Report

Description:

When doing an experiment like this it is important to remove as many possible factors that could impact the performance as possible. I made sure to close down other applications on my computer to minimize the risk of background processes affecting the experiment. I also tried to minimize the code needed for running an algorithm only one minute as well as keeping track of another variable. This code will affect the performance somewhat but is ultimately negligible. I ran each experiment 5 times and took the average of these to make sure individual results weren't anomalies. If I did an experiment for the first time I began with running the algorithm a few times to let the JVM optimization take place before the tests.

The reason why using a stringbuilder is much faster than concatenating is that each time you concatenate a string a new array have to be created and all the old plus the new characters have to be copied to that array. The stringbuilder however uses internally a resizable array similarly to the ArrayList class meaning that it doesn't have to create a new array and copy elements to it nearly as often.

Tables:

Comparing the **length** of a string after one minutes using concatenation and stringbuilder appending.

Test #	Short string concat	Long string concat	Short string append	Long string append
1	104 761	956 960	31 834 974	593 576 960
2	104 097	885 760	31 941 769	559 250 560
3	103 651	934 640	31 731 590	532 610 000
4	101 459	910 640	31 644 067	618 228 720
5	100 164	908 240	31 370 001	482 697 520
Average	**102 826**	**919 248**	**31 704 480**	**557 272 752**

Comparing the number of **concatenations** or stringbuilder **append** operation executed in one minute.

Test #	Short string concat	Long string concat	Short string append	Long string append
1	104 761	11 962	31 834 974	7 419 712
2	104 097	11 072	31 941 769	6 990 632
3	103 651	11 683	31 731 590	6 657 625
4	101 459	11 383	31 644 067	7 727 859
5	100 164	11 353	31 370 001	6 033 719
Average	**102 826**	**11 491**	**31 704 480**	**6 965 909**

The number of elements sorted during one minute.

Test #	Integer insertion sort	String insertion sort
1	12 497	8 541
2	12 617	8 575
3	12 475	8 307
4	12 617	8 426
5	12 431	8 129
Average	**12 527**	**8 396**