

q	-	1
b	-	2
c	-	3
d	-	4
e	-	5
f	-	6
g	-	7
h	-	8
i	-	9
j	-	10
k	-	11
l	-	12
m	-	13
n	-	14
o	-	15
p	-	16
q	-	17
r	-	18
s	-	19
t	-	20
u	-	21
v	-	22
w	-	23
x	-	24
y	-	25
z	-	26

abc

abc/""

ab/3

au/""

lc/""

single

pair

l 'a' / (23)

l/3

123

single

pair

123

123/d

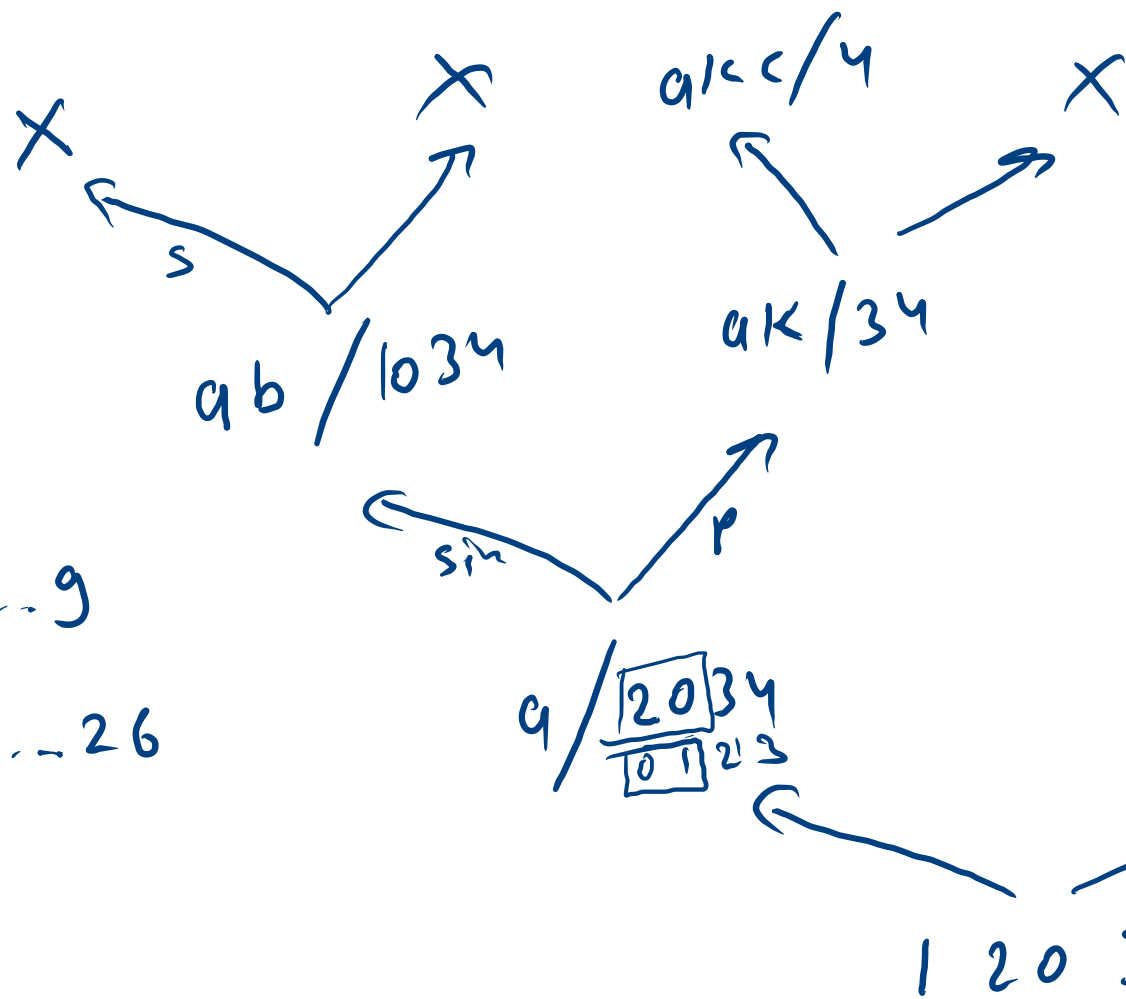
a/234

l/34

single

1234

single 1...9  
 pair 10...26



```
int n = charAt(0) - '0'
String s = str.substring(0, 2);
int n2 = Integer.parseInt(s);
```

