Count A+b+c+ Subsequences

Str : $a_1 b_1 c_1 d a_2 b_2 c_2$

$a_1 b_1 c_1$

$a_1 b_2 c_2$

$a_2 b_2 c_2$

$a_1 a_2 b_2 c_2$

$a_1 b_1 b_2 c_2$

$a_1 b_1 c_2$

$a_1 b_1 c_1 c_2$

a + b + c +

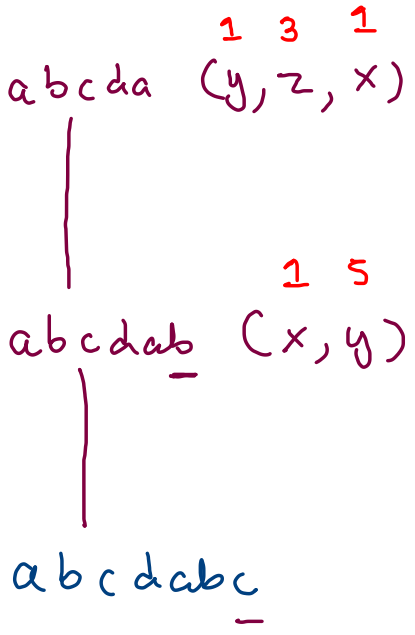
one or more 'a'

one or more 'b'

one or more 'c'

