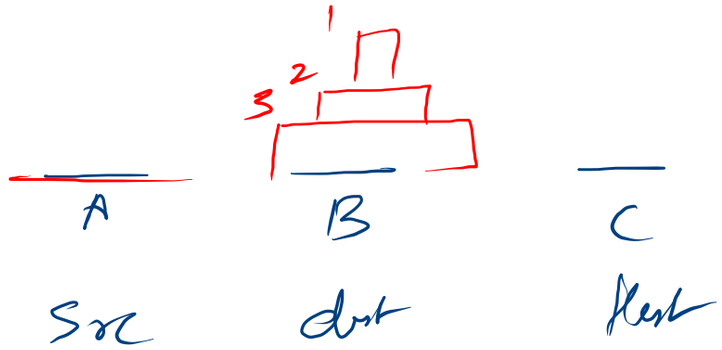
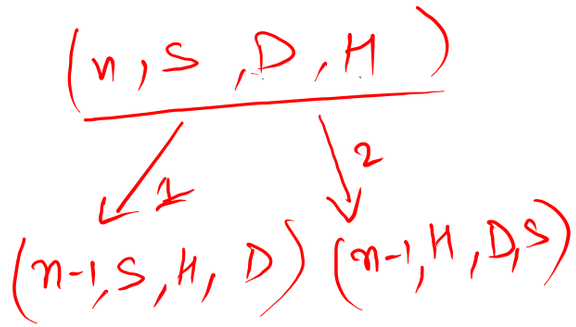
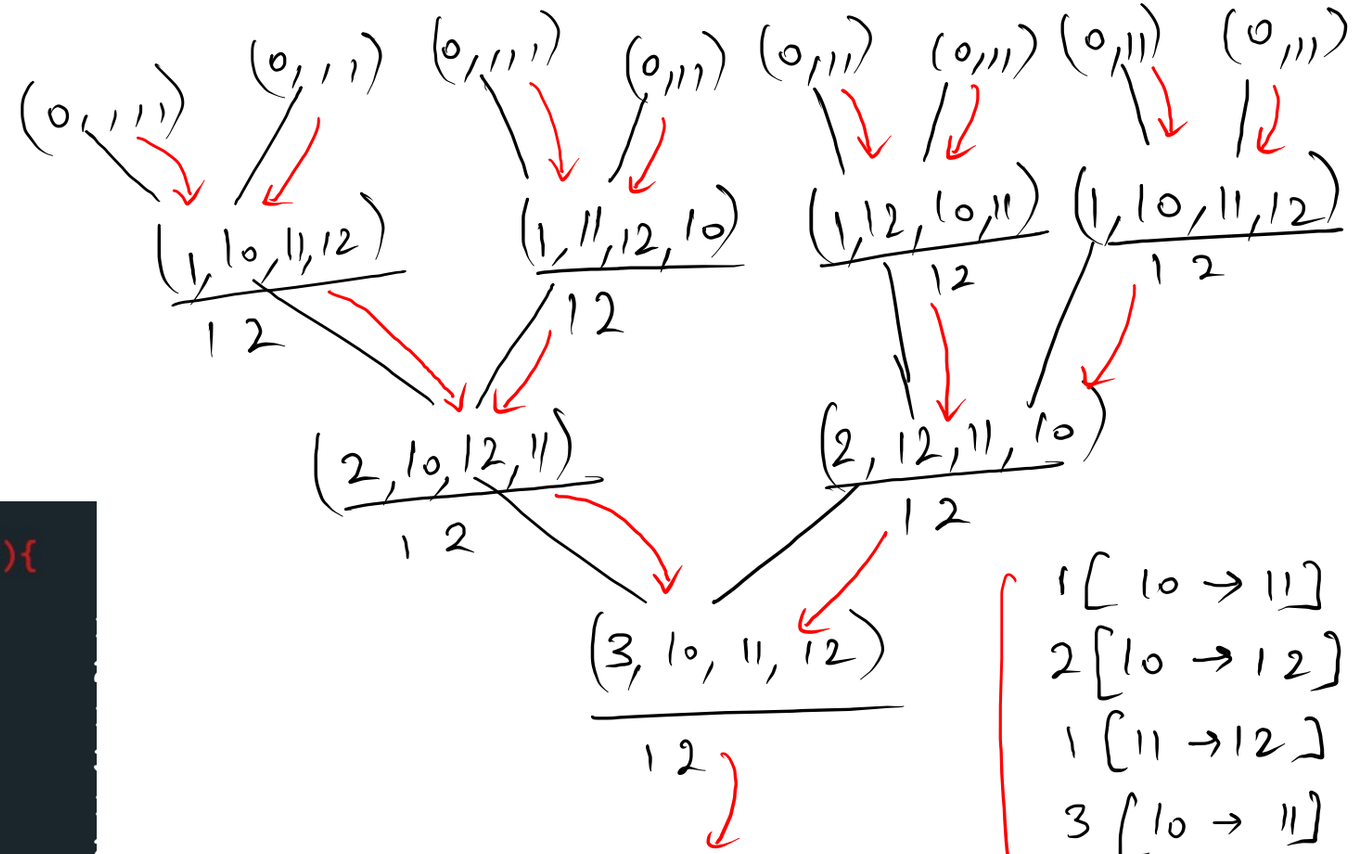


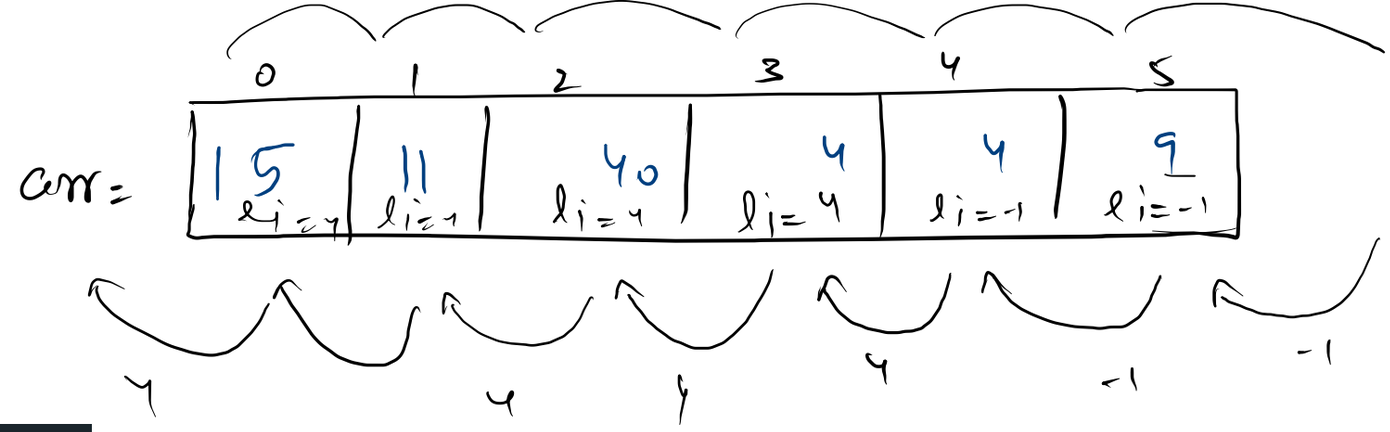





```java
3        10        11        12
public static void toh(int n, int src, int dest, int helper){
    if(n == 0){
        return;
    }
    toh(n-1,src,helper,dest); -1
    System.out.println(n+"["+src+" -> "+dest+"]");
    toh(n-1,helper,dest,src); - 2