l / 0 3 4

1	→ 'a'	(0 + 'a')
2	→ 'b'	(1 + 'a')
3	→ 'c'	(2 + 'a')
4	→ 'd'	(3 + 'a')

val  
 char ch = (char)(val - 1 + 'a')

```
int x = Integer.parseInt(str.charAt(0)+""); 2
if(x>=1 && x<= 9){
    char ch = (char)(x-1+'a'); 'b'
    printEncodings(str.substring(1), asf+ch);
}
```

```
// pair
if(str.length() >= 2){
    String s = str.substring(0,2); "12"
    int val = Integer.parseInt(s); 12
    if(val >= 10 && val <= 26){ 0
        char ch = (char)(val-1+'a');
        printEncodings(str.substring(2), asf+ch);
    }
}
```

ake / ""

```
if(str.length() == 0){
    System.out.println(asf);
    return;
}
```

lc / ""

ab / 3

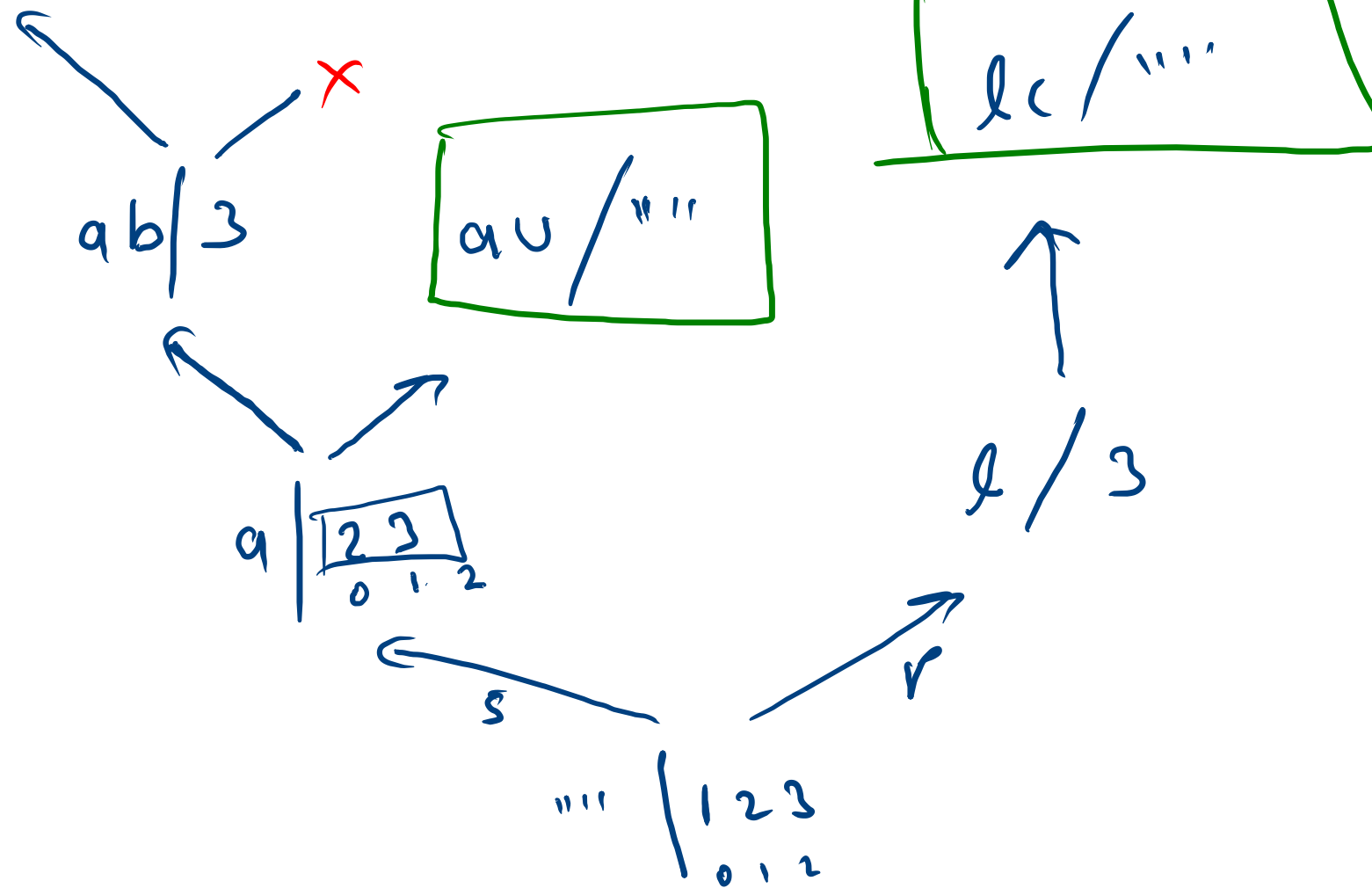
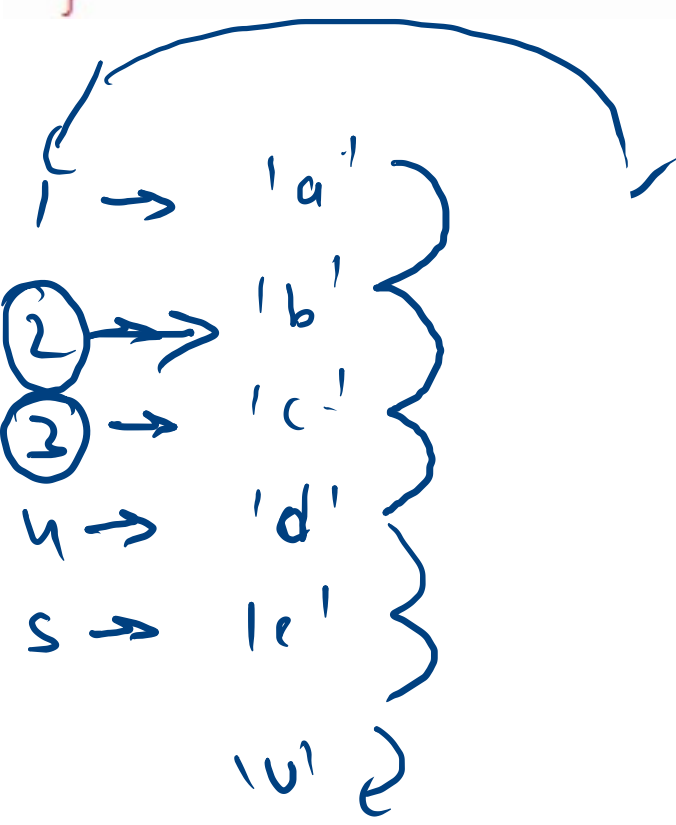
au / ""

