DP 5

Content

01. Longest common subsequence
02. Longest Palindromic subsequence
03. Edit distance Google
04. Wild card matching Leetcode Hard

Q Given 2 strings, find the length of longest common subsequence.

S_1 = abc → { a, b, c, ab, ac, abc, bc }

S_2 = ace → { a, c, e, ac, ae, ace, ce }

Ans = 2

S_1 = abbcdgh

S_2 = bachegf

{ ach, acg, bcg } Ans = 3

S_1 = abbcdgf

S_2 = achegf

{ acgf } Ans = 4

S_1 = abbcdgf (0 1 2 3 4 5 6)

S_2 = achegf (0 1 2 3 4 5)

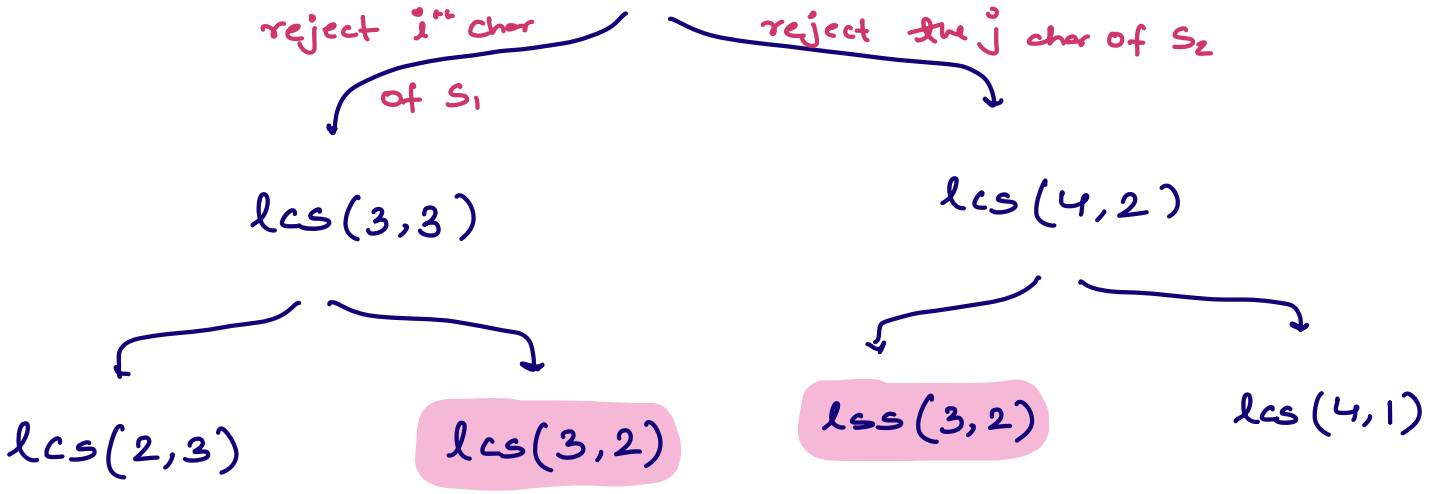
lcs(6, 5)

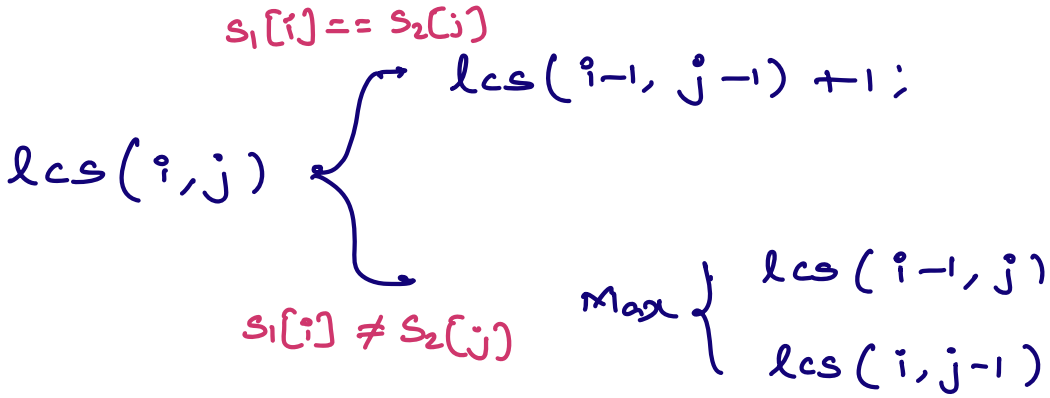
↓ if ($S_1[6] == S_2[5]$)

