1. Given -

* Size of characters = n
* Where k : order and S : alphabet size

Time Complexity:

* My program generates the substring (of k and k+1 chars of the original string) in a constant time operation using the generate() method (It does uses substring function, so it depends on it as well but in my analysis, I am avoiding it).
* To generate alphabet size, the gets() method performs in linear time operation.
* The Laplace method is computed in the linear time operation because of the size() method. {it is given that ST performs per operation in log(n) time. Thus, max(log(n) + n) = O(n)} . Though not that it matters but I think that because I am using HashMap, contains could be a constant time operation.
* **The constructor iterates over a for loop {linear}, calls to contains () method {given logarithmic operation}, calls generate {constant}} Thus, the running time of the constructor is O(n log(n)).**
* The other methods like – likelihood is linear because it calls Laplace() method. The totalLikelihood() runs in as {O(n)\*log(n) for the priority queue insertion \*o(n) for the laplace 🡪 O( log(n))}.
* getK() and getone() methods are constant.

Thus, my program would take O( log(n)) this much time to build the MarkovModel.

Space Complexity

* My program has two collections – Priority Queue and HashMap (ST). Assuming both requires space proportional to the number of keys.

The number of keys would be : . Thus, for the symbol table it would be . For the priority Queue, I am only storing Thus it would be

Thus, my programs take space o(2\* ).

1. To compute just the log likelihood of a new string of length n, under this model would be linear (O( log(n)). Because the outer- for loop is linear, then to store the string generated in here to a PQ for later use in the BestModel.java is log(n) and the last call to likelihood() method is linear (likelihood is linear because it calls laplace() and laplace calls getS() which is a linear time function). Thus, total turns out to be (O( log(n)).

If I ignore the time to store the strings in a PQ, then to compute the log likelihood of the given string would be .

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with low confidence

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

1. This one is not a question asked in the memo.txt - but I wanted to include my results for the last case on the gradescope. Also, to compare the results with the solution provided by the instructor – I include hers as well.

Graphical user interface, text

Description automatically generated