2b) Starting from 1000 times of concatenation operations, StringBuilder approaches is bettern than the other.

# of concatenation operations	Approach 1 (String)	Approach 2 (StringBuilder)
10	0 ms	0 ms
100	0 ms	0 ms
1000	2 ms	0 ms
10000	49 ms	1 ms
100000	2042 ms	2 ms

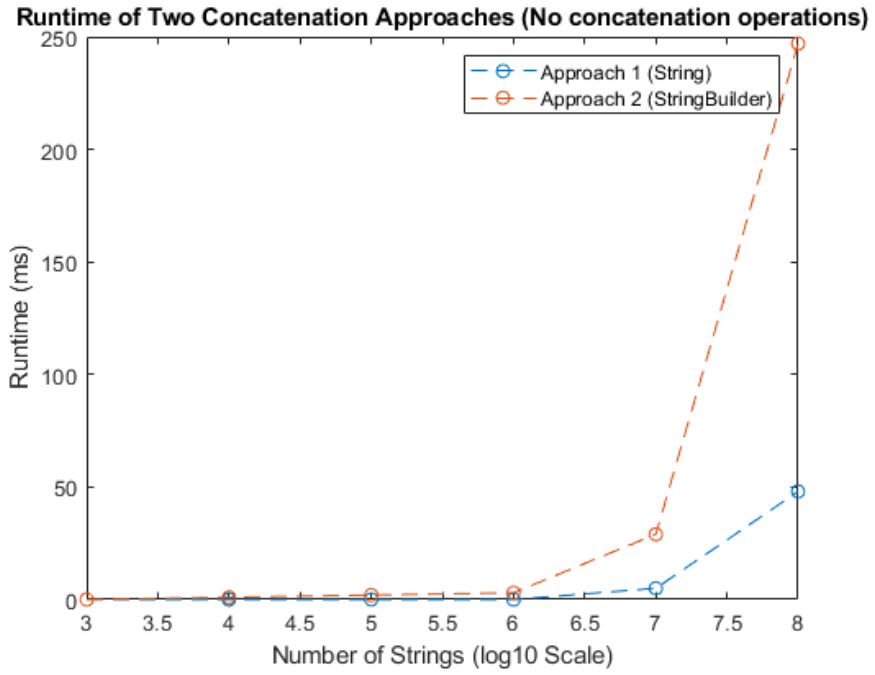
Result for CompareLargeStringConcatenation

3a) Result for CompareManySmallStringsConcatenation

# of Strings / Operations	0		1		2		3		4	
Type of Approach	A1	A2	A1	A2	A1	A2	A1	A2	A1	A2
1000	0 ms	0 ms	1 ms	0 ms	1 ms	1 ms	0 ms	1 ms	0 ms	1 ms
10000	0 ms	1 ms	0 ms	1 ms	1 ms	1 ms	2 ms	1 ms	3 ms	0 ms
100000	0 ms	2 ms	4 ms	1 ms	3 ms	1 ms	2 ms	2 ms	6 ms	1 ms
1000000	0 ms	3 ms	8 ms	8 ms	24 ms	5 ms	20 ms	5 ms	47 ms	12 ms
10000000	5 ms	29 ms	29 ms	55 ms	110 ms	41 ms	222 ms	52 ms	295 ms	66 ms
100000000	48 ms	247 ms	264 ms	315 ms	1076 ms	387 ms	1985 ms	519 ms	2939ms	659 ms

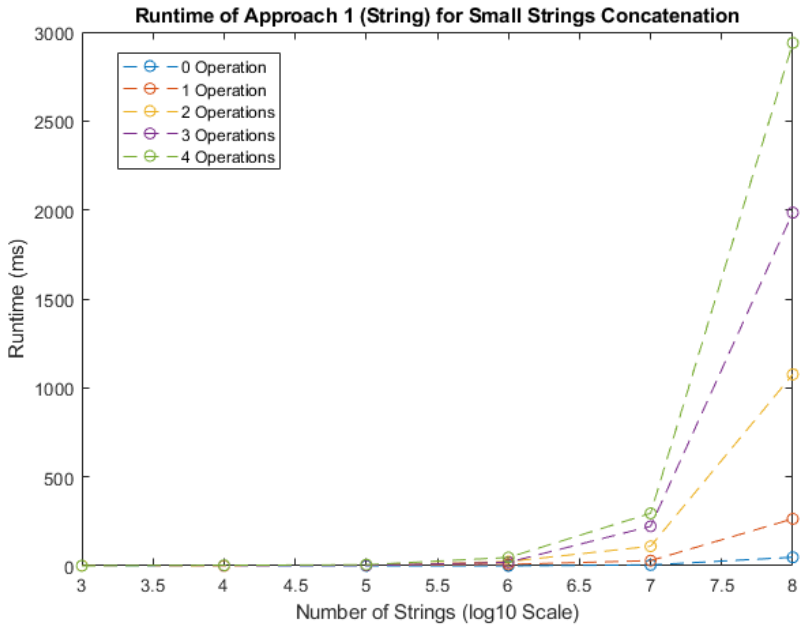
(A1: Approach 1 (String). A2: Approach 2 (StringBuilder))

