4)

a)

1	2	3	4	5
Ø	B_2	A_2	B_1	A_1
A_1	Ø	B_2	A_2	B_1
A_2	B_1	A_1	Ø	B_2
B_1	A_1	Ø	B_2	A_2
B_2	A_2	B_1	A_1	Ø

B A B A Ø

b)

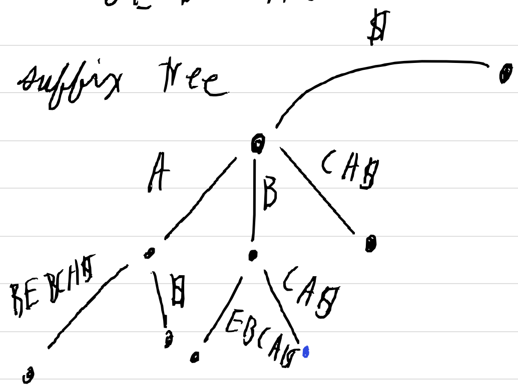
Ø	C_2	G_3	G_2	A_2	T_2	T_1	G_1	A_1	C_1
A_1	C_1	Ø	C_2	G_3	G_2	A_2	T_2	T_1	G_1
A_2	T_2	T_1	G_1	A_1	C_1	Ø	C_2	G_3	G_2
C_1	Ø	C_2	G_3	G_2	A_2	T_2	T_1	G_1	A_1
C_2	G_3	G_2	A_2	T_2	T_1	G_1	A_1	C_1	Ø
G_1	A_1	C_1	Ø	C_2	G_3	G_2	A_2	T_2	T_1
G_2	A_2	T_2	T_1	G_1	A_1	C_1	Ø	C_2	G_3
G_3	G_2	A_2	T_2	T_1	G_1	A_1	C_1	Ø	C_2
T_1	G_1	A_1	C_1	Ø	C_2	G_3	G_2	A_2	T_2
T_2	T_1	G_1	A_1	C_1	Ø	C_2	G_3	G_2	A_2

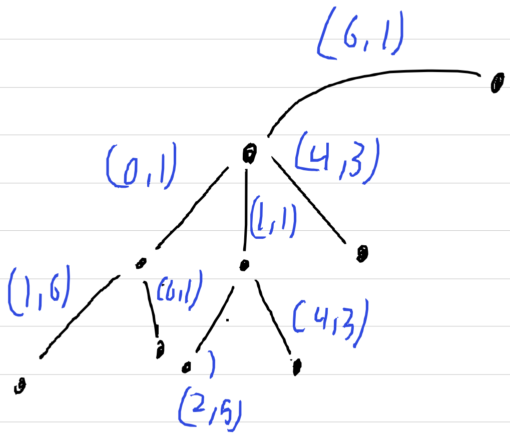
C G G A T T G A C Ø

0 1 2 3 4 5 6

5) T = A B E B C A $

a) suffix tree





b) Suffix Array

6	$
5	A $
0	A B E B C A $
3	B C A $
1	B E B C A $
4	C A $
2	E B C A $

P = B A A C B

A B E B C A $

c) BWT

$ A B E B C A
A $ A B E B C
A B E B C A $
B C A $ A B E
B E B C A $ A
C A $ A B E B
E B C A $ A B

6)

	A	B	B	A	C	A	B	\$
i:	0	1	2	3	4	5	6	7
o:	2	6	5	3	7	1	4	0

7)

	C	B	A	A	C	C	\$
i:	0	1	2	3	4	5	6
o:	5	3	1	2	6	4	0

8) $\$ab$ - no $\quad$ $\$$ B never row 0

$\$zg$ - no $\quad$ $\$$ B never row 0

$$n\$y \Rightarrow \begin{bmatrix} \$ & y & n \\ y & n & \$ \\ n & \$ & y \end{bmatrix} \Rightarrow y < n \Rightarrow \text{no}$$

$$gx\$ \Rightarrow \begin{bmatrix} \$ & x & g \\ g & \$ & x \\ x & g & \$ \end{bmatrix} \rightarrow xg\$ \Rightarrow \text{yes}$$

$$y\$n \rightarrow \begin{bmatrix} \$ & n & y \\ n & y & \$ \\ y & \$ & n \end{bmatrix} \rightarrow n\,y\,\$ \Rightarrow \text{yes}$$

$$yn\$ \rightarrow \begin{bmatrix} \$ & n & y \\ y & \$ & n \\ n & y & \$ \end{bmatrix} \Rightarrow y < n \Rightarrow \text{no}$$

4) yes

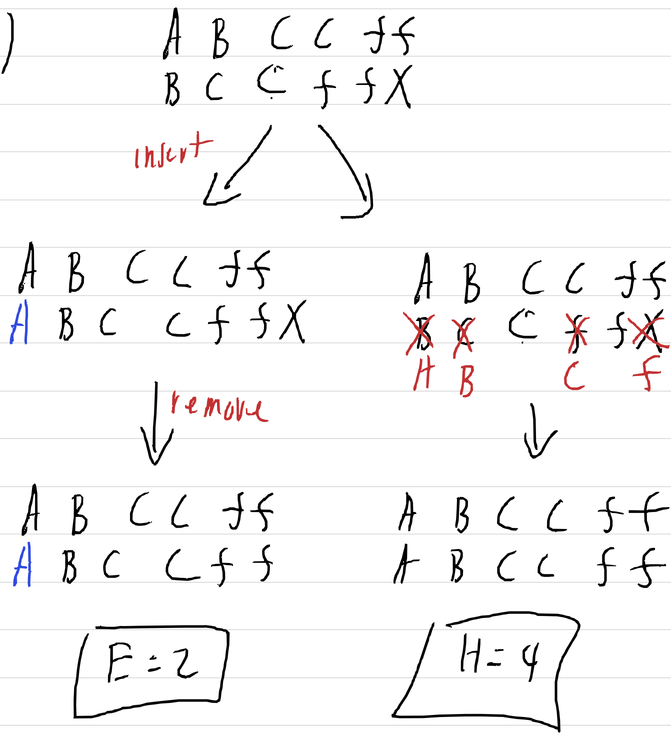
A B C D

A B C D ~~E~~ $\Rightarrow E=2$

for $E=1$ we need
mis matched length
$\Rightarrow H=\infty$

No it is not possible because
for edit distance to be < hamming
it means we bennefit
from insert / delete
but for an insert / delete
to Edit match in 1 Δ the
original strings must have
different length which would
imply hamming $=\infty$

10)



11)

The order of sorted suffixes and sorted rotations of $ terminated string are equivilant because the sorting of our rotations never will look past the $. This is because $ is the first letter lexiographically (by definition) in our alphabet and there only ever exists one in a string. This means that there exists only one $ for every index for every sorted rotation and thus we never have to compare $ to $ and never need a subsequent tiebreaker afterwards. This means that every sorted rotation IS a suffix for the purpose of sort comparisons and the set of all roations excluding content after to the $ is identical to the set of all suffixes. Thus the sorted ordering of these two formats is identical and yeilds a datastrucutre representing the same offests in the original string.