UPPSALA UNIVERSITET

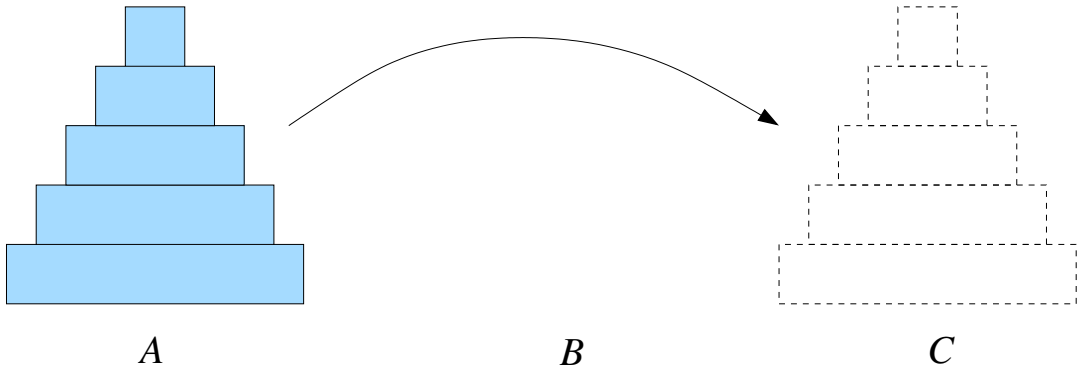
MA1: L02

Tom Smedsaas

Recursion: Two examples.

UPPSALA UNIVERSITET

Tower of Hanoi



A stack of tiles of different sizes to be moved from place **A** to place **C**, observing the following rules:

- Only one tile may be moved at a time
- You must never put a larger tile on top of a smaller one

In order for this to be possible, a "help place" ***B*** is needed.

UPPSALA UNIVERSITET

Tower of Hanoi

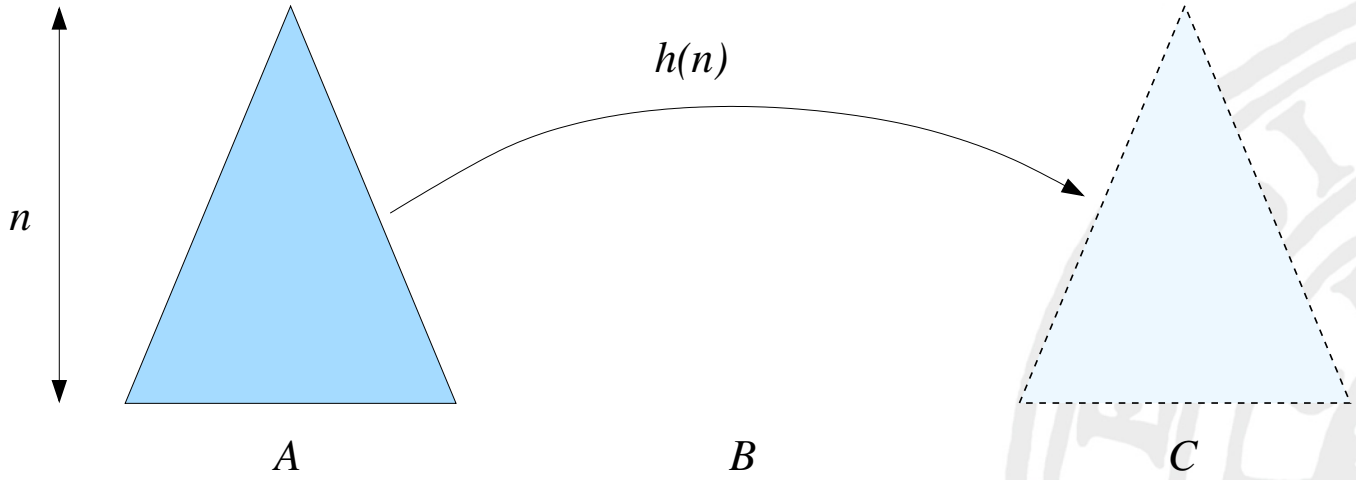
A small demonstration.

There are more on YouTube!

Tower of Hanoi

Denote the problem of moving *n* tiles from one place to another by *h(n)*.