lcs(5,4) + 1

↓ $S_1[5] == S_2[4]$

lcs(4,3) + 1






```
int [][] dp = new int[n][m]
```

(→)

```
int lcs (string s1, string s2, i, j)        // i: n-1, j: m-1
{
    if (i<0 || j<0) return 0;
    if (dp[i][j] != -1) return dp[i][j];
      ans = 0
    if (s1[i] == s2[j]){
        ans = lcs(s1, s2, i-1, j-1) + 1
    }
    else {
        int a = lcs(s1, s2, i-1, j);
```

TC: O(n*m)

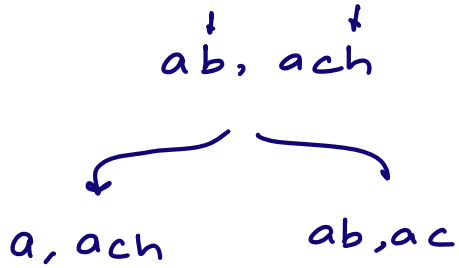
SC: O(n*m)

```
        int b= lcs(s1, s2, i, j-1);

        ans= Max(a,b)
    }
    db[i][j]= ans;
    return ans;
}
```

db[i][j] → max length of common subsequence in (0-i) of s_1 & (0-j) of s_2

		a	c	h	e	g	f
		0	1	2	3	4	5
a	0	1	1	1	1	1	1
b	1	1	1	1	1	1	1
b	2	1	1	1	1	1	1
c	3	1	2	2	2	2	2
d	4	1	2	2	2	2	2
g	5	1	2	2	2	3	3
f	6	1	2	2	2	3	4



```
for (i=0; i<n ; i++)
    for (j=0; j<m; j++){
        if ( s1[0] == s2[0]){
            dp[0][0] = 1;
        }
        else if ( j==0 ){
            if (s1[i] == s2[j] ) dp[i][j] = 1
            else  dp[i][j] = dp[i-1][j];
        else if (i==0)
            if (s1[i] == s2[j] ) dp[i][j] = 1
            else  dp[i][j] = dp[i][j-1];
        }
        else {
            if (s1[i] == s2[j])
                dp[i][j] == dp[i-1][j-1] + 1;
            else{
                dp[i][j] = max ( dp[i-1][j], dp[i][j-1]);
            }
        }
    }
}
return dp[n-1][m-1];
```

Q Given a string, find **longest palindromic subsequence**

A = "scalar" Ans = 3

A = "abcdefb" Ans = 3

A = "abdcgba" Ans = 5

palindrome of A = palindrome of reverse of A

S_1 = scalar
S_2 = ralacs
} find LCS → 3

A = abcdefb
B = bfedcba
} Ans = 3

A = abdcgba
B = abgcdba
} Ans = 5

10:02 → 10:12 pm

Q Given 2 strings S_1 & S_2, find min operations to be performed in S_1 so that it becomes equal to S_2.

Operations allowed

01. Insert → we can insert any character
02. Replace → we can replace any character
03. Delete → we can delete any character