}
```

1 [10 → 11]
2 [10 → 12]
1 [11 → 12]
3 [10 → 11]
1 [12 → 10]
2 [12 → 11]
1 [10 → 11]

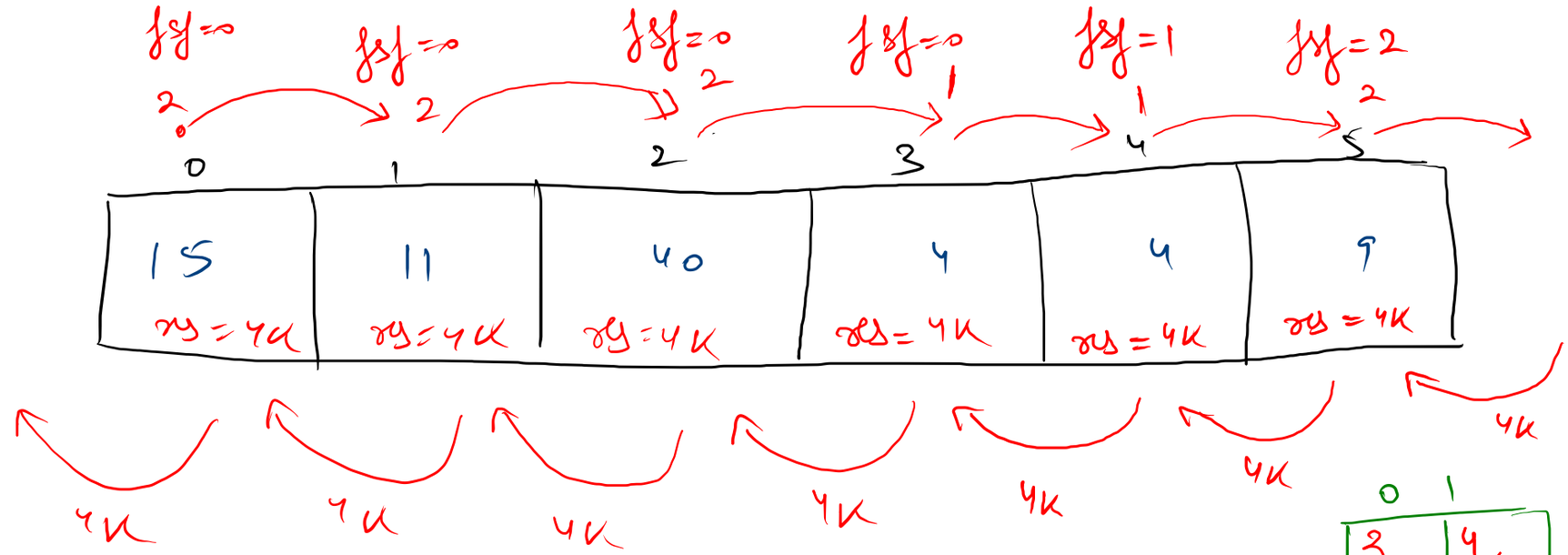
K = 4



0

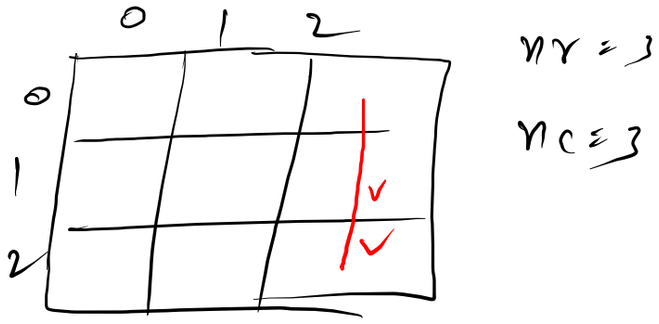
```java
public static int lastIndex(int[] arr, int idx, int k){
    if(idx == arr.length){
        return -1;
    }
    int li = lastIndex(arr,idx+1,k);

    if(li == -1){
        if(arr[idx] == k){
            return idx;
