1DV507
Olga Christensen
oc222ba

Time Measurements Report

Exercise 3

Experiment:

The task was to compare how many concatenations of strings (containing one character or 80 characters) could be done in one second using Strings concatenation or String Builder.

First, I have implemented four methods, one for each test, that concatenate strings of two different sizes and using two different constructs.

To see how many iterations could be done in 1 second, I used a "for-loop" measuring the time of execution for a specific amount of concatenations. Then I manually changed the number of iterations so the time measurement would be approximately 1 second.

I have manually executed each test 15 times, discarding the results of the first 5 ones, making a note of the results of the last 10 tests.

What was interesting, when trying to pick a number of iterations for String Builder to concatenate long strings, I have found that when I reach a number somewhere between 1.074.800 and 1.074.700 the time measurements start to vary extremely. For example running a method 1.074.700 times takes under 800 milliseconds, but running it 1.074.800 times takes over 1 second. I guess it has something to do with the way String Builder allocates the memory when growing.

The results show that building a string with StringBuilder is much faster that using the string's plus operator. That is because the latter method creates new instances of StringBuilder every time a new string is added, opposite to creating only one instance when explicitly using the StringBuilder.

Results:

In approximately 1 second
N - number of iterations
L - the length of the final string

Method	N	L
1. str += short	27.000	27.000
2. str += long	21.000	168.000
3. sb.append(short)	54.000.000	54.000.000
4. sb.append(long)	1.074.750	85.980.000

Results table:

	1	2	3	4.1	4.2
1	1037	1038	1054	1163	769
2	1004	1044	1047	1165	746
3	1019	1050	1044	1159	753
4	1023	1047	1045	1159	767
5	1023	1048	1035	1143	766
6	1015	1059	1033	1149	765
7	1033	1042	1047	1156	768
8	1013	1043	1053	1155	765
9	1027	1081	1033	1151	797
10	1012	1079	1026	1150	765
Iterations	27.000	2.100	54.000.000	1.074.800	1.074.700
Length	27.000	168.000	54.000.000	85.984.000	85.976.000

Exercise 4

Experiment:

The task was to compare time measurements of two different sorting methods (insertion and merge sorting) of array containing integers or strings.

I used the same approach as in exercise 3:

- Created 4 methods to test time measurements of each sorting algorithm;
- Manually set the size of an array to be tested, so the each method would take around 1 second to run;
- Ran each test 15 times, using only last 10 cases in my statistics;
- Drew conclusions.

Results:

The findings of the experiment show that in approximately 1 second we can sort

	Method	Array size	Parameter
1.	Insertion	43.000	integers
2.	Merge	3.010.000	integers
3.	Insertion	9.500	strings
4.	Merge	700.000	strings

1DV507
Olga Christensen
oc222ba

Results table:

	1	2	3	4
1	1050	1061	1065	1012
2	1068	1053	1057	1023
3	1068	1050	1049	1026
4	1048	1060	1052	1031
5	1060	1055	1058	1019
6	1051	1052	1043	1022
7	1062	1048	1049	1016
8	1058	1049	1050	1030
9	1059	1052	1055	1027
10	1049	1050	1041	1021
N	43.000	3.010.000	9.500	700.000