l / 3

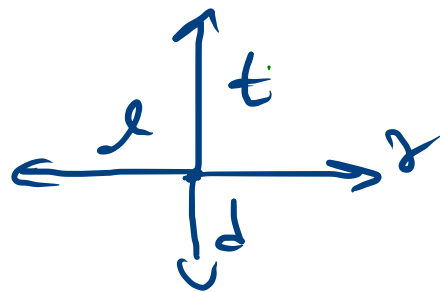
a | 23  
0 1 2

"" | 123  
0 1 2

12 | 3  
01 | 2



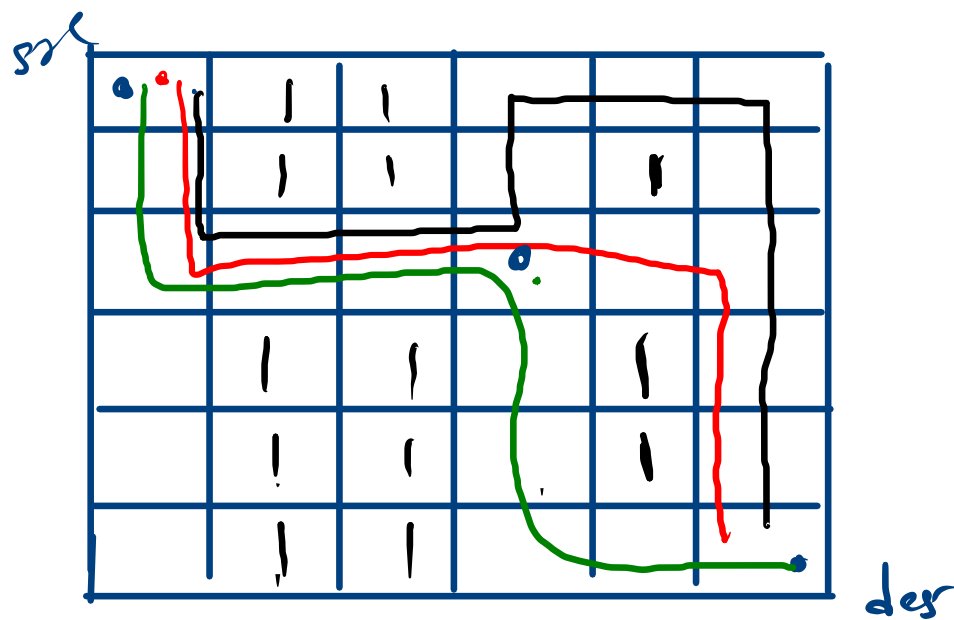
t l d r



0/1

black ✓  
green ✓

t  
l  
d  
r



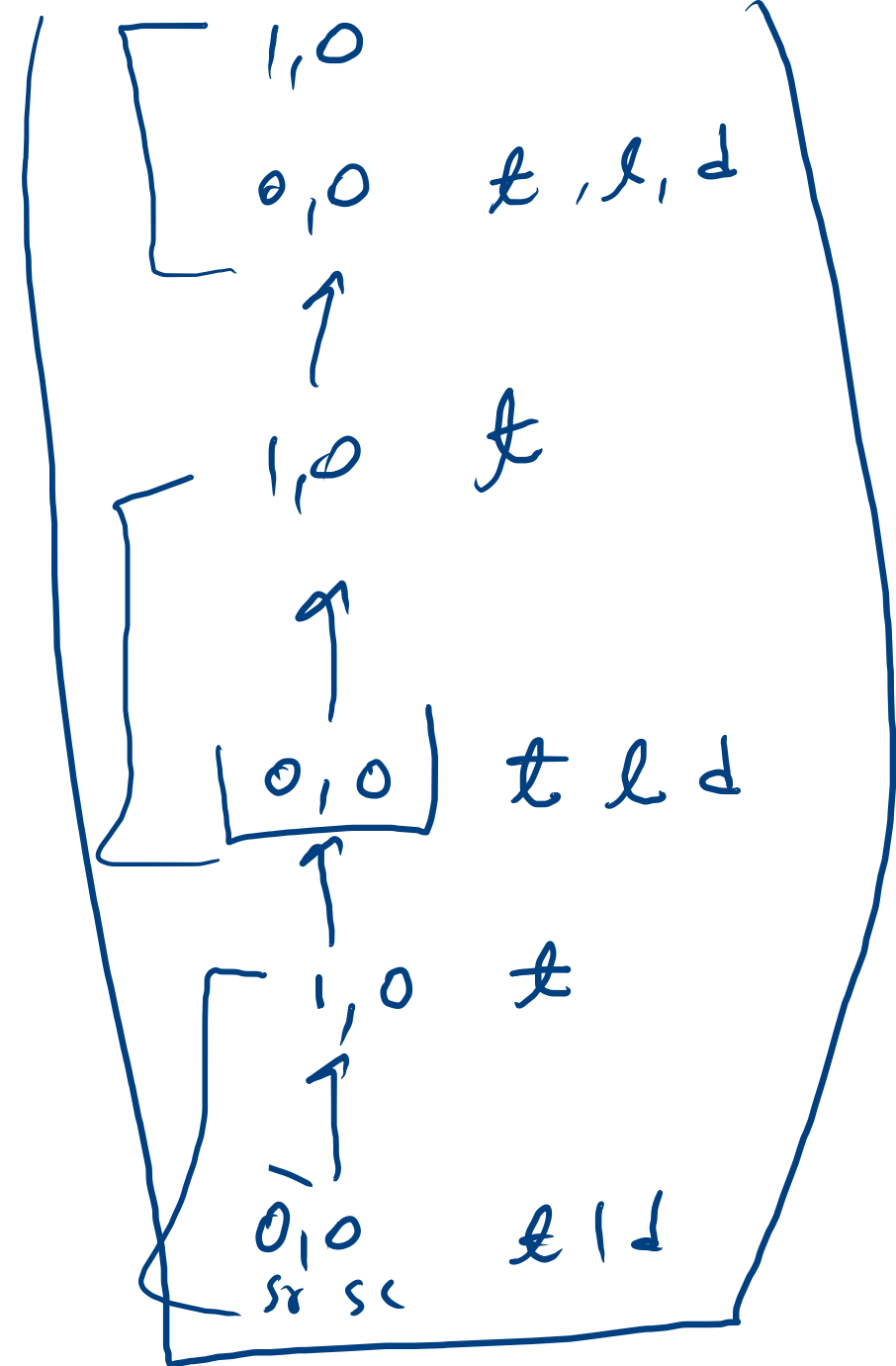
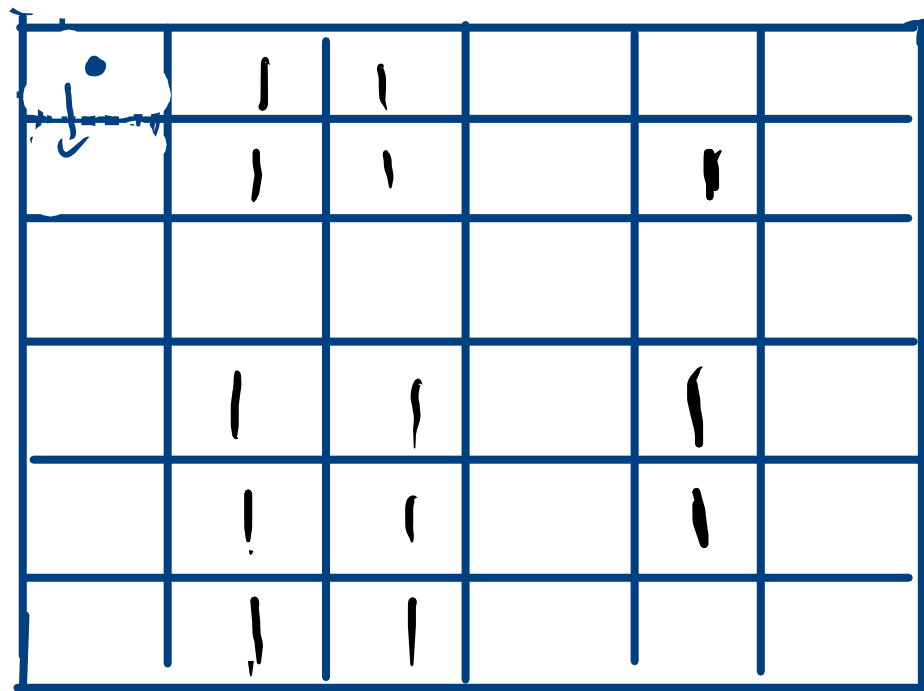
- ① d d r r r d d d r r
- ② d d r r r r r d d d
- ③ d d r r r t t r r d d l l l

```

public static void floodfill(int[][] maze, int sr, int sc, String asf) {
    if(sr<0 || sc<0 || sr>= maze.length || sc >= maze[0].length || maze[sr][sc]==1) return;
    if(sr==maze.length-1 && sc == maze[0].length-1){
        System.out.println(asf);
        return;
    }

    floodfill(maze, sr-1, sc, asf+"t");// t
    floodfill(maze, sr, sc-1, asf+"l");// l
    floodfill(maze, sr+1, sc, asf+"d");// d
    floodfill(maze, sr, sc+1, asf+"r");// r
}