Str : a b d c c

- a b c
- a b c
- a b c c



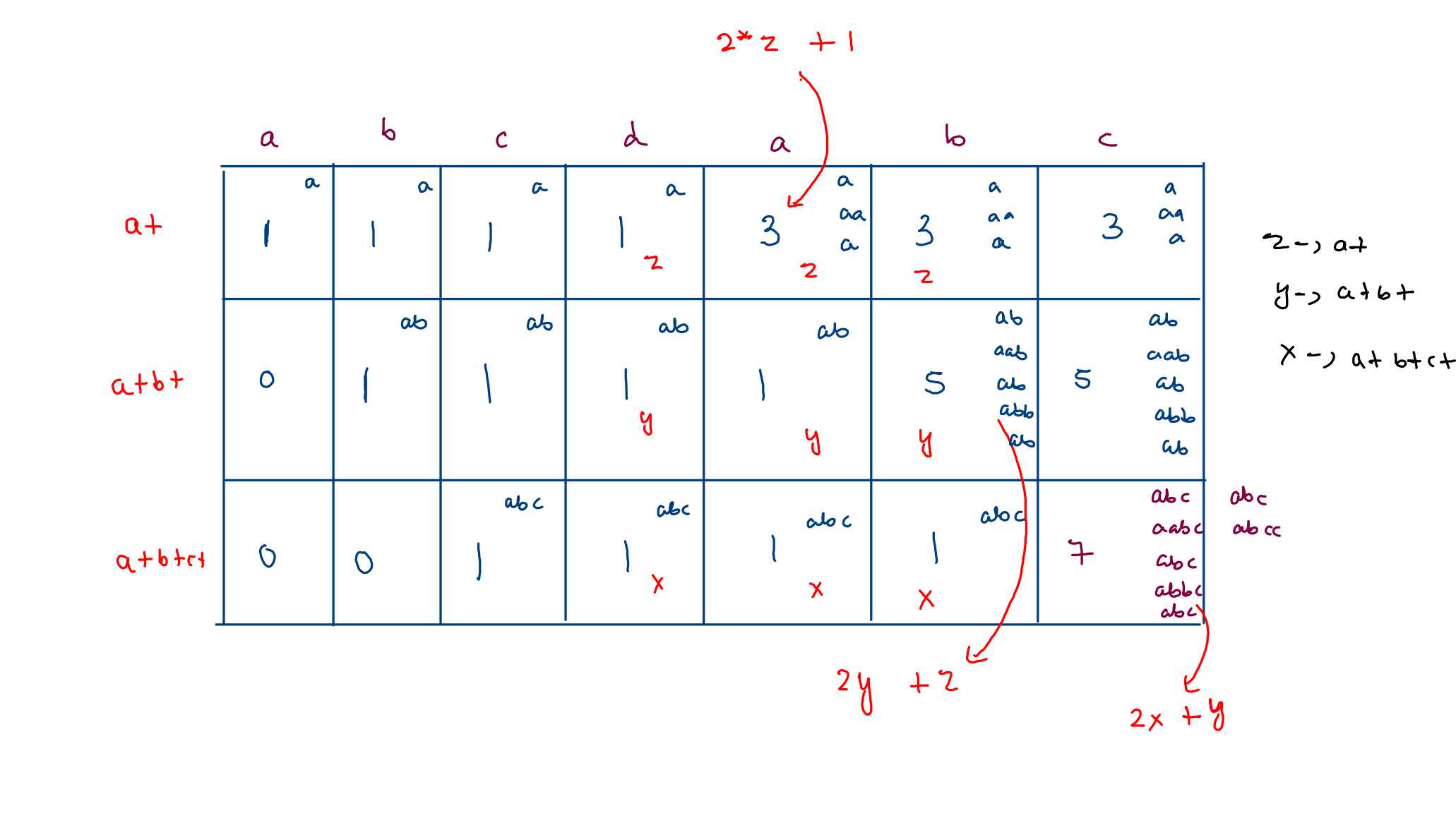
x
↓
a+b+c+

y
↓
a+b+

z
↓
a+

2x + y

2 + 5 = 7


2*z +1
a
b
c
d
a
b
c
a+
a+b+
a+b+c+
z -> a+
y -> a+b+
x -> a+b+c+
2y +z
2x +y

```java
public static int count(String str) {
    int a = 0; //count of a+
    int ab = 0; //count of a+b+
    int abc = 0; //count of a+b+c+

    for(int i=0; i < str.length();i++) {
        char ch = str.charAt(i);

        if(ch == 'a') {
            a = 2*a + 1;
        }
        else if(ch == 'b') {
            ab = 2*ab + a;
        }
        else if(ch == 'c') {
            abc = 2*abc + ab;
        }
    }

    return abc;
}
```

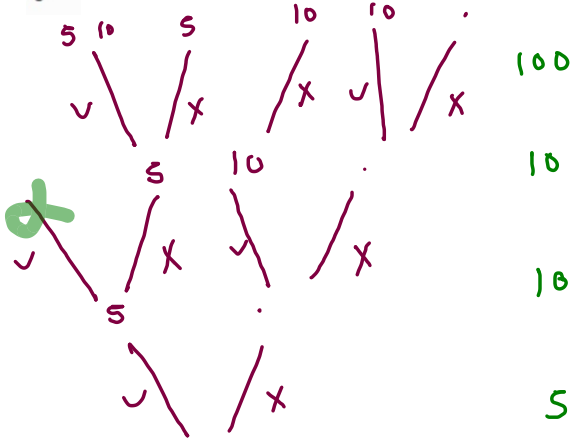
str: a b d c ab c

	a	b	d	c	a	b	c
a+ (a)	1 a	1 a	1 a	1 a	3 aa a a	3 aa a a	3 aa a a
a+b+ (ab)	0	1 ab	1 ab	1 ab	1 ab	5 abb ab aab ab ab	5 abb ab aab ab ab
a+b+c+ (abc)	0	0	0	1 abc	1 abc	1 abc	7 abcc abc abbc abc