Eg:- $S_1 = d\ f\ a\ c\ l$ (g)
$S_2 = f\ g\ l$

→ delete d
→ delete a
→ replac c

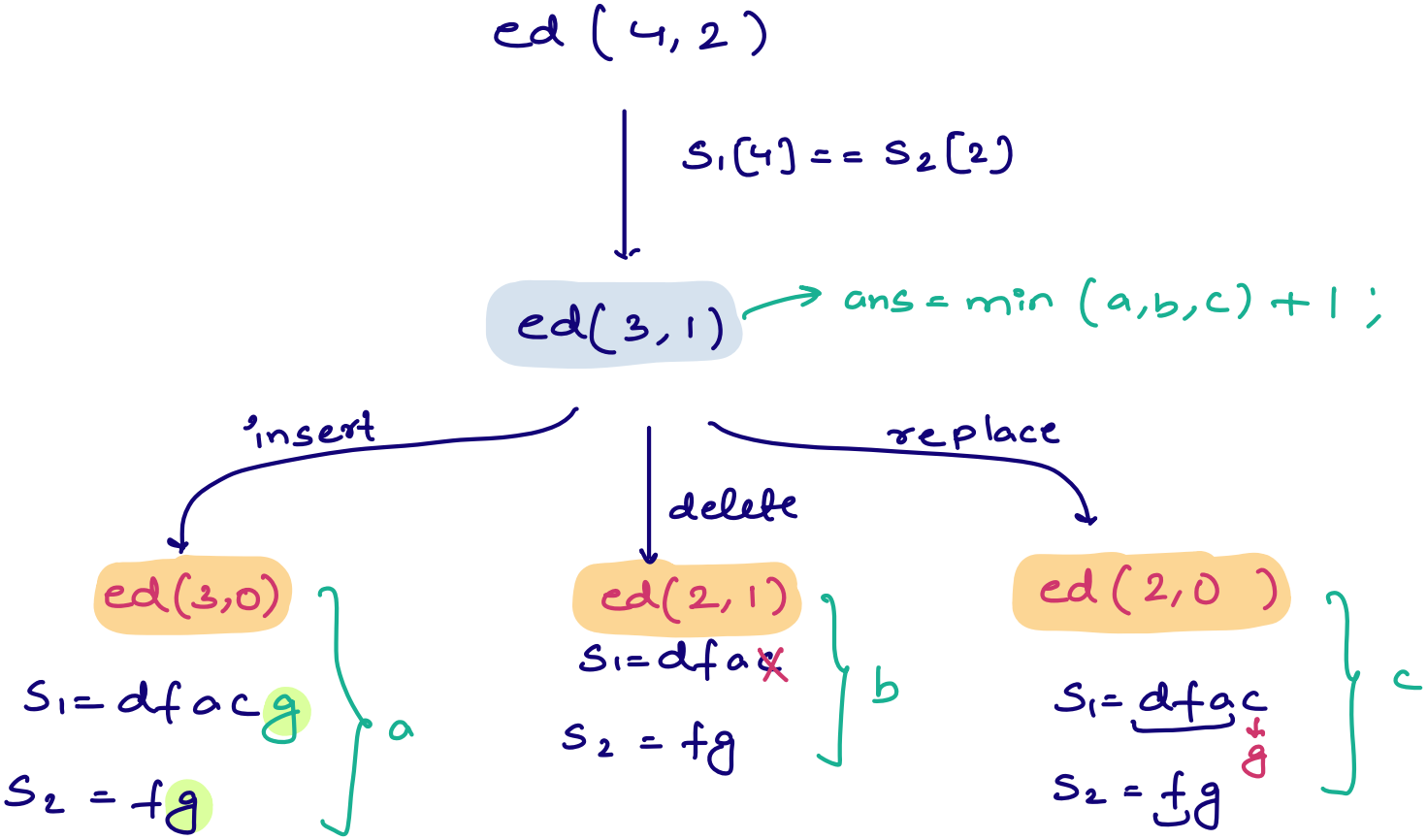
} Ans = 3

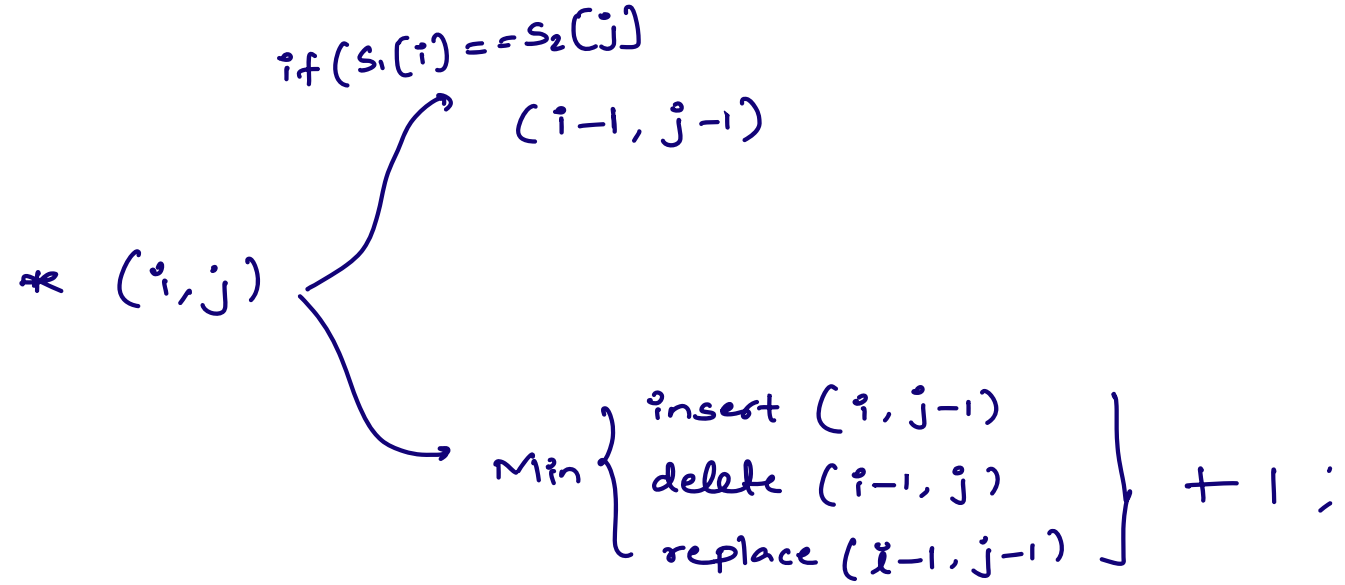
Eg:- $S_1 = d\ f\ a\ x\ z$
$S_2 = f\ x\ z$

} Ans = 4

$S_1 = d\ f\ a\ c\ l$ (indices 0 1 2 3 4)

$S_2 = f\ g\ l$ (indices 0 1 2)





int [][] dp = new int [n] [m] ⟶ -1

```
int ed(string s1, s2, i, j)
{
    if (i<0 && j<0) return 0;
    if (i<0) return j+1      // j+1 no. of char to insert
    if (j<0) return i+1      // i+1 char to be deleted
    if (dp[i][j] != -1) return dp[i][j];
    int ans=0
    if (s1[i]==s2[j])
    {
        ans = ed(s1, s2, i-1, j-1);
    }
    else {
        int a = ed(s1, s2, i, j-1);   // insert
        int b = ed(s1, s2, i-1, j);   // delete
        int c= ed(s1, s2, i-1, j-1);  // replace
        ans = min(a,b,c)+1;
    }
    dp[i][j] = ans;
    return ans;
}
```

TC : O(n*m)

SC : O(n*m)

* Regular Expression Matching

Given an input string s & pattern p, implement pattern matching with support for ? & *

? → matches any single character

* → matches any sequence of character

s = "aa"
p = "a"

Ans = false

s = "aa"
p = "*"

Ans = true

s = "cb"
p = "?a"

Ans = false

s = "ab"
p = "a*b"

Ans = true

s = a
p = a*b

Ans = false

s = a x y z p q b
p = a * b
} Ans = true

s = a b c
p = a ? *
} Ans = true

Approach

Case 1

```
      0 1 2 3 4 5
s1 =  a b c d e f  →

      a * d ? f  →
p =   0 1 2 3 4
```

isMatch(5,5) → ismatch(4,3)

Case 2

```
      0 1 2 3 4
s =   a b c d e
      a * d ?
p =   0 1 2 3
```

ismatch(4,3) → ismatch(3,2)

