exactSearch() worst possible scenario

Code	Times done	Cost	Total Cost
`int totalMatches = 0;`	n/m	1	O(1)
`for(int i = 0; i < data.size() - pattern.size(); i++)`	n-m	n	O(1+n)
`for (int j = 0; j < pattern.size(); j++)`	m	m	O((1+n)m)
`if(pattern.get(j) == data.get(i+j))`	m	m	O((1+n)2m)
`matchingChar++;`	m	m	O((1+n)3m)
`else break;`	0	0	
`if(matchingChar == patternString.length())`	1	1	O((1+n)3m)
`totalMatches++;`	1	1	O((1+n)3m)
`UI.println("Found at "+i+" : "+pattern);`	1	1	O((1+n)3m)

The total cost of this algorithm is O((1+n)3m). this becomes an overall cost of O(n) for the algorithm

approximateSearch() worst possible scenario (assuming all 5/5 errors are used each case)

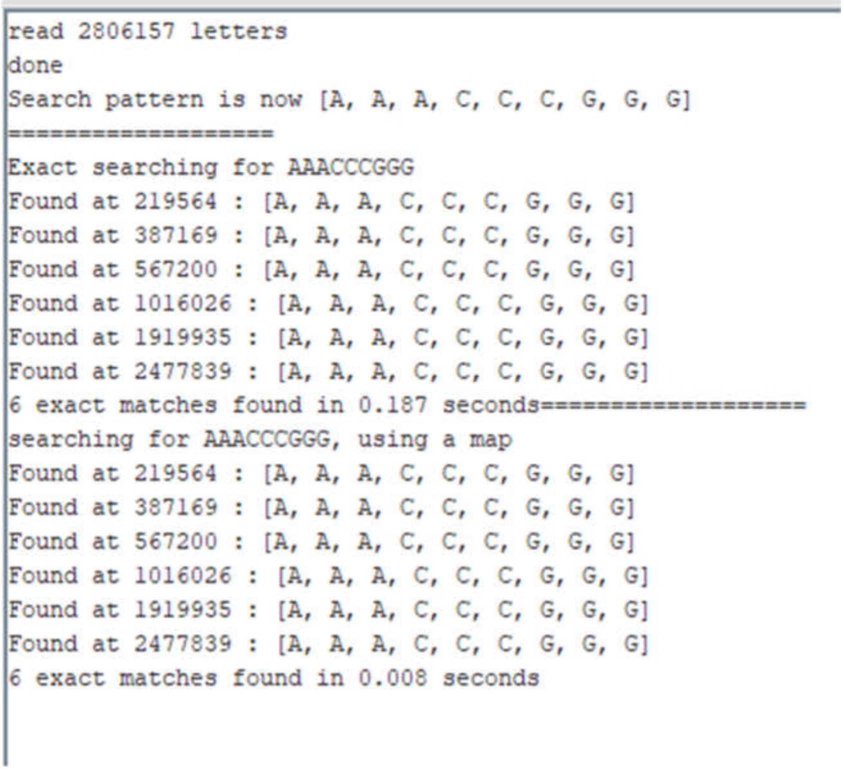
Code	Times done	Cost	Total Cost
`int totalMatches = 0;`	n/m	1	O(1)
`for(int i = 0; i < data.size() - pattern.size(); i++)`	n-m	n	O(1+n)
`int matchingChar = 0;`	m	1	O((1+n)m)
`int errors = 0;`	m	1	O((1+n)m)
`for (int j = 0; j < pattern.size(); j++)`	m	m	O((1+n)m)
`if(pattern.get(j) == data.get(i+j))`	m	m	O((1+n)2m)
`matchingChar++;`	m	m	O((1+n)3m)
`else error++;`	5	1	O((1+n)3m)
`if(errors > maxErrors) break;`	1	1	O((1+n)3m)
`if(matchingChar >= patternString.length()-maxErrors)`	1	1	O((1+n)3m)
`totalMatches++;`	1	1	O((1+n)3m)
`UI.println("Found at "+i+" : "+pattern);`	1	1	O((1+n)3m)

The Overall cost of this algorithm is still O(n), just like the first one. However this algorithm will clearly have more outputs because of the nature of allowing for errors, and therefor will take a longer time to complete. However if both algorithms are to output the same number of positions, then the only difference between the two is that approximateSearch() functions keeps track of errors, but that is only ever a maximum of 5 extra steps per loop, so the difference is marginal.

Challenge

When searching using the map, rather than searching for the full length of the string you immediately reduce the possible searches by searching through only the strings that begin with the same 5 characters as the key in the map. This can drastically speed up a search, since rather than searching through n-m strings, you only need to search through the known number within the map.

Doing this however is very costly in time and in memory, to create the map each possible key must be searched through the map and all its occurrences must be noted down. In our case there is a total of 1024 different keys of length 5, and each of these on average has 4000 occurrences. Because of this, it takes a large amount of time to set up the map. However once setup the use of the map drastically increases the speed of searches (in my tests by a factor of 10).



```
read 2806157 letters
done
Search pattern is now [A, A, A, C, C, C, G, G, G]
===================
Exact searching for AAACCCGGG
Found at 219564 : [A, A, A, C, C, C, G, G, G]
Found at 387169 : [A, A, A, C, C, C, G, G, G]
Found at 567200 : [A, A, A, C, C, C, G, G, G]
Found at 1016026 : [A, A, A, C, C, C, G, G, G]
Found at 1919935 : [A, A, A, C, C, C, G, G, G]
Found at 2477839 : [A, A, A, C, C, C, G, G, G]
6 exact matches found in 0.187 seconds===================
searching for AAACCCGGG, using a map
Found at 219564 : [A, A, A, C, C, C, G, G, G]
Found at 387169 : [A, A, A, C, C, C, G, G, G]
Found at 567200 : [A, A, A, C, C, C, G, G, G]
Found at 1016026 : [A, A, A, C, C, C, G, G, G]
Found at 1919935 : [A, A, A, C, C, C, G, G, G]
Found at 2477839 : [A, A, A, C, C, C, G, G, G]
6 exact matches found in 0.008 seconds
```

For an average consumer, searching with a map is defiantly NOT worth the time and memory to set up. However a company with a server running 24/7 that will constantly be querying the data and looking for strings would defiantly find worth in using a map to speed up the searching.

As you increase the number of characters within the sequence, you greatly increase the total number of possible combinations, by doing this you increase the total memory usage of the map as it will now have to store more individual pieces of data.

This algorithm would be of no use for doing approximate searches since it relies on the first characters of the string being exactly correct. Because of this you wold never be able to get the full range of possible sequences available when searching with a degree of error.