String

xmp

N=23

```
          1         2
m: 01234567890123456789012
S: ABC ABCDAB ABCDABCDABDE
W: ABCDABD
i: 0123456
```

i=1, j=0,

pat = "ABCDABD" M=7

	2	0	1	2	3	4	5	6
lps[]	0	0	0	0	1	2	0	

should be 0

① while (i<M) {
if p[i] == p[j] x
B A
j=0 lps[i]=j=0 ✓
i++ = 2

② while(i<M){
if p[i] == p[j]
C A x
j=0. lps[i]=j=0. ✓
i++ = 3

③ while (i<M) {
if p[i] == p[j] x
D A
j=0 lps[i]=j=0 ✓
i++ = 4

④ while(i<M){
if p[i] == p[j]
A A
j++ = 1
lps[4]=1, i++=5

⑤ while(i<M){
if p[i] == p[j]
B B
j++ = 2
lps[i]=2. ✓
i++ = 6.

⑥ while(i<M){
if p[i] == p[j]
D C x
j=0.
j=lps[j-1]=0

⑦ while(i<M)
if p[i] == p[j]
D A x
j=0. lps[6]=j=0.
i++ = 7. end

Partial Match
(Failure Function)

i	0	1	2	3	4	5	6
lps[i]	0	0	0	0	1	2	0

```
          1         2
m: 01234567890123456789012
S: ABC ABCDAB ABCDABCDABDE
W: ABCDABD
i: 0123456
```

k 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

t[k] A B C A B C D A B A B C D A B C D A B D E

★ pat = "ABCDABD" M=7 N=23

	0	1	2	3	4	5	6
lps[]	0	0	0	0	1	2	0

i=0, j=0,

① while((N-i) >= (M-j)) {
if p[j] == T[i] (A A) j++=1, i++=1
if j==M x
else i<N p[j] != T[i] (B B) x

② while((N-i) >= (M-j)) {
if p[j] == T[i] (B B) j++=2, i++=2
if j==M x
if i<N. p[j] != T[i] (C C) x

③ while((N-i) >= (M-j)) {
if p[j] == T[i] (C C) j++=3, i++=3
if j==M x
else i<N. p[j] != T[i] (D D)
if j!=0. j=lps[j-1]=0 ✓

④ while((N-i) >= (M-j)) {
if p[j] == T[i] (A D) x
if 0==M x
else i<N. p[j] != T[i] (A D)
j==0, i++=4.

⑤ while((N-i) >= (M-j)) {
if p[j] == T[i] (A A) i++=5, j++=1
if j==M x
else i<N p[j] != T[i] (B B) x

⑥ while((N-i) >= (M-j)) {
if p[j] == T[i] (B B) i++=6, j++=2
if j==M x
else i<N. p[j] != T[i] (C C)

⑦ while((N-i) >= (M-j)) {
if p[j] == T[i] (C C) i++=7, j++=3
if j==M x
else i<N. p[j] != T[i] (D D)

⑧ while((N-i) >= (M-j)) {
if p[j] == T[i] (D D) i++=8, j++=4.
if j==M x
else i<N. p[j] != T[i] (A A)

```
          1         2
m: 012345678901234567890123
S: ABC ABCDAB ABCDABCDABDE
W: ABCDABD
i: 0123456
```

lps[] 0 0 0 0 1 2 0

k 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

tx A B C A B C D A B A B C D A B C D A B D E

pat = "ABCDABD" M=7 N=23

(9) while (N-i >= M-j) {
if p[4] == T[3] (A A) i++=10, j++=5
if j==M x
else i<N, p[5] (B B) = T[6]

(10) while (N-i >= M-5) {
if p[5] == T[6] (B B) i++=10, j++=6
if j==M x
if i<N, p[6] (D) = p[10] (O)
j!=0 j = lps[6-1] = 2

(11) while (N-10 >= M-2) {
if p[2] == T[10] (C O) x
if j==M x
if i<N, p[j] (C) = T[10] (O)
j!=0 j = lps[2-1] = 0

(12) while (N-10 >= M-0) {
if p[0] == T[10] (A O) x
if j==M x
else i<N p[0] (A) = T[10] (O)
j=0, i++=11.

