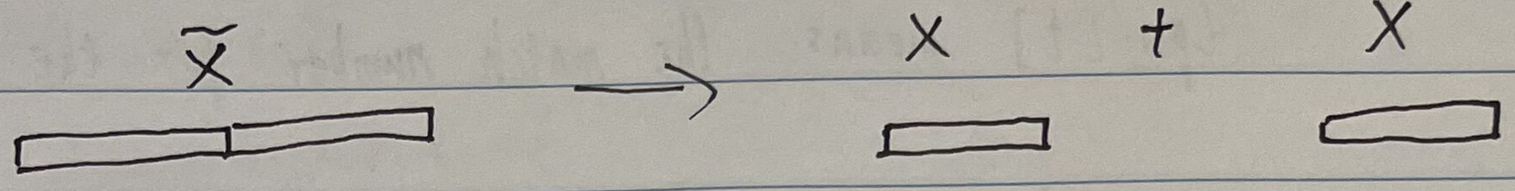
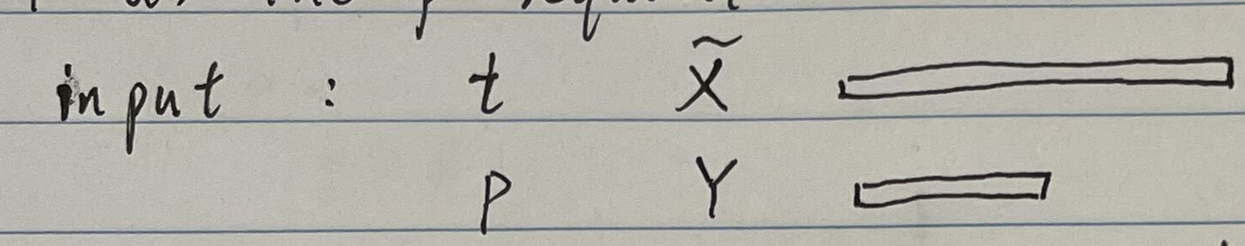
Question 2

First, creat a new $\tilde{X}$ by linking 2 X



Then we use $\tilde{X}$ as the t sequence in Z-algorithm, and use Y as the p sequence.



Finally, use Z-algorithm to find matching pattern.
If the sequence Y is in sequence $\tilde{X}$, it means X and Y come from an identical circular sequence. Otherwise X and Y come from different circular sequences.

Question 3

S = TCATCATGATGATCATCT Length = 18

P = ATCATCT Length = 7

lps = [0,0,0,1,2,3,0]

lps' = [0,0,0,0,0,3,0]

kmp Search:

step 1: TCA……

ATC…

× lps[3] = 0 skip 1-0=1 bytes

step 2: TCA……

ATC…

× lps[2] = 0 skip 1-0=1 byte

step 3: TCATCATGATGA……

ATCATCT

× lps[6] = 3 skip 6-3=3 bytes

step 4: TCATCATGATGA……

ATCATCT…

× lps[3] = 0 skip 3-0=3 bytes

step 5: TCATCATGATGA……

ATCA…

× lps[3] = 0 skip 3-0=3 bytes

step 6: TCATCATGATGATCATCT

ATCATCT match

lps[4] and lps[5] are not used, so there is no difference when using lps' to instead lps.

However, I think the claim is valid.

For example:

S = TCA TCA CAT CAT CT

P = ATC ATCT

lps = [0, 0, 0, 1, 2, 3, 0]

lps' = [0, 0, 0, ~~0~~, 0, 3, 0]

using lps	using lps'
step 3: TCATCACAT CATCT	TCATCACAT CATCT
ATCATCT	ATCATCT
x lps[5] = 2	x lps'[5] = 0
skip 5-2=3 bytes	skip 5-0=5 bytes
step 4: TCATCA CAT CATCT	TCATCA CAT CATCT
ATCATCT	ATCATCT
x lps[2] = 0	match
skip 2-0=2 bytes	
step 5: TCATCACAT CATCT	no need 1 more step
ATCAT CT	
match	

At this situation, using lps will increase 1 step compared to using lps', which means lps' is more efficient than lps.

In my opinion, if matching stops at position i and lps[i] is not '0', the next step will not get right match. So if we want to accelerate the process, we need to reduce none-zero value in lps. And in lps, later none-zero value will cover the front position, which means the claim in the Question 3 will not result in missing right match.

Question 4

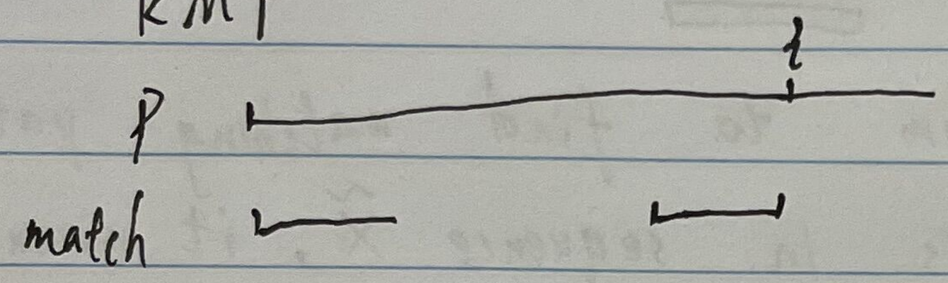
KMP Algorithm:

lps[i] means the match number for the string ending with i

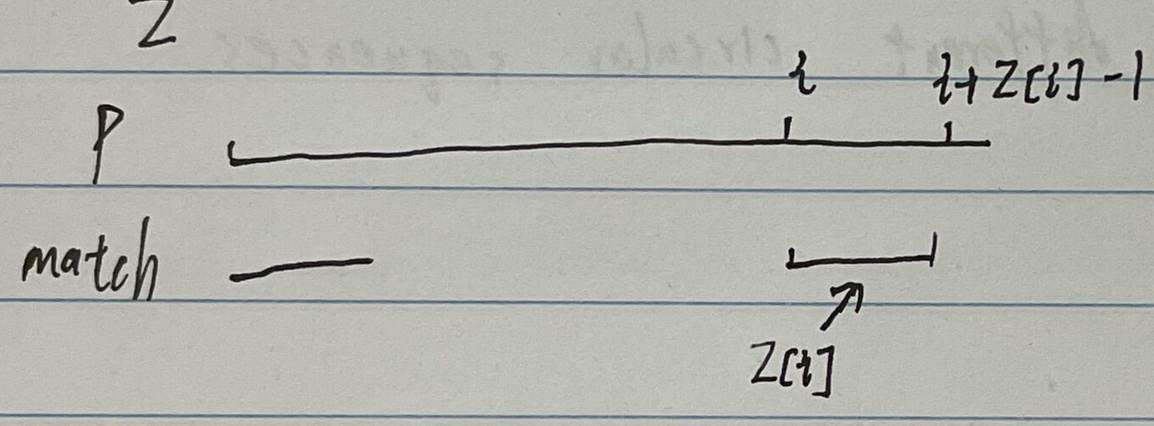
Z Algorithm:

z[i] means the match number for the string beginning with i

KMP



Z



Z[i] means the length of matching string

So lps[i+Z[i]-1] = Z[i]

pseudo code: input: string S, Zarray

output: lps'

1) lps' = 0, 0, ... 0 (length of S)

2) for i in range(1, length(S)):

    if Z[i] > 0:

        lps[i+Z[i]-1] = Z[i]