        }else{
            return -1;
        }
    }else{
        return li;
    }
}
```

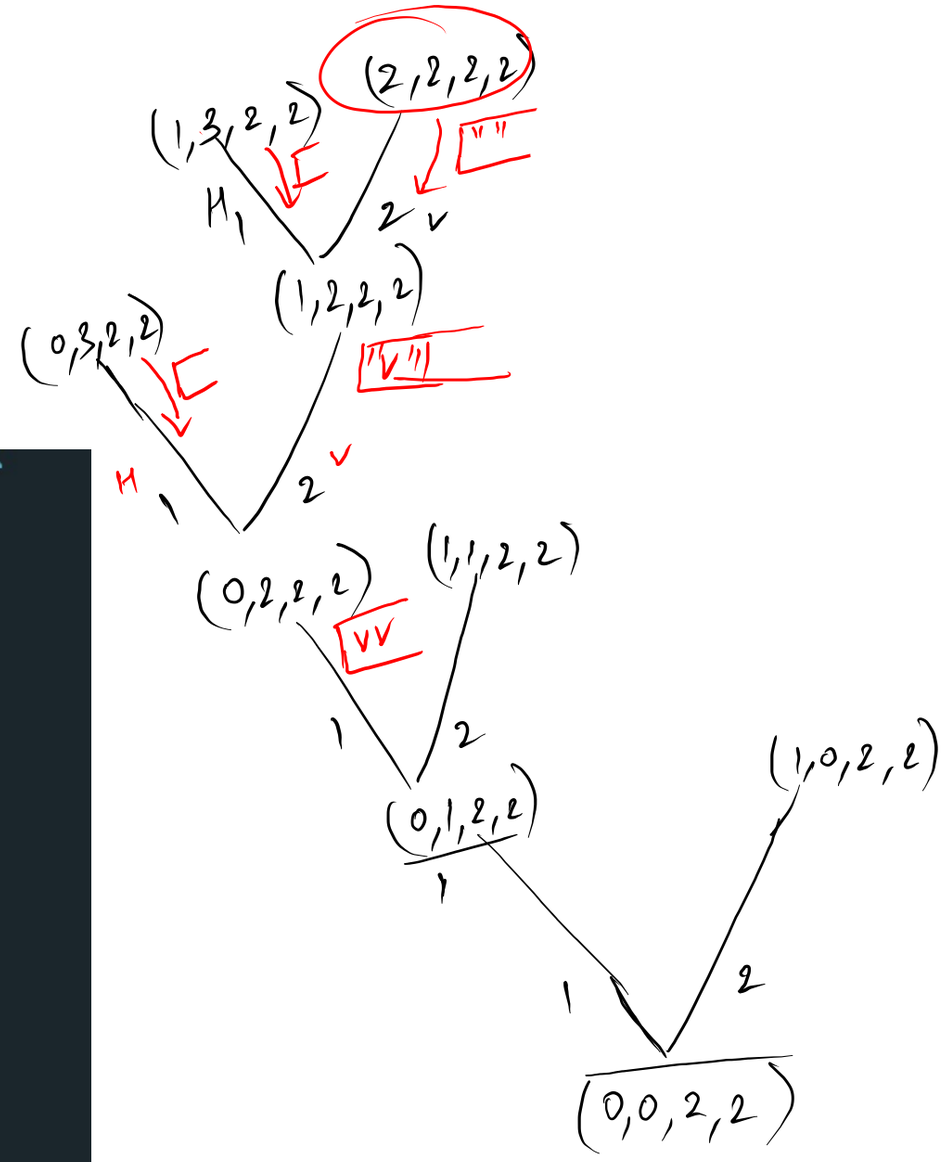
0




```java
public static int[] allIndices(int[] arr, int k, int idx, int fsf) {
  if (idx == arr.length) {
    return new int[fsf];
  }
  if (arr[idx] == k) {
    int res[] = allIndices(arr, k, idx + 1, fsf + 1);
    res[fsf] = idx;
    return res;
  } else {