(13) while (N-i >= M-0) {
if p[0] == T[11] (A A) i++=12, j++=1
if j==M x
else i<N, p[j] (B) = T[12] (B)

(14) while (N-12 >= N-1) {
if p[1] == T[12] (B B) i++=13, j++=2
if j==M x
else i<M p[2] (C) = T[13] (C)

(15) while (N-13 >= N-2) {
if p[2] == T[13] (C C) i++=14, j++=3
if j==M x
else if i<N, p[3] (D) = T[14] (D)

(16) while (N-14 >= N-3) {
if p[3] == T[14] (D D) i++=15, j++=4
if j==M x
else i<N, p[4] (A) = T[15] (A)

```
        1         2
m: 012345678901234567890123
S: ABC ABCDAB ABCDABCDABDE
W: ABCDABD
i: 0123456
```

☆

	0	1	2	3	4	5	6
lps[]	0	0	0	1	2	0	

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
t[k]	A	B	C		A	B	C	D	A	B		A	B	C	D	A	B	C	D	A	B	D	E

match.

☆ pat = "ABCDABD" M=7 N=23

(17) while(N-i >= M-j) {
if p[j] == T[i] (j++, i++)
A A
if j == M x
else i<N, p[j] != T[i]
B B

(18) while(N-i >= M-j) {
if p[j] == T[i] (j++, i++)
B B
if j == M x
else p[j] != T[i]
D C
j = 0 j = lps[j-1] = 2

(19) while (N-i >= M-2) {
if p[j] == T[i]
C C (i++, j++)
if j == M x
else i<N, p[j] != T[i]
D D

(20) while (N-i >= M-3) {
if p[j] == T[i] (i++, j++)
D D
if j == M x
if else i<N p[j] != T[i]
A A

