Applied Algorithms

CSCI-B505 / INFO-I500

Lecture 8.

Recursions

M. Oguzhan Kulekci, September 19/20, 2022

- Recursions

Recursions

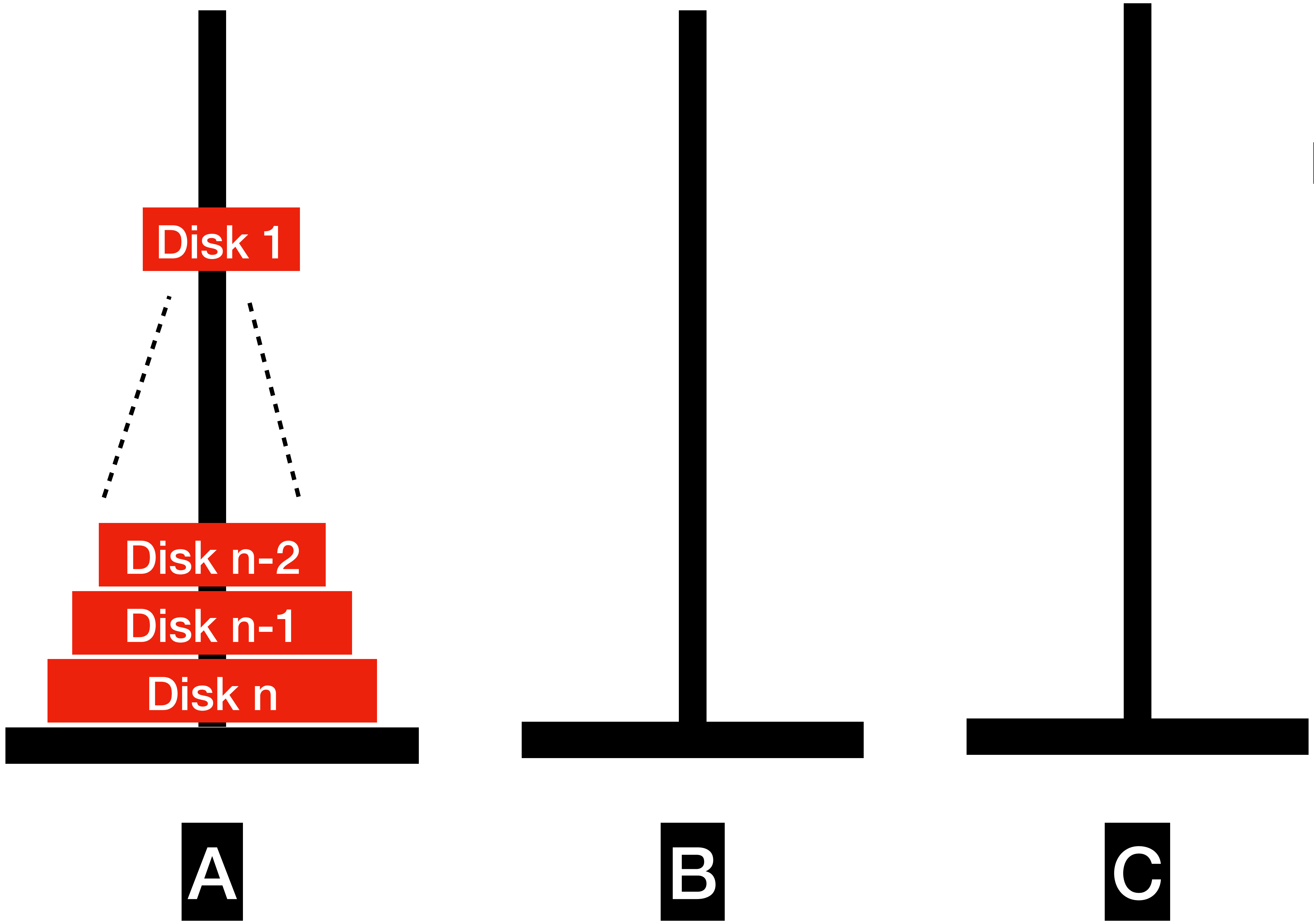
A sequence where each element is formulated according to previous ones.