    int res[] = allIndices(arr, k, idx + 1, fsf);
    return res;
  }
}
```

```
0 -> .;
1 -> abc
2 -> def
3 -> ghi
4 -> jkl
5 -> mno
6 -> pqrs
7 -> tu
8 -> vwx
9 -> yz
```

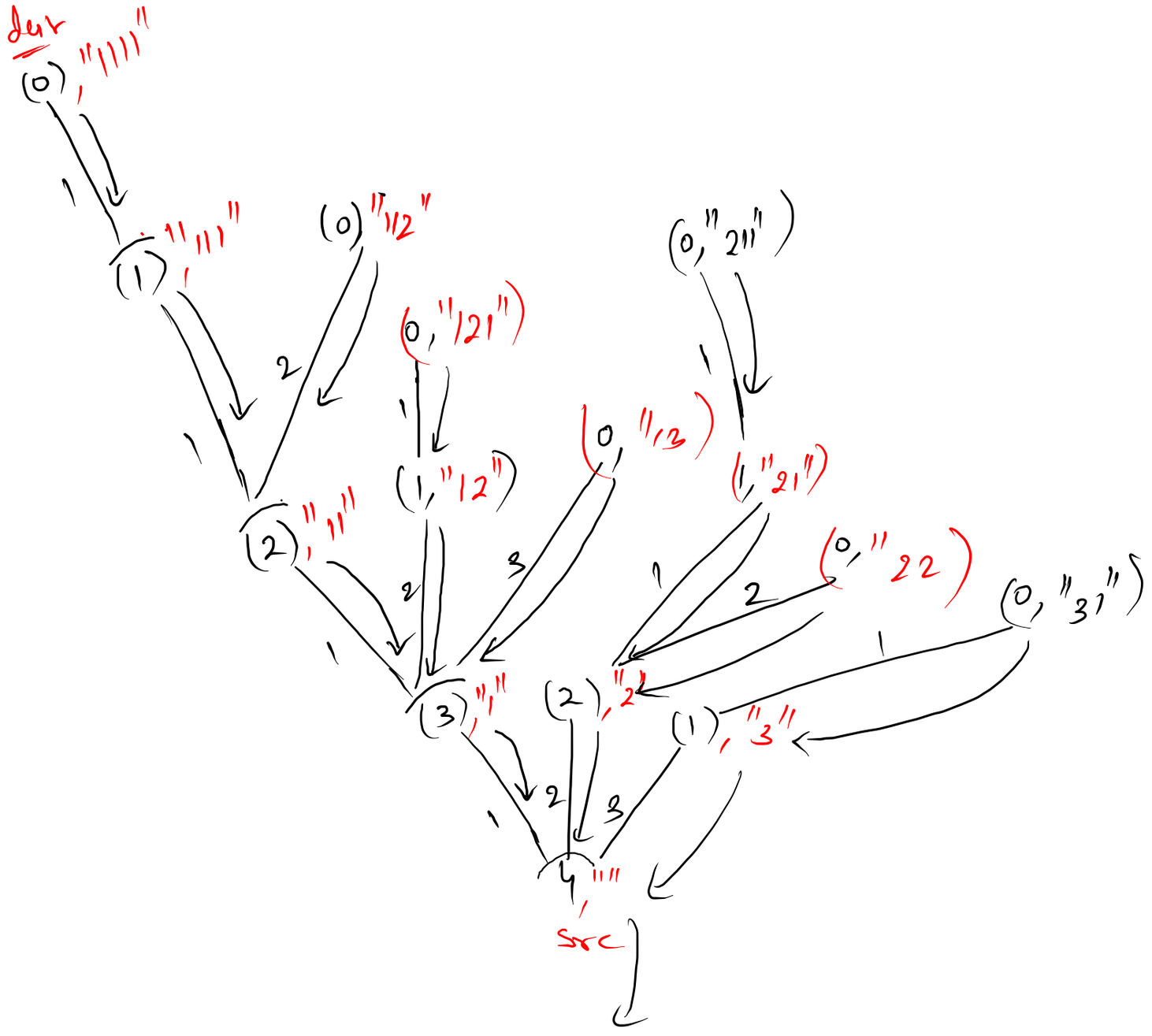


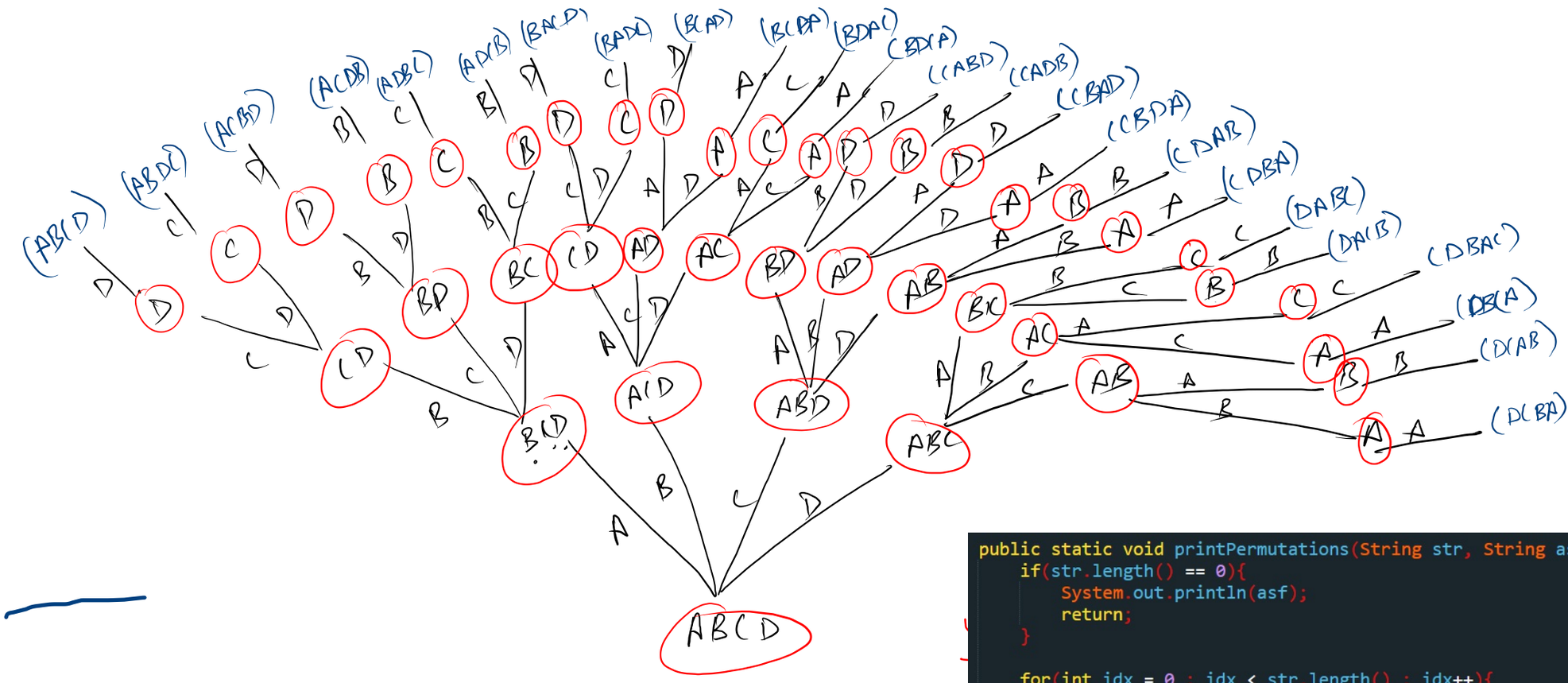



```java
public static ArrayList<String> getMazePaths(int sr, int sc, int dr
int dc) {
    if(sr > dr || sc > dc){
        return new ArrayList<String>();
    }