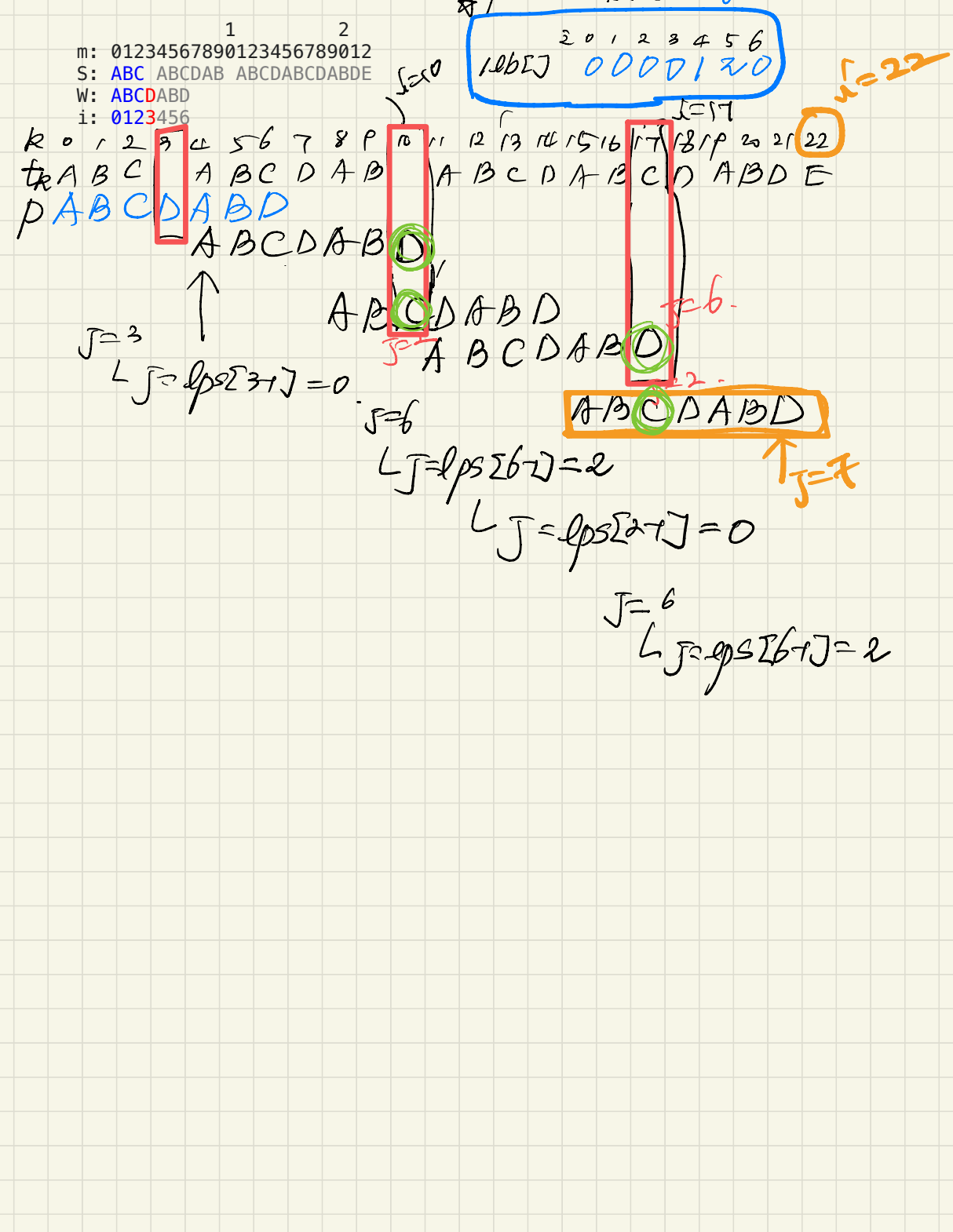
(20) while(N-i >= M-4) {
if p[j] == T[i] (i++, j++)
A A
if j == M x
else i<N, p[j] != T[i]
B B

(22) while(N-i >= M-5) {
if p[j] == T[i] (i++, j++)
B B
if j == M x
else i<N, p[j] != T[i]
D D

(23) while (N-i >= M-6) {
if p[j] == T[i] (j++, i++)
D D
if j == M

찾음 Found at (i-j)
= (22-7) = 15

☆☆☆


1 2
m: 01234567890123456789012
S: ABC ABCDAB ABCDABCDABDE
W: ABCDABD
i: 0123456
lps[j] 0 0 0 0 1 2 0
k 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
tk A B C A B C D A B A B C D A B C D A B D E
P A B C D A B D
A B C D A B D
A B C D A B D
A B C D A B D
A B C D A B D
j=0
j=17
j=22
j=3
j=7
j=6
j=2
j=7
j=6
L j=lps[3-1]=0
L j=lps[7-1]=2
L j=lps[2-1]=0
L j=lps[6-1]=2

```
void computeLPS(String pat, int M, int lps[]) {
    int j=0, i=1, lps[0]=0;
    while(i<M) {
        if(pat.charAt(i)==pat.charAt(j)) {
                lps[i++]=++j;
        } else {
            if(j!=0) {
                j=lps[j-1];
            } else {
                lps[i++]=j;
            }
        }
    }
}
```

```
void KMP(String pat, String txt){
    int M = pat.length();
    int N = txt.length();
    int lps[] = new lps[M];
    int j = 0, i = 0;
    computeLPS(pat, M, lps);
    while ((N-i) >= (M-j)){
        if (pat.charAt(j) == txt.charAt(i)){
            i++, j++;
        }
        if (j == M){
            System.out.println("Found pattern " + (i-j));
            return;
        }
        else if (i < N && pat.charAt(j) != txt.charAt(i)){
            if (j != 0)
                j = lps[j-1];
            else
                i++;
        }
    }
}
```