Linnéuniversitetet
Kalmar Växjö

Report

Assignment 4

Author: Rashed Qazizada
Lnu Name: rq222ah
Course code: 1DV507

Time Measurements

Exercise 2: Concat with plus (+) v/s StringBuilder with method Append

The report deals with the results based on five runs of the program and in these runs we see the difference in the results between each of them. The table below shows the results for each run. The average values of the experiment is also presented.

The method Concatination.java deals with the number of short and long concatinations that are stored in a String data type. The current time is stored in a variable named 'time' using the nanoTime(). A while loop is used after storing the current time and this loop continues to concat or append strings if the difference of the current time and the time before is less than 1 second. When one second has passed, a result is ready to be printed with amount of concats/appends and the length of the string. This approach is used for the concat using plus and String Builder.

Execution 1	String Length	No.of Concatinations
Short concat	21303	21303
Short append	48121	48121
Long concat	173200	2165
Long append	402000	5025

Execution 2	String Length	No.of Concatinations
Short concat	25127	25127
Short append	46898	46898
Long concat	178880	2236
Long append	387360	4842

Execution 3	String Length	No.of Concatinations
Short concat	28869	28869
Short append	49844	49844
Long concat	206320	2579
Long append	400720	5009

Execution 4	String Length	No.of Concatinations
Short concat	29024	29024
Short append	49559	49559
Long concat	206000	2575
Long append	399200	4990

Execution 5	String Length	No.of Concatinations
Short concat	28327	28327
Short append	49767	49767
Long concat	191200	2390
Long append	399280	4991

Average	String Length	No.of Concatinations
Short concat	23124	23124
Short append	46362	46362
Long concat	174206	2362
Long append	389641	4927

The results suggest that a StringBuilder is faster than String concatenation with a plus (+). The reason is because in String Builder takes each char of the String and adds it to it's array within it. On the other hand String Concatination is much slower because it creates new String each time you concat, so it has to copy the old String and add the new String to it. This implies that it consumes a lot of memory.
Hence, String Builder is a much faster approach.

Exercise 3: SortingAlgorithms

The report deals with the results based on five runs of the program and in these runs we see the difference in the results between each of them. The table below shows the results for each run. The average values of the experiment is also presented.

The array chosen is given a size of 1000. This is from the SortingAlgorithms.java program in Assignment 3. The clock starts and the insertionSort() is called to sort the array. The array ius checked under one second. After one second has passed, the sorted integers will be returned.

Excecution 1	Sorted Integers/Strings
Integers	1181000
Strings	78000

Excecution 2	Sorted Integers/Strings
Integers	2565000
Strings	208000

Excecution 3	Sorted Integers/Strings
Integers	2390000
Strings	209000

Excecution 4	Sorted Integers/Strings
Integers	2441000
Strings	206000

Excecution 5	Sorted Integers/Strings
Integers	2511000
Strings	212000

Average	Sorted Integers/Strings
Integers	1579132
Strings	170300

The results portray the nnumber of integers and strings that can be sorted in 1 second. The results vary at each execution after the elements were sorted using the Insertion Sort and Merge sort. The Merge sort was however a VG Exercise in the Assignment 3.