Case 3

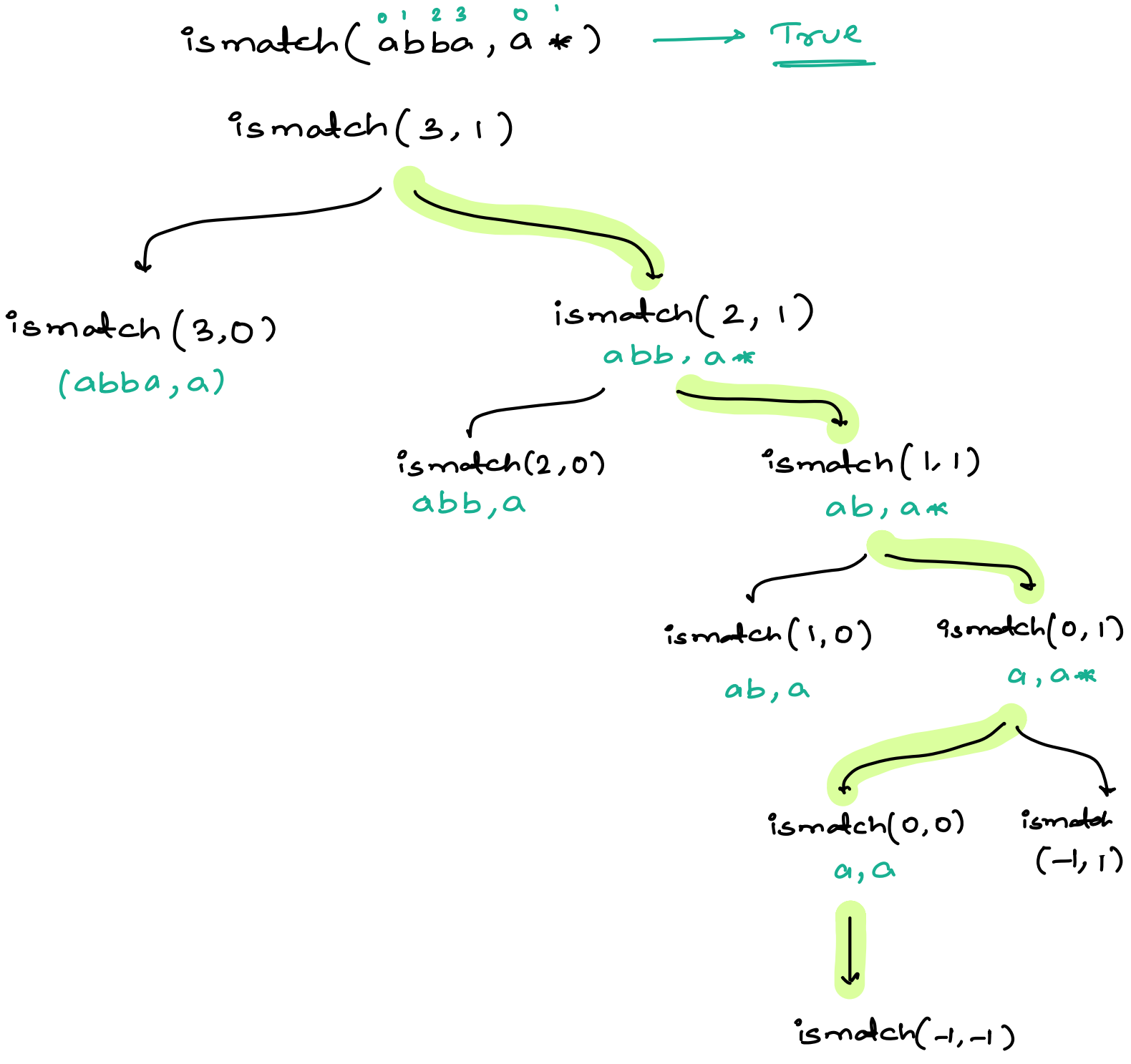
```
      0 1 2 3
s =   a b b a

      a *
p =   0 1
```

