Glasswall Technical Test

Max Harris

To benchmark my choices in designs, I created 8 different versions of the CountOccurrences method.

First of all, let's outline how many occurrences of each predefined word, exists in each text file. It must be noted that I noticed the 'Bacon' parameter began with an uppercase. I made sure that the CountOccurence method was case insensitive, as other occurrences of bacon appeared in the text files, all in lower case.

These occurrences can be found below:

	Bacon	**pork**	**prosciutto**
Bacon10.txt	7	32	11
Bacon25.txt	18	80	19
Bacon50.txt	32	162	39

I tested my methods by also processing the data with a 3rd party source (https://string-functions.com/countsubstrings.aspx), and cross checking my outputs with the 3rd parties.

When designing my methods to count the occurrences of a substring within a string, I opted for a for loop or a for each loop. This was due to these being the fastest loops available. In terms of iterating through each line in the lines array, I used one of two options. These options were Regex and using the inbuilt indexOf method for strings to find the occurrence of the words in the string lines of the text files. Another design I tested was multithreaded variants of each of my methods to test whether sequential, or concurrent processes were more efficient in terms of speed.

I have provided a screenshot of my benchmarks on the next page, following on from that, I will analyse my findings.

Method	SearchValue	FileToRead	Mean	Error	StdDev	Gen 0	Gen 1	Allocated
CountOccurrences	Bacon	Files/Bacon10.txt	60.31 us	0.761 us	0.711 us	4.8218	0.2441	40 KB
CountOccurrences2	Bacon	Files/Bacon10.txt	64.11 us	0.683 us	0.606 us	5.2490	0.3662	43 KB
CountOccurrences3	Bacon	Files/Bacon10.txt	51.42 us	0.373 us	0.349 us	3.4790	0.2441	29 KB
CountOccurrences4	Bacon	Files/Bacon10.txt	65.94 us	0.333 us	0.311 us	3.9063	0.2441	32 KB
CountOccurrences5	Bacon	Files/Bacon10.txt	56.76 us	0.845 us	0.706 us	4.8218	0.3052	40 KB
CountOccurrences6	Bacon	Files/Bacon10.txt	64.64 us	0.778 us	0.650 us	5.2490	0.2441	42 KB
CountOccurrences7	Bacon	Files/Bacon10.txt	66.97 us	1.100 us	1.029 us	3.9063	0.2441	32 KB
CountOccurrences8	Bacon	Files/Bacon10.txt	52.65 us	0.147 us	0.130 us	3.4790	0.2441	29 KB
CountOccurrences	Bacon	Files/Bacon25.txt	92.59 us	1.420 us	1.259 us	10.3760	1.2207	85 KB
CountOccurrences2	Bacon	Files/Bacon25.txt	92.39 us	0.713 us	0.666 us	11.2305	1.3428	90 KB
CountOccurrences3	Bacon	Files/Bacon25.txt	81.46 us	1.201 us	1.123 us	7.3242	0.9766	60 KB
CountOccurrences4	Bacon	Files/Bacon25.txt	165.41 us	2.005 us	1.876 us	8.0566	0.9766	66 KB
CountOccurrences5	Bacon	Files/Bacon25.txt	94.31 us	1.127 us	1.054 us	10.3760	1.2207	85 KB
CountOccurrences6	Bacon	Files/Bacon25.txt	92.15 us	1.005 us	0.891 us	11.2305	1.3428	90 KB
CountOccurrences7	Bacon	Files/Bacon25.txt	174.71 us	3.392 us	5.078 us	8.0566	0.9766	66 KB
CountOccurrences8	Bacon	Files/Bacon25.txt	81.08 us	0.618 us	0.548 us	7.3242	0.9766	60 KB
CountOccurrences	Bacon	Files/Bacon50.txt	159.88 us	3.156 us	3.991 us	19.2871	3.6621	159 KB
CountOccurrences2	Bacon	Files/Bacon50.txt	123.80 us	1.674 us	1.566 us	20.5078	3.9063	163 KB
CountOccurrences3	Bacon	Files/Bacon50.txt	128.18 us	1.731 us	1.620 us	13.1836	2.9297	109 KB
CountOccurrences4	Bacon	Files/Bacon50.txt	285.48 us	1.193 us	1.057 us	14.1602	2.9297	115 KB
CountOccurrences5	Bacon	Files/Bacon50.txt	153.44 us	1.865 us	1.744 us	19.2871	3.6621	159 KB
CountOccurrences6	Bacon	Files/Bacon50.txt	126.06 us	2.186 us	2.045 us	20.5078	3.9063	163 KB
CountOccurrences7	Bacon	Files/Bacon50.txt	289.17 us	3.203 us	2.996 us	14.1602	2.9297	115 KB
CountOccurrences8	Bacon	Files/Bacon50.txt	127.48 us	0.637 us	0.564 us	13.1836	2.9297	109 KB
CountOccurrences	pork	Files/Bacon10.txt	63.70 us	1.123 us	1.051 us	7.4463	0.3662	61 KB
CountOccurrences2	pork	Files/Bacon10.txt	69.57 us	0.539 us	0.478 us	8.0566	0.3662	64 KB
CountOccurrences3	pork	Files/Bacon10.txt	57.18 us	0.545 us	0.510 us	4.1504	0.2441	34 KB
CountOccurrences4	pork	Files/Bacon10.txt	85.05 us	0.419 us	0.350 us	4.6387	0.2441	38 KB