- Fibonacci numbers: $F_n = F_{n-1} + F_{n-2}, \forall n > 1$, with $F_0 = 1,\ \ F_1 = 1.$
- The factorial $Fact(n) = n \cdot Fact(n-1)$, with $Fact(1) = 1.$
- Many other examples…

- *Recursion can serve as a powerful tool to solve complex relations.*
- *Recursive functions in programming are the functions calling themselves !!!*

Recursions

Towers of Hanoi ...



Move all disks to tower C without violating the rules
-only one disk moves at a time
-a larger disk cannot be placed over a smaller disk

Assume moving (n-1) disks to C takes T(n-1) steps.
Then moving n disks can be done with

- Move (n-1) disks to B, which takes T(n-1) steps.
- Move largest disk at the bottom to tower C.
- Move (n-1) disks on B to C, again in T(n-1) steps.

It takes **T(n) = 2T(n-1) + 1** total steps.

$$T(n) = 2T(n-1) + 1 = 4T(n-2) + 2 + 1$$
$$= 2^3T(n-3) + 2^2 + 2 + 1$$
$$= 2^{n-1}T(1) + 2^{n-2} + \ldots + 2 + 1$$
$$= 2^{n-1} + 2^{n-2} + \ldots + 2 + 1$$
$$= 2^n - 1 \in O(2^n)$$

Backward and forward substitutions can be used for solutions of such recursions

Recursions

How can we generate all permutations of a sequence ?

- Assume we need the permutations of n items.
- The first position can be one of n items.
- Following positions are the permutation of the remaining (n-1) items, which can be generated again with the same method.