```



```

if(sr<0 || sc<0 || sr>= maze.length ||
sc >= maze[0].length || maze[sr][sc]==1 ||
visited[sr][sc] == true) return;

if(sr==maze.length-1 && sc == maze[0].length-1){
    System.out.println(asf);
    return;
}

```

```

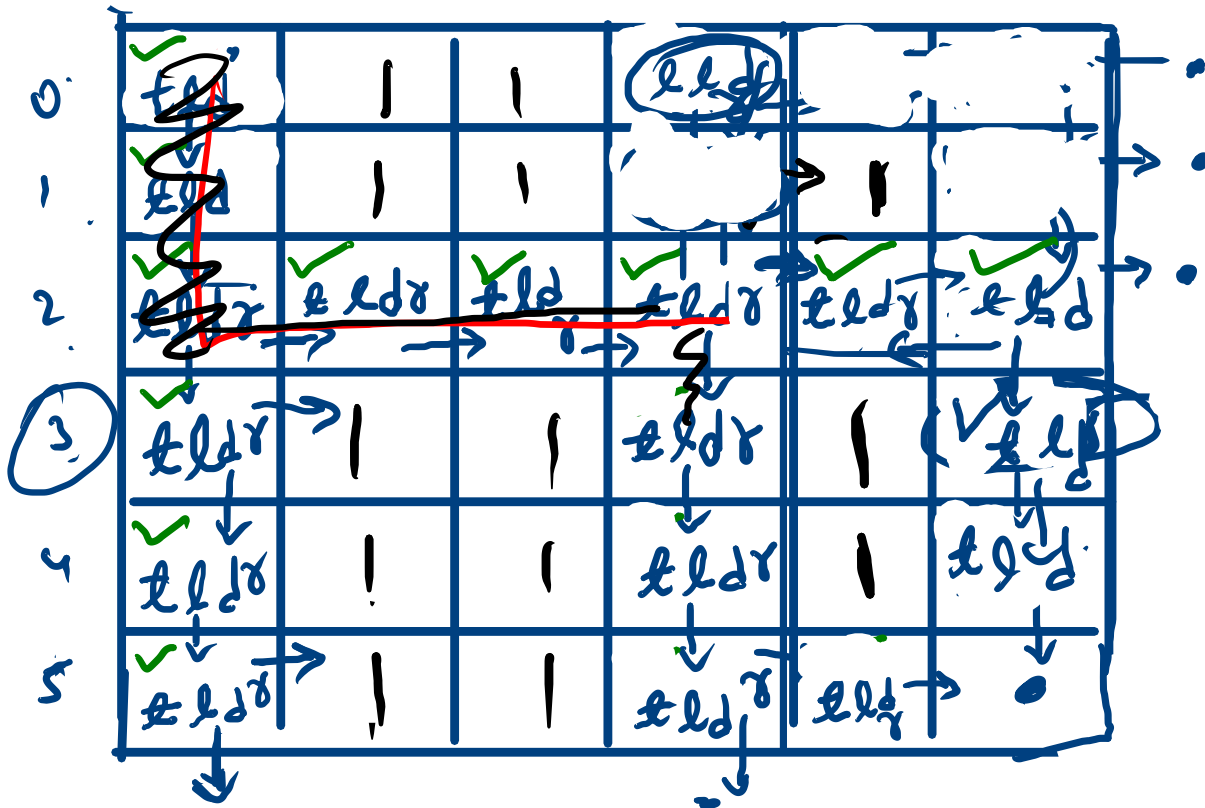
visited[sr][sc] = true;
floodfill(maze, sr-1, sc, asf+"t", visited); // t
floodfill(maze, sr, sc-1, asf+"l", visited); // l
floodfill(maze, sr+1, sc, asf+"d", visited); // d
floodfill(maze, sr, sc+1, asf+"r", visited); // r

```

with  $sr, sc = \text{false}$

maze

5



ddrrrrrrrr  
d l l l l l

d d r r r r d d d r r

arr

[ mark true

[ mark false

1, 3
2, 3
2, 2
2, 1
2, 0
1, 0
0, 0

```

if(sr<0 || sc<0 || sr>= maze.length ||
sc >= maze[0].length || maze[sr][sc]==1 ||
visited[sr][sc] == true)return;

if(sr==maze.length-1 && sc == maze[0].length-1){
    System.out.println(asf);
    return;
}

visited[sr][sc] = true;
floodfill(maze, sr-1, sc, asf+"t", visited); // t
floodfill(maze, sr, sc-1, asf+"l", visited); // l
floodfill(maze, sr+1, sc, asf+"d", visited); // d
floodfill(maze, sr, sc+1, asf+"r", visited); // r
visited[sr][sc] = false;

```

	0	1	2	3	4
0	1	1			1
1	1	1		1	
2	1	1	1		
3	1	1	1	1	
4	1	1	1	1	

2,2, "ddrr"

2,1, "ddr"

2,0, "dd"

1,0 "d"

0,0 ""

asc

t

tlrr

tlrr

tlr

tlr



0...20  
+v ←  
0 ←

[10 20 30 40 50]  
60

target = 60 ←

[10

20

✓

x

30

40

50]

shell

10, 20, 30, .

10, 50, .

20, 40, .