CountOccurrences5	pork	Files/Bacon10.txt	63.52 us	1.244 us	1.573 us	7.4463	0.3662	61 KB
CountOccurrences6	pork	Files/Bacon10.txt	69.49 us	0.576 us	0.539 us	8.0566	0.3662	64 KB
CountOccurrences7	pork	Files/Bacon10.txt	84.85 us	0.553 us	0.518 us	4.6387	0.2441	37 KB
CountOccurrences8	pork	Files/Bacon10.txt	58.12 us	0.102 us	0.091 us	4.1504	0.2441	34 KB
CountOccurrences	pork	Files/Bacon25.txt	112.41 us	0.985 us	0.873 us	17.2119	1.8311	141 KB
CountOccurrences2	pork	Files/Bacon25.txt	100.56 us	0.500 us	0.467 us	18.6768	2.0752	147 KB
CountOccurrences3	pork	Files/Bacon25.txt	97.38 us	0.263 us	0.234 us	8.9111	1.0986	74 KB
CountOccurrences4	pork	Files/Bacon25.txt	215.33 us	1.460 us	1.219 us	9.7656	1.2207	80 KB
CountOccurrences5	pork	Files/Bacon25.txt	113.62 us	1.083 us	1.013 us	17.2119	1.8311	141 KB
CountOccurrences6	pork	Files/Bacon25.txt	102.25 us	0.555 us	0.520 us	18.6768	2.0752	147 KB
CountOccurrences7	pork	Files/Bacon25.txt	216.85 us	1.383 us	1.293 us	9.7656	1.2207	80 KB
CountOccurrences8	pork	Files/Bacon25.txt	97.67 us	1.251 us	1.109 us	8.9111	1.0986	74 KB
CountOccurrences	pork	Files/Bacon50.txt	207.07 us	4.044 us	4.153 us	33.4473	5.8594	275 KB
CountOccurrences2	pork	Files/Bacon50.txt	138.73 us	1.287 us	1.141 us	35.6445	5.8594	280 KB
CountOccurrences3	pork	Files/Bacon50.txt	166.50 us	2.859 us	2.534 us	16.8457	3.6621	138 KB
CountOccurrences4	pork	Files/Bacon50.txt	360.56 us	4.208 us	3.514 us	18.0664	3.4180	145 KB
CountOccurrences5	pork	Files/Bacon50.txt	208.53 us	2.093 us	1.958 us	33.4473	5.8594	275 KB
CountOccurrences6	pork	Files/Bacon50.txt	143.42 us	1.278 us	1.196 us	35.6445	5.8594	280 KB
CountOccurrences7	pork	Files/Bacon50.txt	369.76 us	3.426 us	3.205 us	18.0664	3.4180	145 KB
CountOccurrences8	pork	Files/Bacon50.txt	160.00 us	2.175 us	2.034 us	16.8457	3.6621	138 KB
CountOccurrences	prosciutto	Files/Bacon10.txt	59.62 us	0.232 us	0.194 us	5.1270	0.3052	42 KB
CountOccurrences2	prosciutto	Files/Bacon10.txt	65.00 us	0.550 us	0.514 us	5.6152	0.2441	45 KB
CountOccurrences3	prosciutto	Files/Bacon10.txt	51.02 us	0.173 us	0.161 us	3.6011	0.2441	30 KB
CountOccurrences4	prosciutto	Files/Bacon10.txt	64.60 us	0.356 us	0.315 us	3.9063	0.2441	33 KB
CountOccurrences5	prosciutto	Files/Bacon10.txt	59.55 us	0.333 us	0.311 us	5.1270	0.3052	42 KB
CountOccurrences6	prosciutto	Files/Bacon10.txt	65.41 us	0.672 us	0.629 us	5.6152	0.3662	45 KB
CountOccurrences7	prosciutto	Files/Bacon10.txt	66.10 us	0.527 us	0.493 us	3.9063	0.2441	32 KB
CountOccurrences8	prosciutto	Files/Bacon10.txt	49.63 us	0.448 us	0.397 us	3.6011	0.2441	30 KB
CountOccurrences	prosciutto	Files/Bacon25.txt	98.42 us	0.321 us	0.285 us	10.3760	1.2207	85 KB
CountOccurrences2	prosciutto	Files/Bacon25.txt	91.15 us	0.659 us	0.617 us	11.3525	1.3428	90 KB
CountOccurrences3	prosciutto	Files/Bacon25.txt	75.68 us	0.398 us	0.353 us	7.3242	0.9766	60 KB
CountOccurrences4	prosciutto	Files/Bacon25.txt	126.69 us	1.919 us	1.795 us	8.0566	0.9766	66 KB
CountOccurrences5	prosciutto	Files/Bacon25.txt	95.50 us	0.784 us	0.655 us	10.3760	1.2207	85 KB
CountOccurrences6	prosciutto	Files/Bacon25.txt	95.98 us	0.941 us	0.880 us	11.3525	1.4648	90 KB
CountOccurrences7	prosciutto	Files/Bacon25.txt	128.24 us	2.525 us	2.807 us	8.0566	0.9766	66 KB
CountOccurrences8	prosciutto	Files/Bacon25.txt	76.59 us	0.145 us	0.128 us	7.3242	0.9766	60 KB
CountOccurrences	prosciutto	Files/Bacon50.txt	149.10 us	0.310 us	0.275 us	19.5313	3.9063	161 KB
CountOccurrences2	prosciutto	Files/Bacon50.txt	123.99 us	0.553 us	0.491 us	20.7520	3.9063	165 KB
CountOccurrences3	prosciutto	Files/Bacon50.txt	108.87 us	0.422 us	0.395 us	13.5498	3.1738	111 KB
CountOccurrences4	prosciutto	Files/Bacon50.txt	250.48 us	2.884 us	2.408 us	14.1602	2.9297	116 KB
CountOccurrences5	prosciutto	Files/Bacon50.txt	149.22 us	0.636 us	0.564 us	19.5313	3.9063	161 KB
CountOccurrences6	prosciutto	Files/Bacon50.txt	127.27 us	0.596 us	0.528 us	20.7520	3.6621	165 KB
CountOccurrences7	prosciutto	Files/Bacon50.txt	239.55 us	1.074 us	0.952 us	14.4043	2.9297	116 KB
CountOccurrences8	prosciutto	Files/Bacon50.txt	116.70 us	0.290 us	0.257 us	13.5498	3.1738	111 KB