$$\pi(a_1, a_2, \ldots, a_n)$$

$$a_1, \pi(a_2, \ldots, a_n)$$

$$a_1, a_2, \pi(a_3, \ldots, a_n)$$

$$a_1, a_n, \pi(a_2, \ldots, a_n)$$

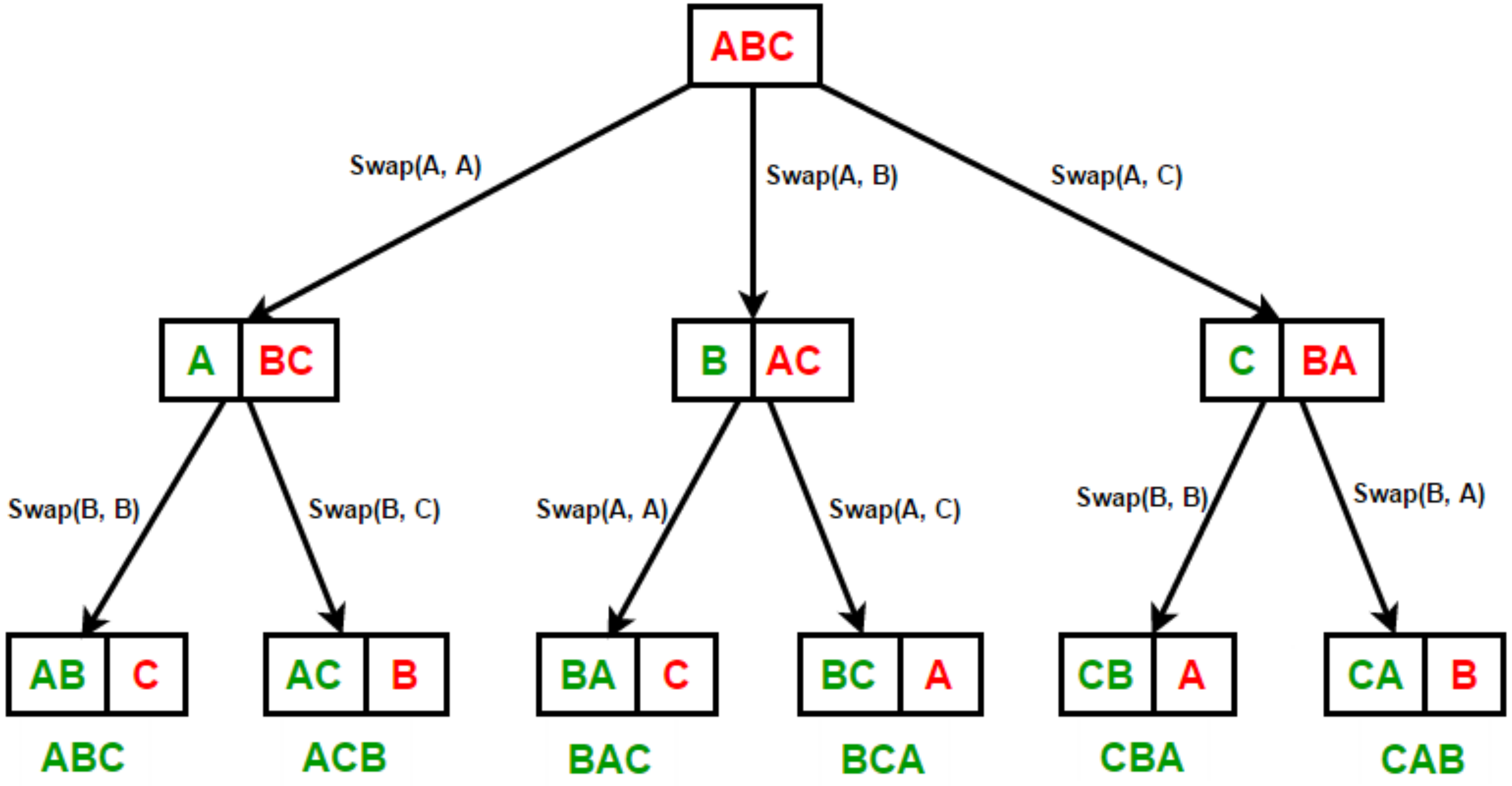
$$a_2, \pi(a_1, \ldots, a_n)$$

$$a_n, \pi(a_1, \ldots, a_{n-1})$$

Recursions

How can we generate all permutations of a sequence ?

```
perm(arr, fixed, n)
  if (fixed = n-1)
      printArray
  else
      for(j=fixed to n-1)
        swap(arr[fixed],arr[j])
        perm(arr, fixed+1,n);
        swap(arr[fixed],arr[j])
```



Recursion Tree for string "ABC"

`perm(ABC,0,3)`

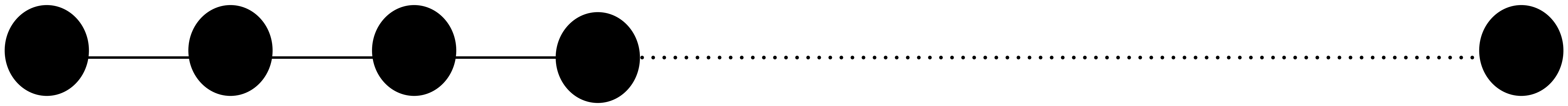
That is actually a **decrease-and-conquer** approach as in towers of Hanoi !

Recursions

Enumerating d-dimensional vectors.

Assume $X = \langle x_1, x_2, \ldots, x_d \rangle$ is a d-dimensional vector, where $x_i > 0$.

How many such distinct vectors can be constructed, when the sum of all dimensions, $S = x_1 + x_2 + x_3 + \ldots x_d$, is given.



$$C = \binom{s-1}{d-1} \approx d \cdot \log s$$

Recursions

Enumerating d-dimensional vectors.

What if the x_i values are allowed to be zero as well on the d-dimensional vector $X = \langle x_1, x_2, \ldots, x_d \rangle$, $x_i \geq 0$, Again we are given the sum S.

$$X = \langle x_1, x_2, x_3, x_4 \rangle$$

$$x_1 + x_2 + x_3 + x_4 = S = 10$$

Case 1: **None** of the x_i s is zero $\binom{4}{0} \cdot \binom{9}{3}$

Case 2: **1** of the x_i s is zero, $\binom{4}{1} \cdot \binom{9}{2}$

Case 3: **2** of the x_i s is zero, $\binom{4}{2} \cdot \binom{9}{1}$

Case 3: **3** of the x_i s is zero, $\binom{4}{3} \cdot \binom{9}{0}$

Yes, we still do not need recursion ***to count****, but, what if …*

Recursions

Enumerating d-dimensional vectors.

What if the x_i values are allowed only to be in a range, e.g. $x_i \in \{1,2,...k\}$, on the d-dimensional vector $X = \langle x_1, x_2, \ldots, x_d \rangle$.