aab c
abc
abc

Maximum Sum Non Adjacent Elements

6
5
10
10
100
5
6



-> 5 10 5

-> 5 100 6

-> 10 100 6 ⟶ 116

arr : 5 10 10 100 5 6

⇓

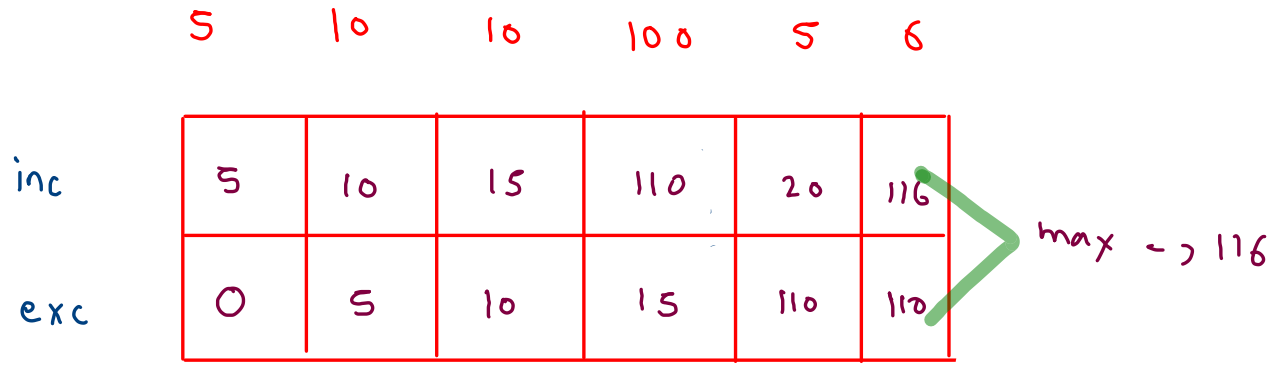
all subseq

⇓

non-adjacent elements subseq

⇓

max sum



valid

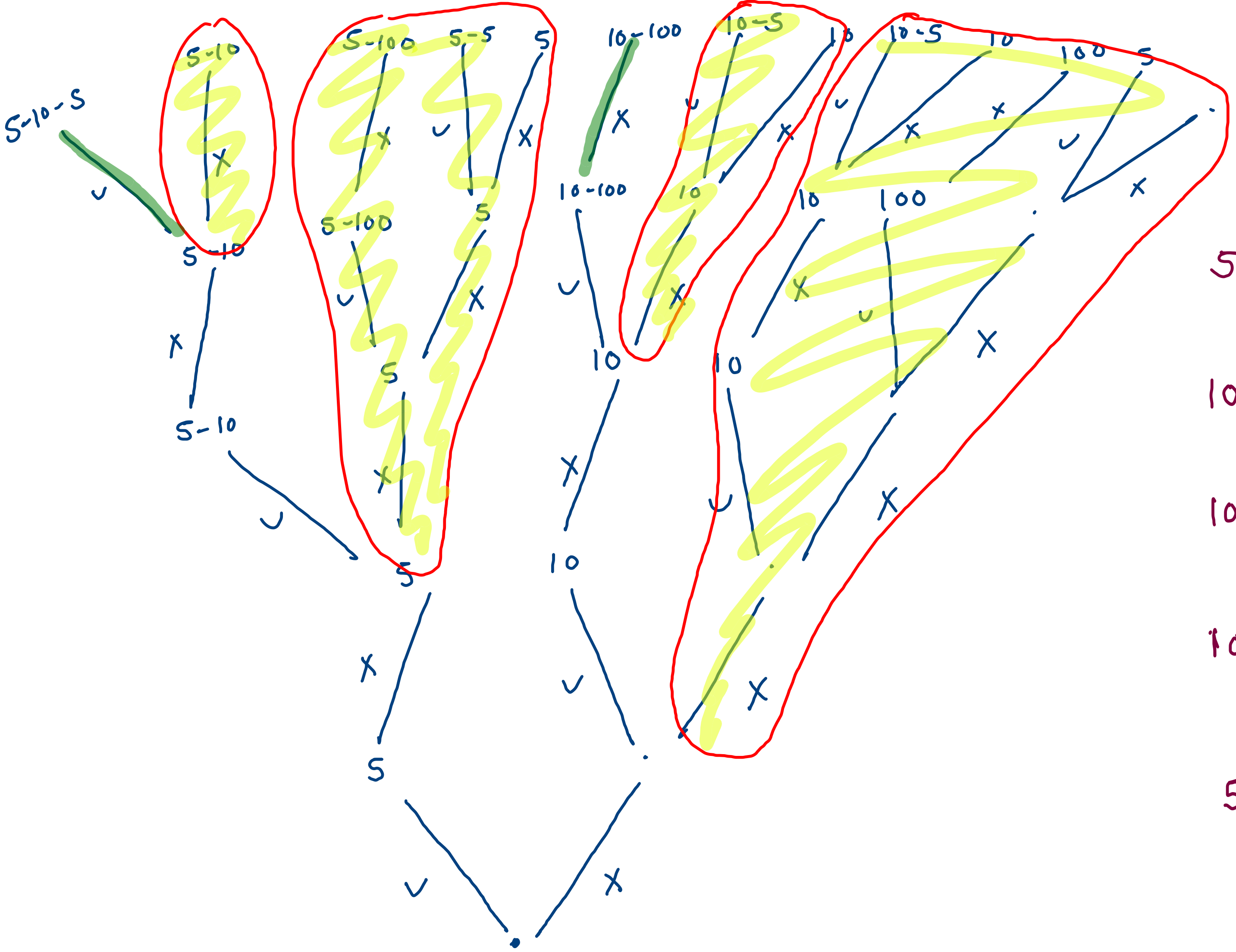
↓

non-adjacent

elements subseq.