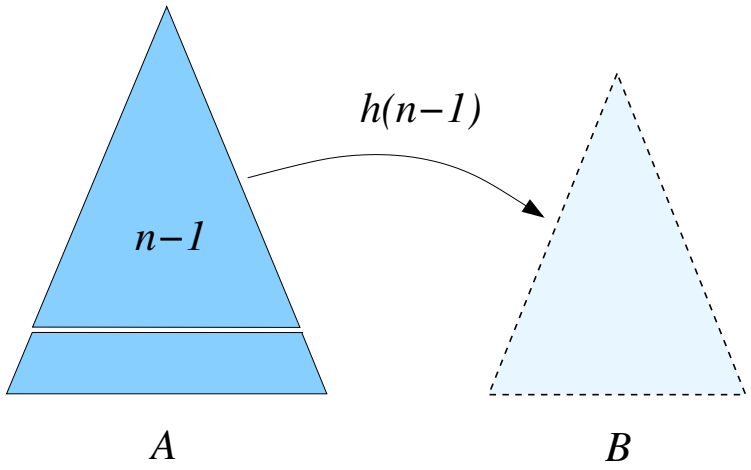
So we want to solve the problem :



Hanois torn

In order to move the bottom tile from A to C, all other tiles must be on B.

So we have to start by solving the problem of moving *n*-1 tiles from A to B:

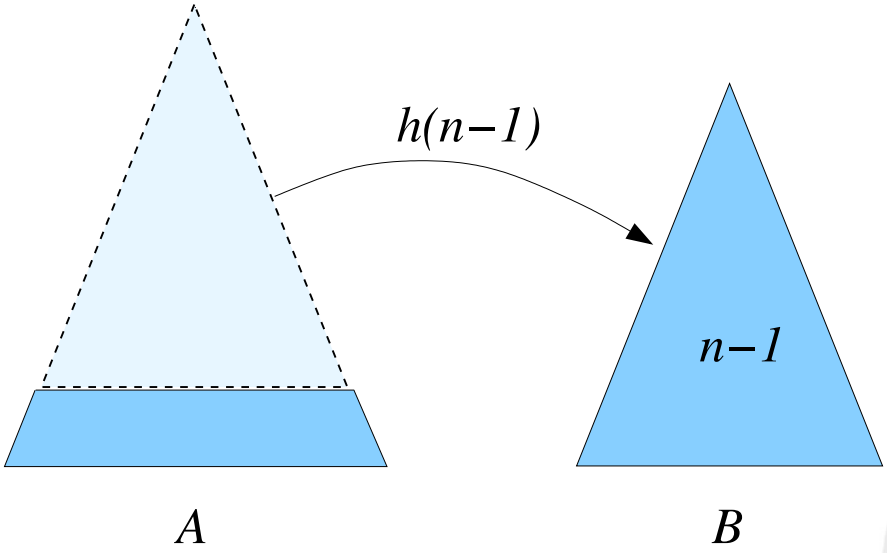


1. Move *n*-1 tiles from *A* to *B* using *C* as temporary storage.

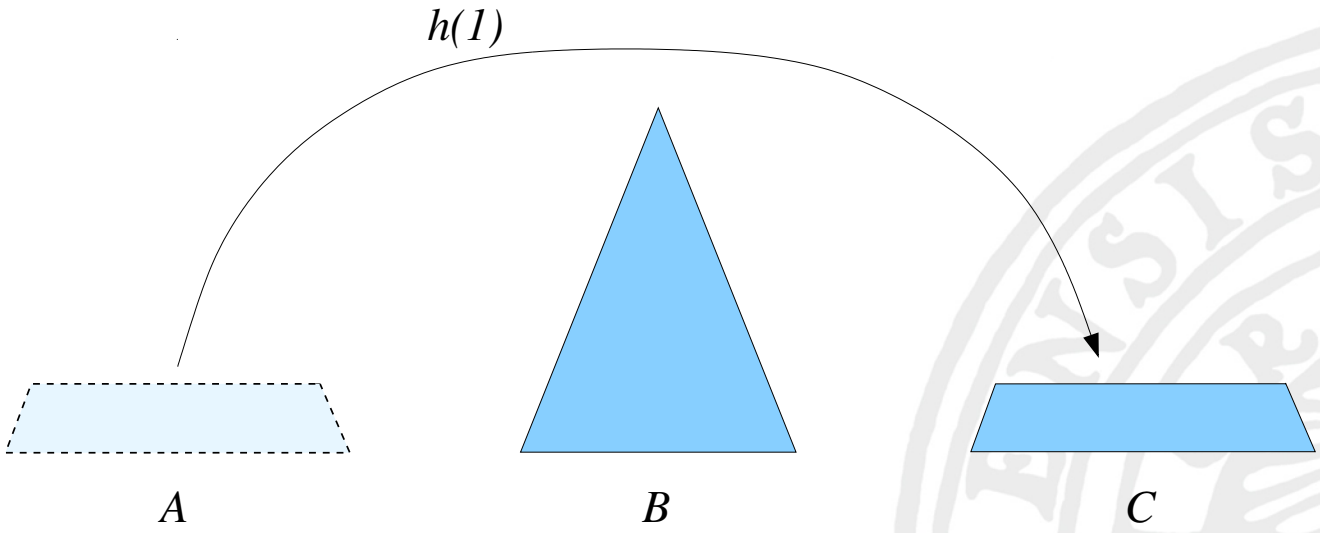
.