- Now, it is more complicated, and recursion can help.
- Assume x_i is fixed to a possible value $z \in \{1,2,...k\}$, then the rest of the vector is again the same problem with the dimension reduced by one and the sum reduced by the z.
- Thus, we can traverse the dimensions of the vector from $i = 1$ to d by all possible values, and for each such case recurse for the remaining vector.

Assume we have a d dimensional integer vector L.

$$L = \langle \ell_1, \ell_2, \ell_3, \ldots, \ell_d \rangle$$

We also know that the inner sum is v and each dimension is between 1 and k.

$$v = \ell_1 + \ell_2 + \ell_3 + \cdots + \ell_d\,, \quad 1 \le \ell_i \le k$$

Assuming all distinct L vectors of given v and k values are ordered, the rank of a vector in this ordered list specifies the vector.

$$d = 3, v = 6, k = 3$$

0	1	2	3
1	1	3	2
2	2	1	3
3	2	2	2
4	2	3	1
5	3	1	2
6	3	2	1

When rank is given as 4, the vector is $\langle 2,3,1 \rangle$.

If the vector is given as $\langle 3,1,2 \rangle$, then its rank is 5.

Algorithm 1: $\psi(k, d, v)$

Input:
k: Maximum value of a dimension.
d: The number of dimensions.
v: The inner sum of the vectors.
Output:
Number of distinct d dimensional vectors with an inner sum of v.

if $\boxed{(v > k \cdot d) || (v < d)}$ **then**
  return *0*;
if $\boxed{(d = 1) || (v = d)}$ **then**
  return *1*;
if $\boxed{(v = d + 1)}$ **then return** d;
if $(1 < v + k - k \cdot d)$ **then**
  $\alpha = v + k - k \cdot d$
else
  $\alpha = 1$
if $(k < v - d + 1)$ **then**
  $\beta = k$
else
  $\beta = v - d + 1$
$sum = 0$;
for $(i = \alpha; i \le \beta; i+ = 1)$ **do**
  $\boxed{sum+ = \psi(k, d - 1, v - i);}$
end
return sum;

$\psi(k, d, v)$:

The total number of distinct d dimensional vectors whose inner sum is v , where each dimension is in range $[1,k]$.

0, *no such vector since* $d \le v \le k \cdot d$

1, *only one way to construct it, either* $\langle 1,1,\ldots,1 \rangle$ *or* $\langle v \rangle$

d, *There are d ways to construct it*

$$\left.\begin{array}{c} \langle 2,1,1,\ldots,1 \rangle \\ \langle 1,2,1,\ldots,1 \rangle \\ \ldots\ldots \\ \langle 1,1,1,\ldots,2 \rangle \end{array}\right\} \text{d items}$$

otherwise, $\sum_{i=\alpha}^{i=\beta} \psi(k, d-1, v-i)$, *where*

$$\alpha = \begin{cases} 1, \textit{if } \ v - k(d-1) \le 1 \\ v - k(d-1), \textit{otherwise} \end{cases}$$

$$\beta = \begin{cases} k, \textit{if } \ v - (d-1) > k \\ v - (d-1), \textit{otherwise} \end{cases}$$

Iterate over all possible values for one dimension and recursively count on the remaining (d-1) dimensions with the updated sum v !