* → it can match with 0 characters

* → can match with one or more character

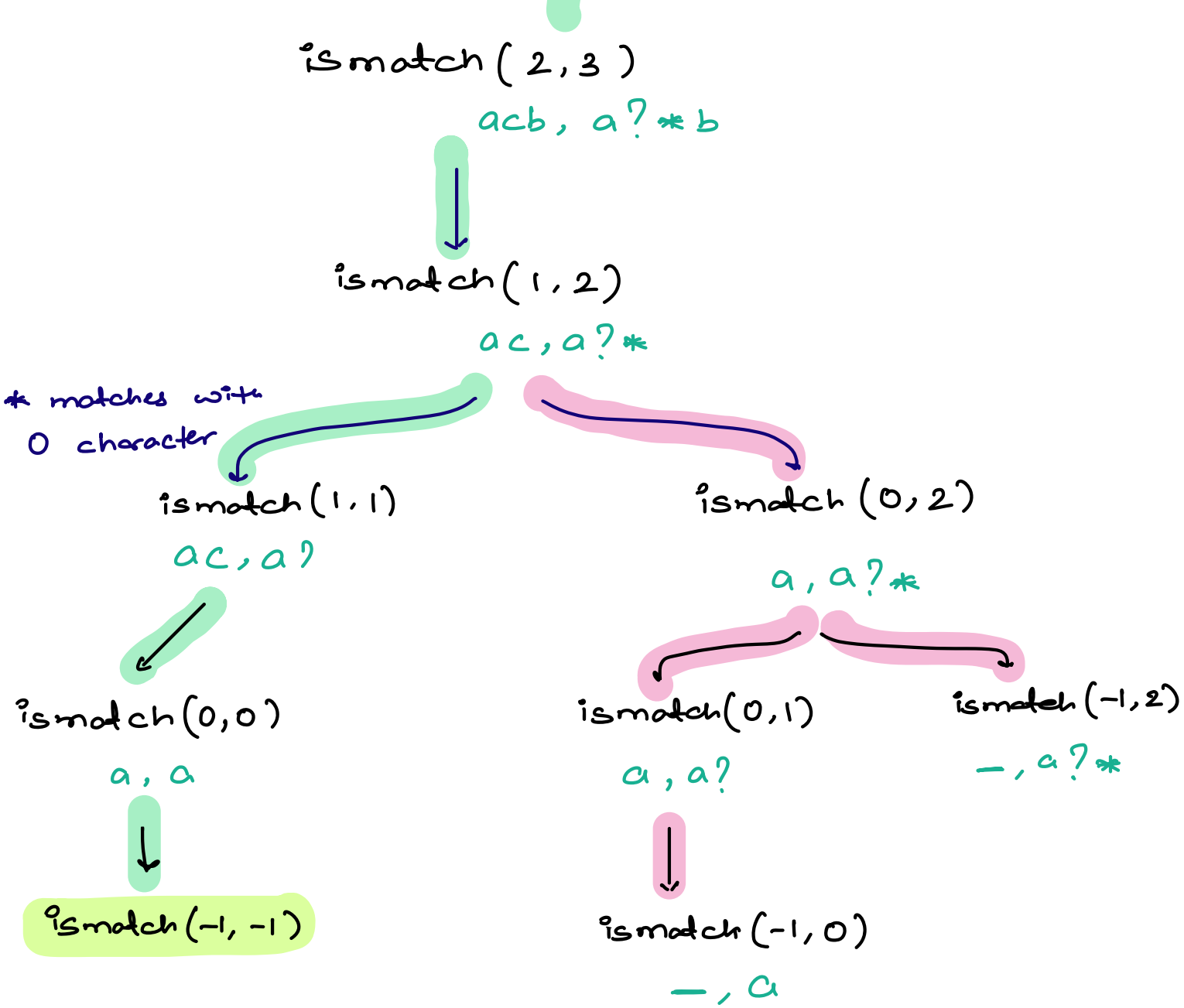
ismatch(abba , a*) → True



S = a c b (0 1 2)

P = a ? * b (0 1 2 3)

True



* Case 4

S = _

P = *

S = abc

P = a*?d

} Ans = false

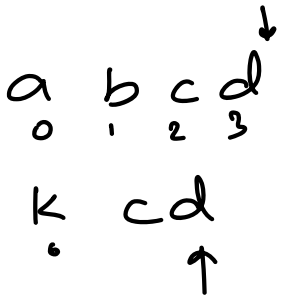
```
boolean ismatch(String s, String p, int i, int j)
{
    if (i<0 && j<0) return true;
    else if (j<0) return false

    if (i<0){
        for(k=0; k<=j; k++){
            if (p[k]!='*') return false
        }
        return true;
    }
    if (s[i]==p[j])
        return ismatch(s, p, i-1, j-1);

    else if(p[j]=='?')
        return ismatch(s, p, i-1, j-1);

    else if (p[j]=='*'){
        return ismatch(s, p, i, j-1) ||
               ismatch(s, p, i-1, j);
    }
    else{
        return false;
    }
}
```

Doubts



lcs(l , m)

Max

```
for ( i=l ; i<n ; i++ )
    for (j=m ; j<m ; j++) {
        if ( match(i,j) {
            m= lcs (i+1, j+1) +1
        }
        else{
        }
    }
```