    if(sr == dr && sc == dc){
        ArrayList<String> myPath = new ArrayList<>();
        myPath.add("");
        return myPath;
    }
    ArrayList<String> HPath = getMazePaths(sr,sc+1,dr,dc);
    ArrayList<String> VPath = getMazePaths(sr+1,sc,dr,dc);

    ArrayList<String> myPath = new ArrayList<>();
    for(String path : HPath){
        myPath.add('h'+path);
    }
    for(String path : VPath){
        myPath.add('v'+path);
    }
    return myPath;
}
```

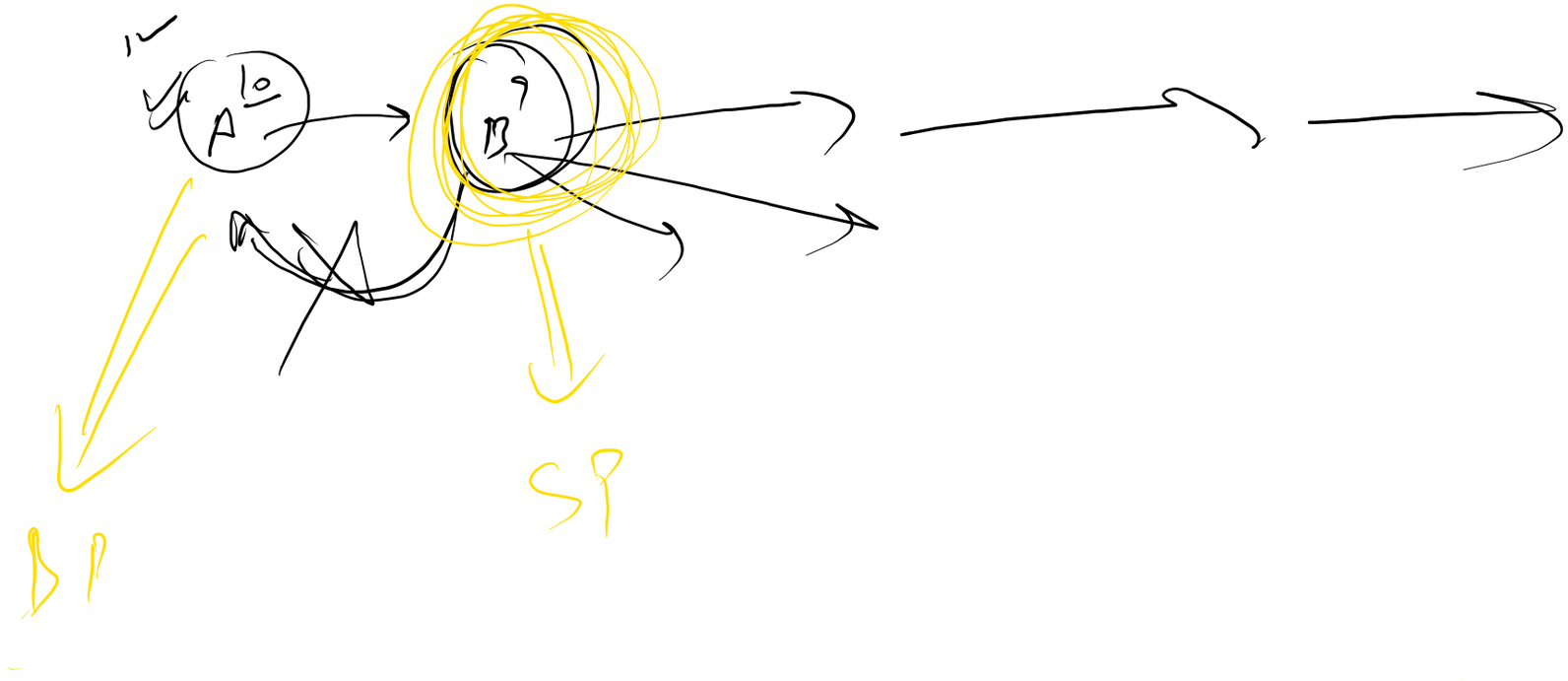
```java
public static void printStairPaths(int n, String psf) {
    if(n == 0){
        System.out.println(psf);
        return;
    }
    if(n-1 >= 0){
        printStairPaths(n-1,psf+"1");
    }
    if(n-2 >= 0){
        printStairPaths(n-2,psf+"2");
    }
    if(n-3 >= 0){
        printStairPaths(n-3,psf+"3");
    }
}
```

[
1111
112
121
13
211
22
31
]





```
public static void printPermutations(String str, String asf) {
    if(str.length() == 0){
        System.out.println(asf);
        return;
    }

    for(int idx = 0 ; idx < str.length() ; idx++){
        char ch = str.charAt(idx);
        String lp = str.substring(0,idx);
        String rp = str.substring(idx+1);
        printPermutations(lp+rp,asf+ch);
    }
}
```


A
B
BP
SP