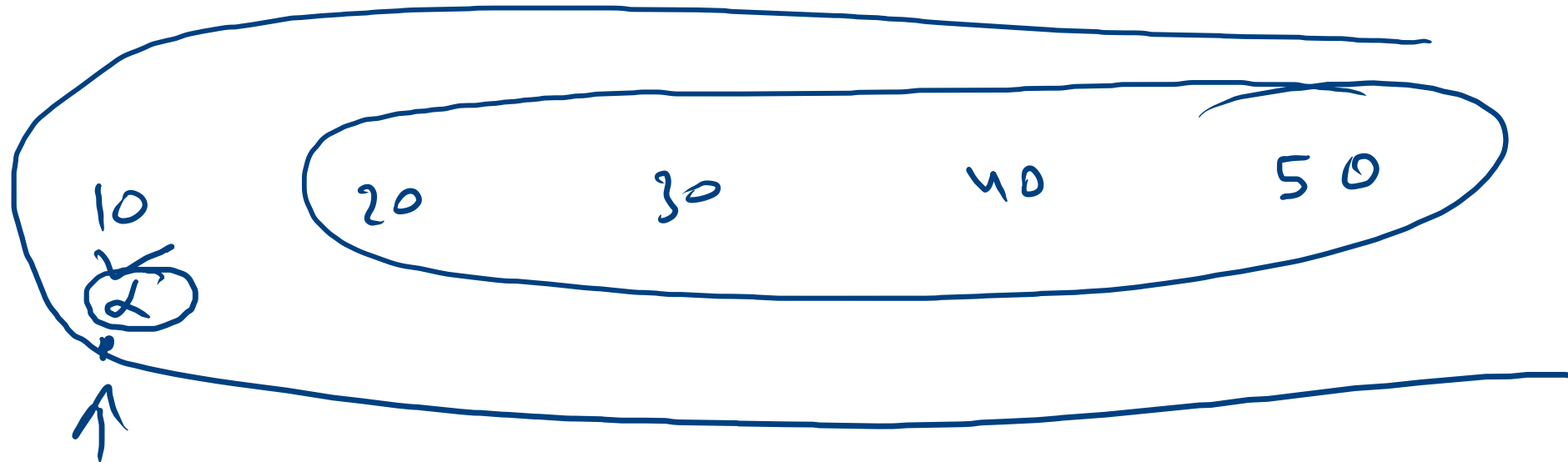
min 4

we only once



10 — 20, 30  
80

—, 20, 40



expectation  $ind = 0$   
 $far \rightarrow 60$

Pair  
 $ind + 1$

$far = 50$

$ind + 1$

$far = 60$

6  
5



subset



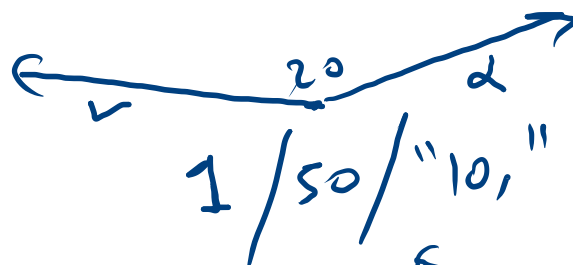
4/0/"10,20,30,0" 4/0/"10,20,30"



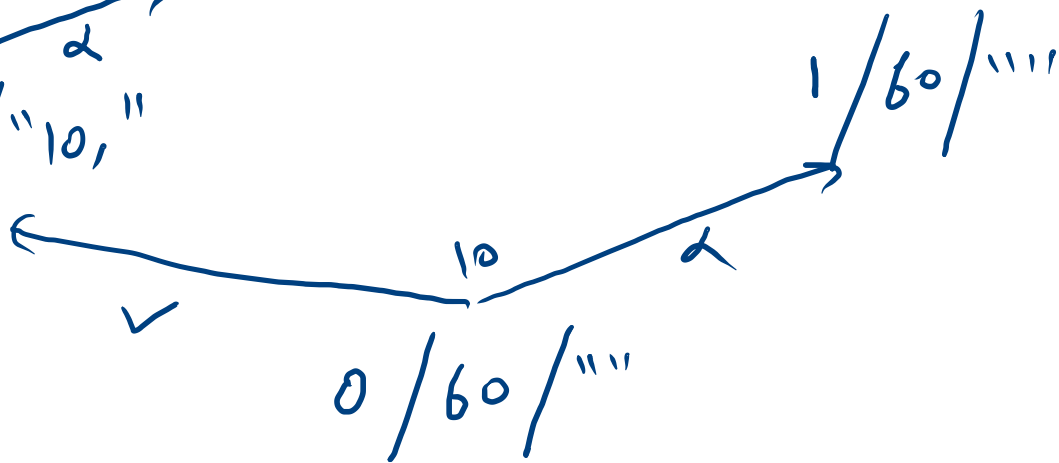
3/0/"10,20,30" 3/30/"10,20,"



2/30/"10,20," 2/50/"10,"