Findings

To analyse the findings from my benchmarks, I created a table with rows for each method, and 2 columns, one being speed, and the other memory efficiency.
There were 9 tests in total for each method, these tests comprising of each combination of input parameters (the SearchValue and the TextFile).
For each test, I marked down which method came on top for that particular case, by inputting that tests number into the table, for both speed and memory efficiency. If two methods had the same result, they would both get the tests number noted into the table. This can be found below:

	Speed	**Memory Efficiency**
CountOccurrence		
CountOccurrence2	3, 6	
CountOccurrence3	1, 4, 5, 8, 9	1, 2, 3, 4, 5, 6, 7, 8, 9
CountOccurrence4		
CountOccurrence5		
CountOccurrence6		
CountOccurrence7		
CountOccurrence8	2, 7	1, 2, 3, 4, 5, 6, 7, 8, 9

It can be seen that overall, the method **CountOccurrence3,** came out on top overall. However in terms of memory efficiency, both **CountOccurrence3** and **CountOccurrence8** came to a tie.

Due to these findings, I opted for **CountOccurrence3** as my chosen method to use for the CountOccurrence method required for this test.

CountOccurrence3 surprised me as my research suggested that this method would not be the overall winner as it used a foreach loop with Regex. Sources online, from blog posts to StackOverflow mentioned that for loops were faster than foreach loops, and the IndexOf method was faster than Regex.

A screenshot of this method can be found below:

```
[Benchmark]
public int CountOccurrences3()
{
    int count = 0;
    GlobalSetup();

    foreach (string line in Lines)
    {
        count += Regex.Matches(line, SearchValue, RegexOptions.IgnoreCase).Count;
    }
    return count;
}
```

This test has made it clear that the use case will be largely what the answer depends on when developing the most efficient method for a scenario.

All of my methods have been commented on my GitHub fork with the only uncommented one being my chosen one, **CountOccurrence3**. This is so you may see my working and thought processes when approaching this challenge.

Thank you for the opportunity, this was a lot of fun and further developed my learning.