UPPSALA UNIVERSITET

Tower of Hanoi



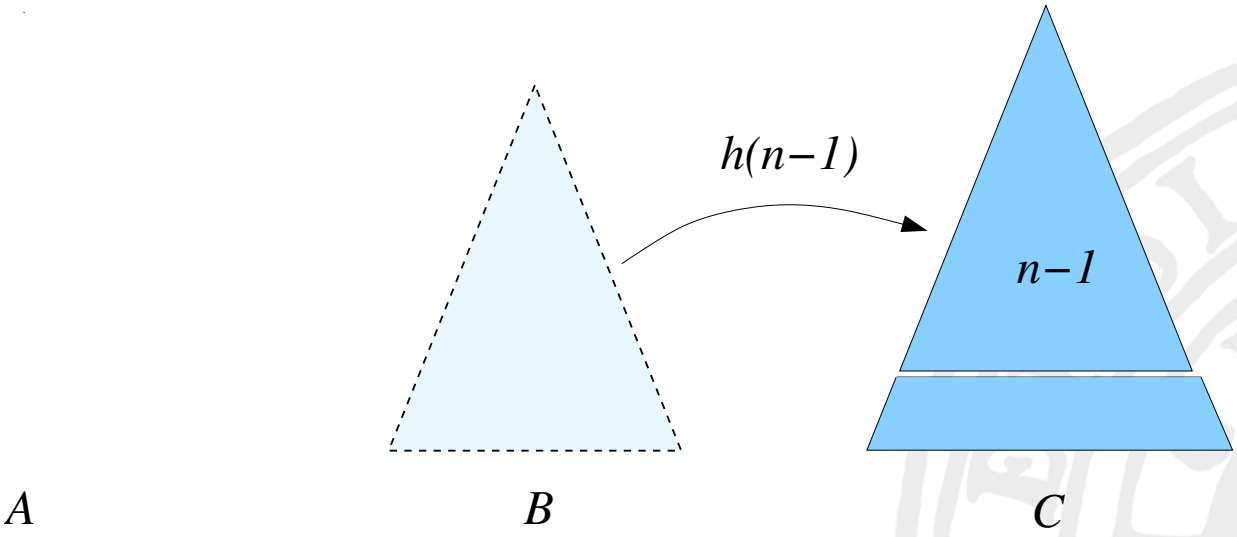
Tower of Hanoi



2. Move 1 tile from *A* ol *C.*

.

Tower of Hanoi



3. Move n – 1 tiles from B to C using A as temporary storage

UPPSALA UNIVERSITET

Tower of Hanoi

Algorithm:

1. Move *n* - 1 tiles from *A* to *B* using *C.*
2. Move 1 tile from *A* to *C.*
3. Move *n* – 1 tiles from *B* to *C* using *A.*

Note that there are two recursive calls and that the information about *from*, *to* and *using* varies. The places A, B and C thus switch roles.

Hint 1: You get the simplest code if you use n = 0 as base case!

Hint 2: It is enough with 5 lines of code inclusive `def`, `if` and `else` to solve the task.

UPPSALA UNIVERSITET

The exchange problem

UPPSALA UNIVERSITET

The exchange problem

Suppose we have coins/bills with the following values: 1, 5, 10, 50, 100.
In how many different ways can we exchange a given amount?