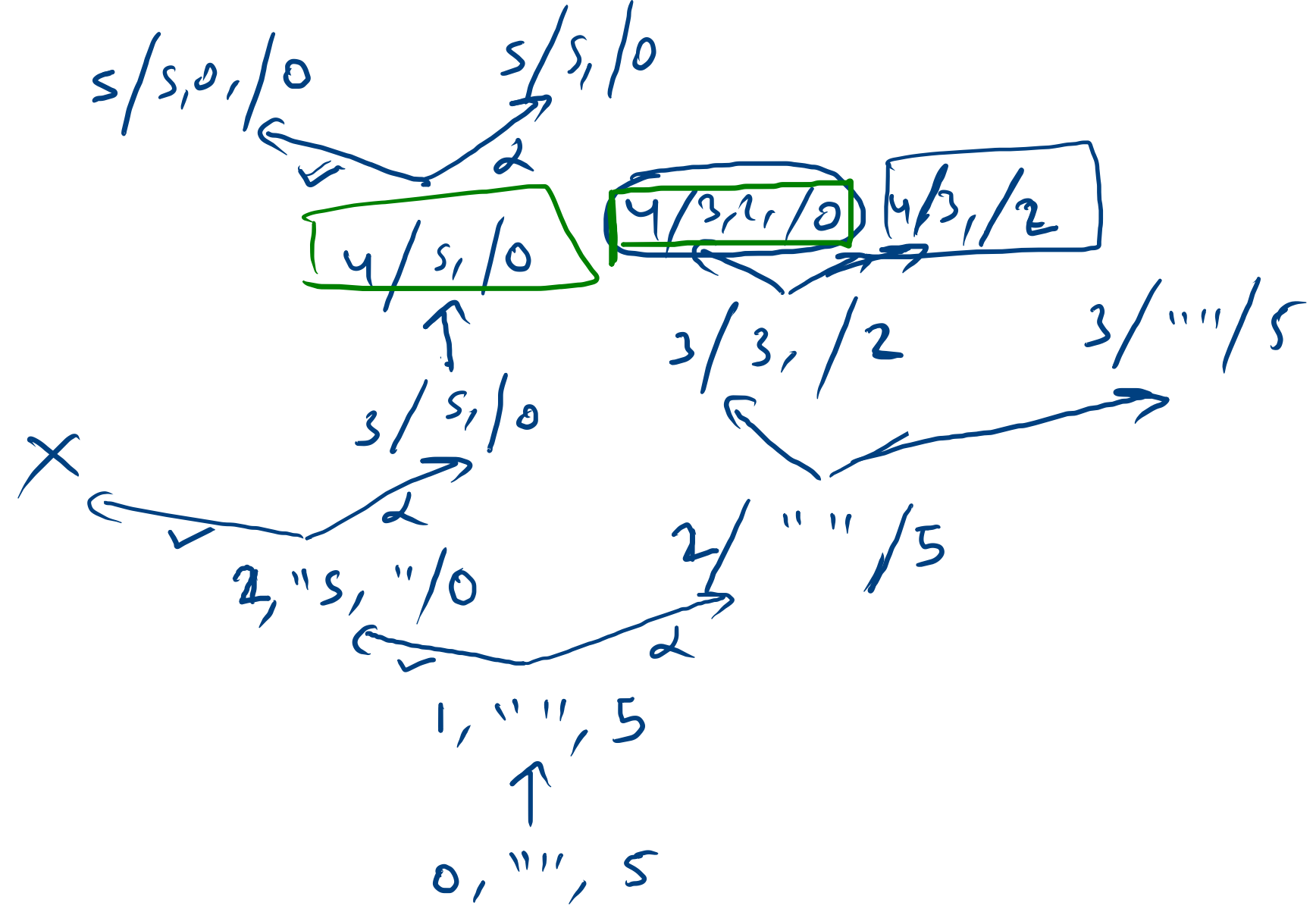
1/50/"10,"



0/60/"

1/60/"

4 0  
 3 2  
 2 3  
 1 5  
 0 7  
 tar = 5



S,  
 3,2,

```

public static void printTargetSumSubsets(int[] arr, int idx, String set, int tar) {
    if(idx == arr.length){
        if(tar == 0)
            System.out.println(set+".");
        return;
    }
    // include
    if(arr[idx] <= tar)
        printTargetSumSubsets(arr, idx+1, set+arr[idx]+", ", tar-arr[idx]);
    // xclude
    printTargetSumSubsets(arr, idx+1, set, tar);
}
  
```

2 < 2 - 2  
 3, 2, 2 - 2