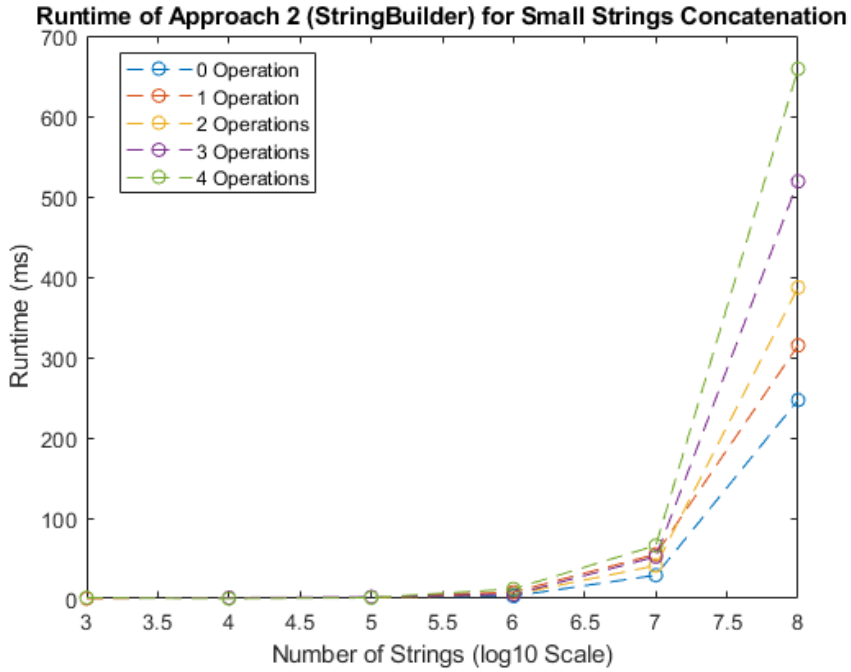
3b) Approach 1 is the best when there is no concatenation operations at all on the strings. Starting from 100000 Strings, the difference between the approaches start to show.

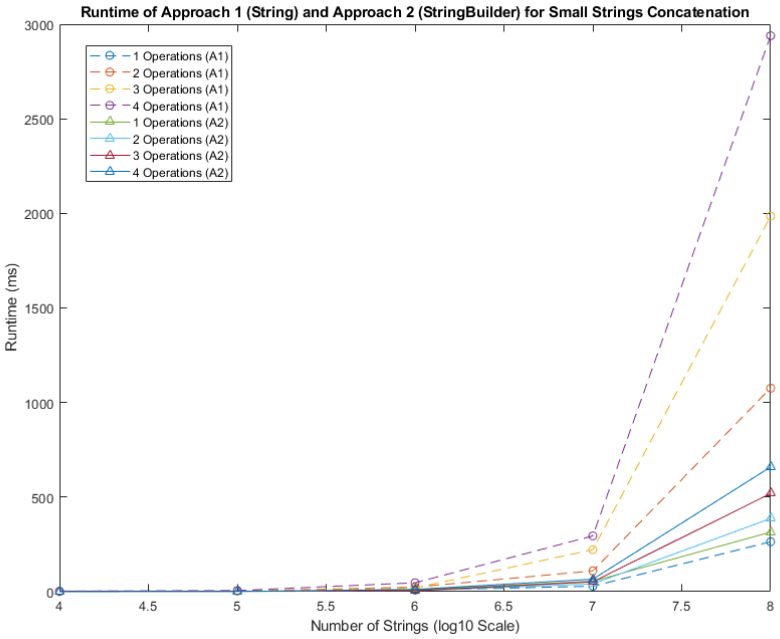


3c) The number of concatenation operations need to be two to guarantee that Approach 2 (StringBuilder) has faster runtime than Approach 1 (String) for each test with different number of Strings. As the number of Strings increases,

it amplifies the difference between two approaches, where 1000000 Strings is critical point that starts to show significant difference. The result is consistent with the result in 2b). Approach 1 experiences performance issue as the number of concatenation operations increasem, and the increase in number of Strings amplifies the performance issue further.






Runtime of Approach 1 (String) and Approach 2 (StringBuilder) for Small Strings Concatenation
1 Operations (A1)
2 Operations (A1)
3 Operations (A1)
4 Operations (A1)
1 Operations (A2)
2 Operations (A2)
3 Operations (A2)
4 Operations (A2)
Runtime (ms)
Number of Strings (log10 Scale)
0
500
1000
1500
2000
2500
3000
4
4.5
5
5.5
6
6.5
7
7.5
8