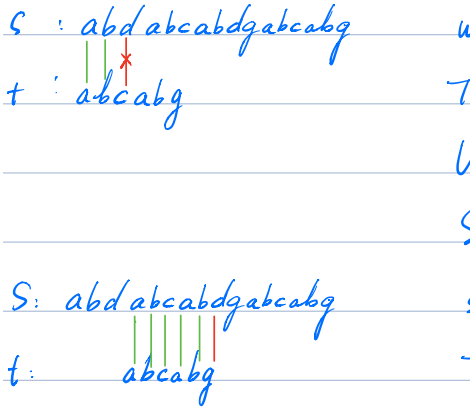
KMP Algo.

The purpose of this algorithm is to efficiently match patterns in string.

WORST CASE PERFORMANCE: O(m+n)

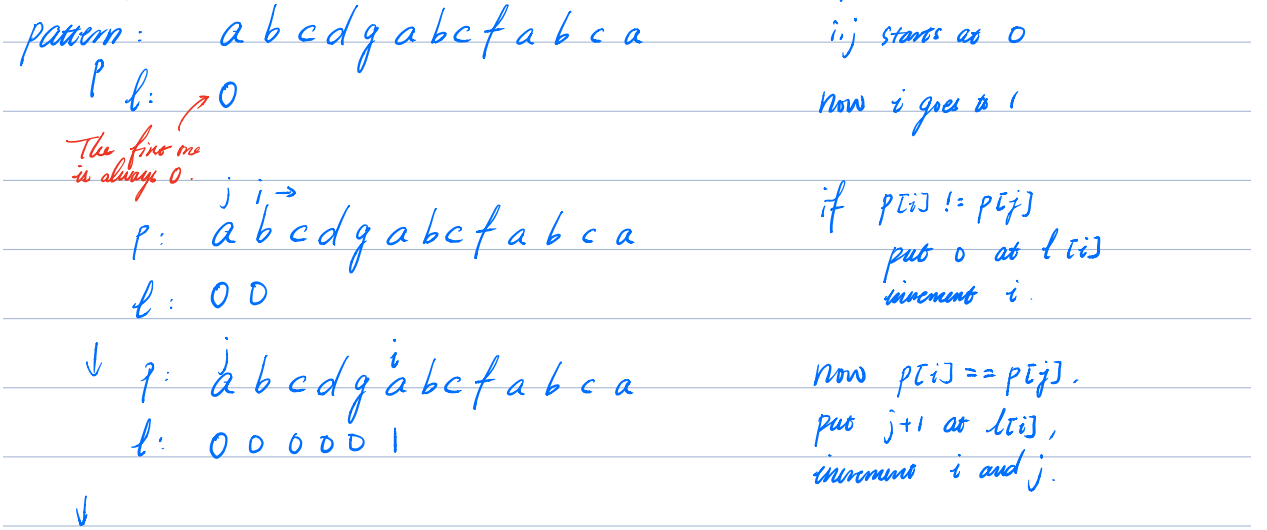
Suppose we have a string "abdabcabdgabcabg" and a pattern "abcabg"

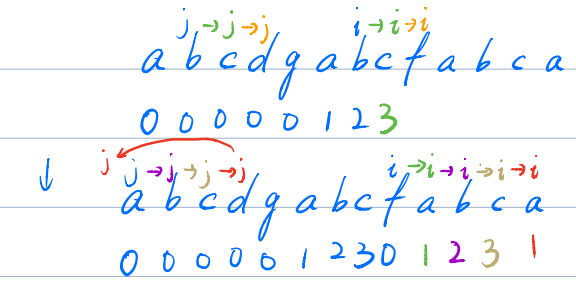
What KMP does is:



THIS IS THE GIST OF KMP ALGO!

To implement this idea, we first preprocess the string to construct a table (list)





ANOTHER MISMATCH!

j will be reset to l[j-1], which is 0.

TIME COMPLEXITY : O(N)

SPACE : O(N)

CAVEATS:

Consider: a a b a a b a a a

0 1 0 1 2 3 4 5 1+1=2

Now p[j] ≠ p[i]. so j needs to be set back to l[j-1] = 2.

However, p[2] ≠ p[i], in this case, we need to further set j to be l[j-1]

Use the table we built:

s: a b x a b c a b c a b y

p: a b c a b y

l: 0 0 0 1 2 0

The value at c is 0, which means in the next round we compare s with p[0]

s: a b x a b c a b c a b y

p: a b c a b y

l: 0 0 0 1 2 0

s: a b x a b c a b c a b y

p: a b c a b y

l: 0 0 0 1 2 0

Value is 2, next comparison would be p[2].

s: a b x a b c a b c a b y

p: a b c a b y

l: 0 0 0 1 2 0

Match!