989,

Size = 6568290 ==> Time = 1.002,

Size = 7014082 ==> Time = 1.014,

Size = 7592313 ==> Time = 1.021,

Size = 6568290 ==> Time = 0.978,

Average

Size = 6951411 ==> Time = 1.0

Merge sort int array

Size = 1872526 ==> Time = 0.98,

Size = 1946160 ==> Time = 1.024,

Size = 1753519 ==> Time = 0.997,

Size = 1872526 ==> Time = 1.016,

Size = 1642072==> Time = 0.989,

Average

Size = 1817360 ==> Time = 1.0