Algorithm 1: $\psi(k, d, v)$

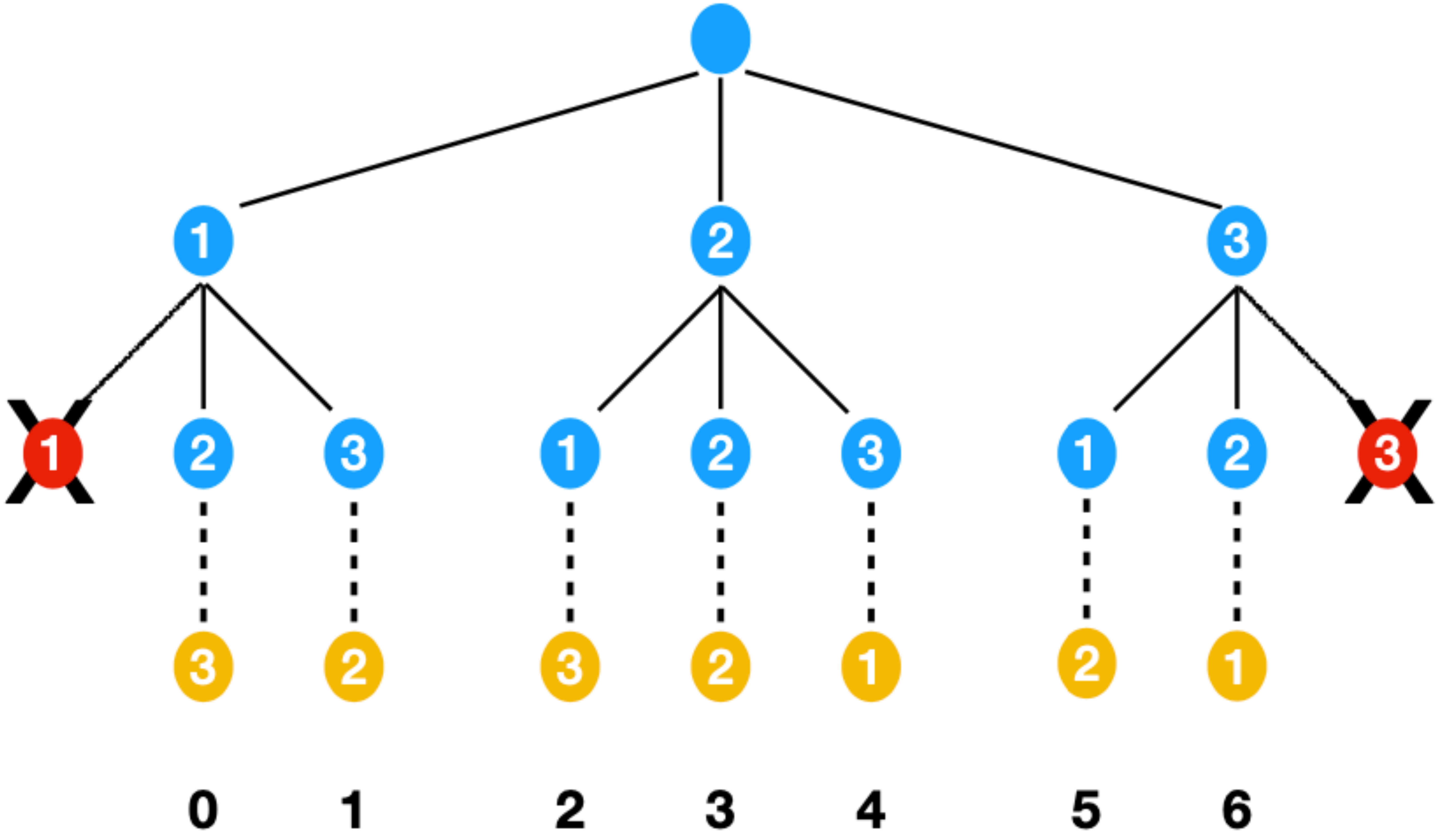
Input:
k: Maximum value of a dimension.
d: The number of dimensions.
v: The inner sum of the vectors.
Output:
Number of distinct d dimensional
vectors with an inner sum of v
.

if $(v > k \cdot d) || (v < d)$ **then**
  return *0*;
if $(d = 1) || (v = d)$ **then**
  return *1*;
if $(v = d + 1)$ **then return** d;
if $(1 < v + k - k \cdot d)$ **then**
  $\alpha = v + k - k \cdot d$
else
  $\alpha = 1$
if $(k < v - d + 1)$ **then**
  $\beta = k$
else
  $\beta = v - d + 1$
$sum = 0$;
for $(i = \alpha; i \le \beta; i += 1)$ **do**
  $sum += \psi(k, d - 1, v - i)$;
end
return sum;

$\psi(k, d, v)$:

The total number of distinct d dimensional vectors whose inner sum is v, where each dimension is in range $[1,k]$.

This is akin to constructing the d-ary tree of height (k-1), where each inner node only creates children that accompany with the restrictions.
For example, if d=3, k=3, v=6 then…



Reading assignment

- Read the recursion and divide-and-conquer chapters from the text books.

- For the d-dimensional array counting problem you can refer to the paper here

  http://www.stringology.org/event/2020/p03.html