Example: The amount 12 can be exchanged in 4 different ways :

1x10 + 2x1, 2x5 + 2x1, 1x5 + 7x1 and 12x1

We want to write a function

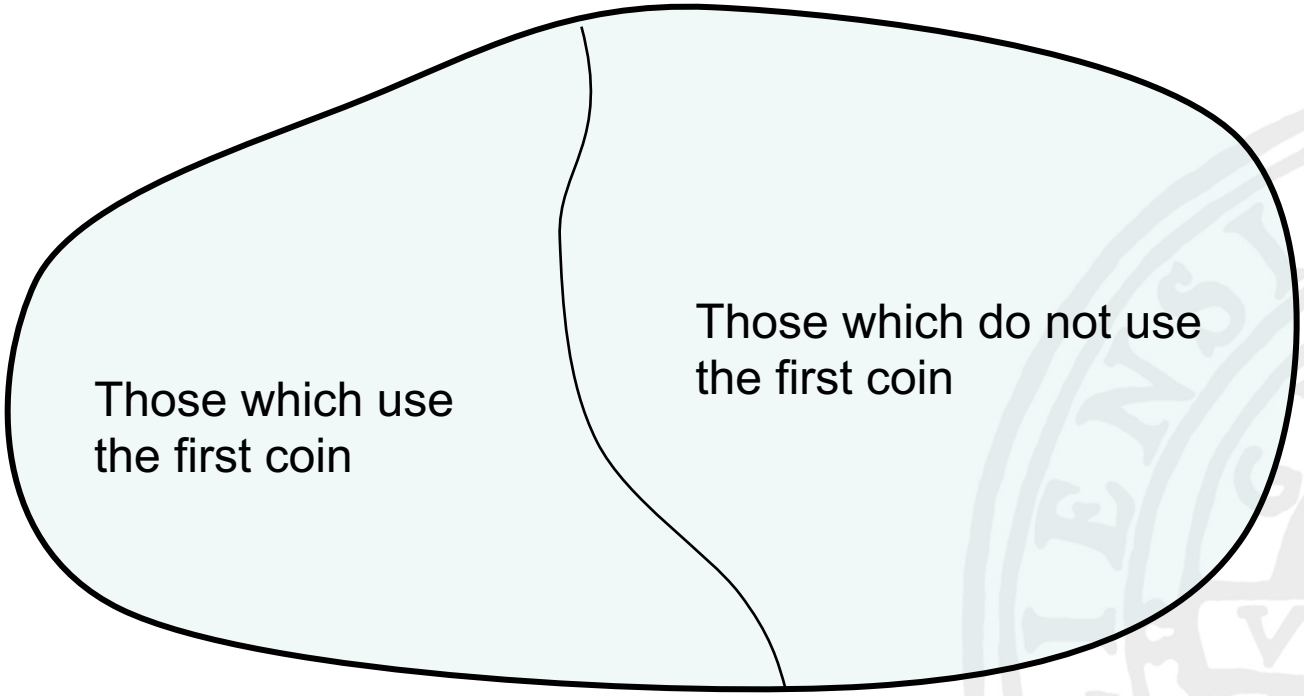
```
exchange(a, coins)
```

which returns the number of ways that one can exchange the amount `a`.
The second parameter contains the different values of coins.

The call `exchange(12, [1, 5, 10, 50, 100])` should return 4.

The exchange problem

The set of possible exchanges



UPPSALA UNIVERSITET

The exchange problem

Thus we get the following code:

```
def exchange(a, coins):
    return \
        exchange(a, coins[1:]) + \
        exchange(a-coins[0], coins)
```

Base case?

- a = 0 Success. Count 1
- a < 0 Unsuccessful. Count 0
- coins empty Unsuccessful. Count 0

UPPSALA UNIVERSITET

The exchange problem

Final version:

```
def exchange(a, coins):
    if a == 0:
        return 1
    elif a < 0 or len(coins) == 0:
        return 0
    else:
        return \
            exchange(a, coins[1:]) + \
            exchange(a-coins[0], coins)
```

Two challanges (optional):

1. Write a function that lists the exchanges made.
2. List possible exchanges given a limited number of each coin.

UPPSALA UNIVERSITET

The end

.