inc → oexc + arr[i]

exc → Math.max(oexc, oinc)



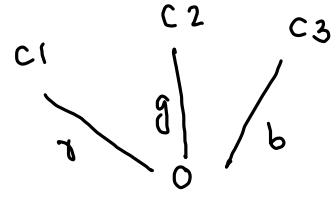
	inc	exc
5	20	110
100	110	15
10	15	10
10	10	5
5	5	0

Paint House

```
4
1 5 7
5 8 4
3 2 9
1 2 4
```

	r	g	b
0	1	5	7
1	5	8	4
2	3	2	9
3	1	2	4

min cost : no two adjacent houses are painted with same colors.



	r	g	b
0	1	5	7
1	5	8	4
2	3	2	9
3	1	2	4

	r	g	b
0	1	5	7
1	10	9	5
2	8	7	18
3	8	10	11

ans: 8

dp[i][j]:

valid ways [min cost to paint houses [0 to i] such that ith house is painted with j^{th} color

	r 0	g 1	b 2
0	6	5	7
1	5	8	9
2	3	10	15
3	1	2	4

	r	g	b
0	6	5	7
1	10	14	14
2	17	20	25
3	0r 21	0g 19	0b 21

i

19

```java
public static int minCost(int[][]cost) {
    int n = cost.length;

    int or = 0;
    int og = 0;
    int ob = 0;

    for(int i=0; i < n;i++) {

        int nr = Math.min(og,ob) + cost[i][0];
        int ng = Math.min(or,ob) + cost[i][1];
        int nb = Math.min(or,og) + cost[i][2];

        or = nr;
        og = ng;
        ob = nb;
    }

    return Math.min(Math.min(or,og),ob);
}
```

Paint House - Many Colors

	r	g	b	y
0	5	9	3	8
1	6	10	4	7
2	1	8	3	9
3	6	4	15	12
4	10	9	7	4

	r	g	b	y
0	5	9	3	8
1	9	13	9	10
2	10	17	12	18
3	18	14	25	22
4	24	27	21	18

$n * m * m$

nm^2

nm

(min, second_min)

5 9 6 3 8 4

min = 3 smin = 4

min = ∞ smin = ∞

```
for (int i = 0; i < n; i++) {
    if (arr[i] < min) {
        smin = min;
        min = arr[i];
    }
    else if (arr[i] < smin) {
        smin = arr[i];
    }
}
```

	r	g	b	y
0	5	9	3	8
1	6	10	4	7
2	1	8	3	9
3	6	4	15	12
4	10	9	7	4

	r	g	b	y
0	5	9	3	8
1	9	13	9	10
2	10	17	12	18
3	18	14	25	22
4	24	27	21	18

omin = 18 osmin = 21

return omin.

	r	g	b	y
	0	1	2	3
0	10	8	5	9
1	6	7	8	12
2	13	4	9	14

cost

```
for(int i=0; i < dp.length;i++) {

    int cmin = Integer.MAX_VALUE; //current min
    int csmin = Integer.MAX_VALUE; //current second min

    for(int j=0; j < dp[0].length;j++) {
        if(i == 0) {
            dp[i][j] = cost[i][j];
        }
        else {
            //try to use omin
            if(dp[i-1][j] != omin) {
                dp[i][j] = cost[i][j] + omin;
            }
            else {
                dp[i][j] = cost[i][j] + osmin;
            }
        }


        //maintain cmin and csmin
        if(dp[i][j] < cmin) {
            csmin = cmin;
            cmin = dp[i][j];
        }
        else if(dp[i][j] < csmin) {
            csmin = dp[i][j];
        }
    }


    omin = cmin;
    osmin = csmin;
}
```

	r	g	b	y
	0	1	2	3
0	10	8	5	9
1	11	12	16	17
2	25	15	20	25

omin = 15 osmin = 20

i

dp

Paint Fence

n = 4

k = 3

r, g, b

valid

be painted so that not more than two consecutive fences have same colors.

	1	2	3	4
last 2 fences same color	X	3 (rr, gg, bb)	6 (rgg, rbb, bgg, brr, grr, gbb)	18
last 2 fence diff color	X	6 (rg, rb, bg, br, gr, gb)	18	48

→ 66