This was for me the hardest homework yet in the class. I hope that the project is more manageable and I will be sure to start it early.

1. I completed the function ***bwt*** it is attached in *bwtSeq.py* and the associated output is *bwtSample1.fa*.
2. I completed the function ***reverseBwt*** it is attached in *reverseBwt.py.* I am having some trouble tonight with the associated output file. I have verified the function works with toy examples, but the runtime for the *bwtHomoSapiens.fa* is very long. I will upload later if it completes tonight.

Note: Used two formula types that were very slow even with the toy examples before moving to implement the solution as shown on the BWT wiki page. This is still slow with the homo sapiens file.

3.1 I completed the function ***fullindexFM***. It is attached in ***fullindexFMv2.py***. I did not figure out how to get the reference positions array without using reverseBWT to get BWT and then to get the reference positions array from T (original sequence). Therefore I do not have an output file but I verified the correctness of this function using toy examples.

3.2 I completed the ***hybridFM*** function, but could not test the implementation. It is attached in *HybridFM.py.*

3.3 I did not complete the ***checkpointingFM*** function.

Theoretically: (1) loop through BWT (L) with checkpoints k-spaced apart. Then instead

of having to scan through all of BWT/L you can can go to nearest checkpoint and adjust

the checkpoint value by characters seen in BWT/L

3.4 Because I did not complete 3.2, 3.3, and the full output from 3.1 I could not compare the methods directly. I can take a theoretical approach though.

Checkpoints utilize O(m) space where m is in 3.1 the number of integers and in 3.2 / 3.3 the number of checkpoints.

For checkpoints if checkpoints are spaced O91) distance apart then lookups are O(1).

Theoretically FM can be brought to linear time.