Burrows-Wheeler Transform (BWT)

Question 1:
Perform a complete Burrows-Wheeler encoding on the following string: ABCABC.

Solution:

Step1: Add a dollar sign to the end of the string then write down all possible shifts of the string.

ABCABC$
BCABC$A
CABC$AB
ABC$ABC
BC$ABCA
C$ABCAB
$ABCABC

Step 2: Sort the strings in alphabetical order (in ascii $ < A so it comes first in the alphabet)

$ABCABC
ABC$ABC
ABCABC$
BC$ABCA
BCABC$A
C$ABCAB
CABC$AB

The resulting string is CC$AABB

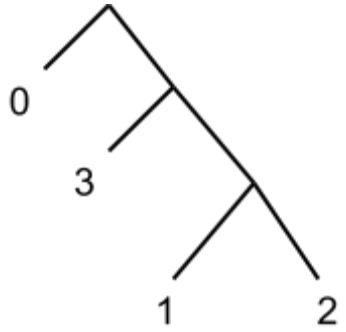
Step 3: Take the string made by the last column and use the MTF algorithm as follows.

CC$AABB

Encoded So Far	Encoding	Dictionary at Beginning of Step	Dictionary at End of Step	Description of Step
C	3	$ABC	C$AB	Encode C as 3 since it is in position 3 of the dictionary. Then move C to the front (MTF).
CC	3,0	C$AB	C$AB	Encode C as 0 since it is in position 0 of the dictionary.
CC$	3,0,1	C$AB	$CAB	Encode $ as 1 since it is in position 1 of the dictionary. Then MTF.
CC$A	3,0,1,2	$CAB	A$CB	Encode A as 2 since it is in position 2 of the dictionary. Then MTF.
CC$AA	3,0,1,2,0	A$CB	A$CB	Encode A as 0 since it is in position 0 of the dictionary.
CC$AAB	3,0,1,2,0,3	A$CB	BA$C	Encode B as 3 since it is in position 3 of the dictionary. Then MTF.
CC$AABB	3,0,1,2,0,3,0	BA$C	BA$C	Encode B as 0 since it is in position 0 of the dictionary.

Step 4:
So our current encoding is 3,0,1,2,0,3,0 and now we calculate frequencies and create a Huffman encoding.
0 = 3/7
1 = 1/7
2 = 1/7
3 = 2/7

Here is what the final Huffman encoding tree will look like:



Therefore the Huffman encodings are:
0 = 0
3 = 10
1 = 110
2 = 111

And our encoding of 3,0,1,2,0,3,0 is: 1001101110100

Now the students should be able to run this algorithm backwards and go from 1001101110100 to ABCABC given only the Huffman encodings and the alphabet ($,A,B,C).

The trickiest part of this (and the most commonly asked on exams) is the last step going from the BWT string CC$AABB to the original ABCABC$

Question 2:
Given the following string which was the last column in the BWT find the original string: CMIAG$

Solution:

<table>
<tr><th>Reverse Burrows-Wheeler Transform</th><th>Description of Step</th></tr>
<tr><td>
<table>
<tr><td>C</td><td>$</td></tr>
<tr><td>M</td><td>A</td></tr>
<tr><td>I</td><td>C</td></tr>
<tr><td>A</td><td>G</td></tr>
<tr><td>G</td><td>I</td></tr>
<tr><td>$</td><td>M</td></tr>
</table>
</td><td>Sort the given string then append the original to the beginning.
(The previous sorted result is shown in blue).</td></tr>
<tr><td>
<table>
<tr><td>C</td><td>$</td><td>M</td></tr>
<tr><td>M</td><td>A</td><td>G</td></tr>
<tr><td>I</td><td>C</td><td>$</td></tr>
<tr><td>A</td><td>G</td><td>I</td></tr>
<tr><td>G</td><td>I</td><td>C</td></tr>
<tr><td>$</td><td>M</td><td>A</td></tr>
</table>
</td><td>Sort the previous result then append the original to the beginning.
(The previous sorted result is shown in blue).</td></tr>
<tr><td></td><td>Sort the previous result then append the original to the beginning.</td></tr>
<tr><td></td><td>etc...</td></tr>
</table>

The original string is MAGIC since that is how BWT works.