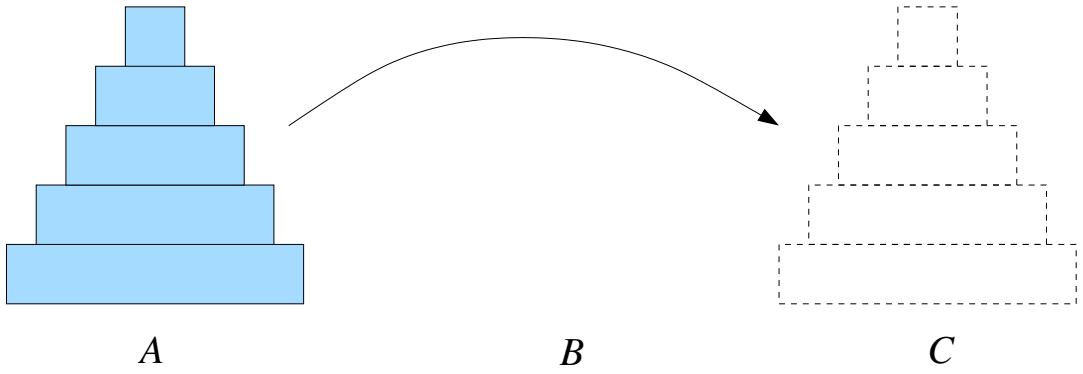
UPPSALA UNIVERSITET

MA1: L02

Tom Smedsaas

Rekursion: Två exempel.

Hanois torn



En trave brickor av olika storlek som ska flyttas från stället ***A*** till stället ***C*** med iakttagande av följande regler:

- Bara en bricka får flyttas i taget
- Man får aldrig lägga en större bricka på en mindre

För att detta ska vara möjligt behövs en ”hjälpplats” ***B***

UPPSALA UNIVERSITET

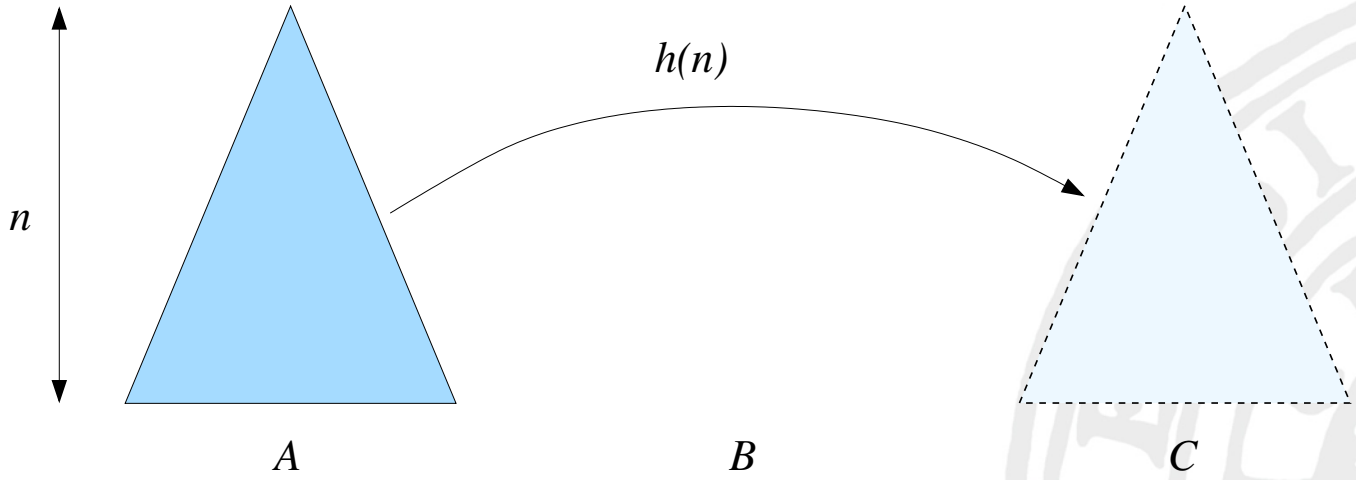
Visualisering

Tom flyttar brickor

Tháp Hà Nội

Beteckna problemet att flytta *n* brickor från ett ställe till ett annat med *h(n)*.

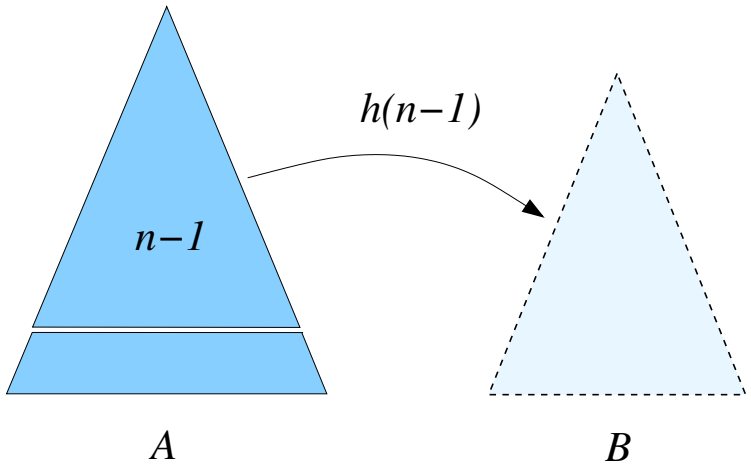
Vi vill alltså lösa problemet:



Hanois torn

För att kunna flytta den understa brickan från A till C måste alla andra brickor ligga på B.

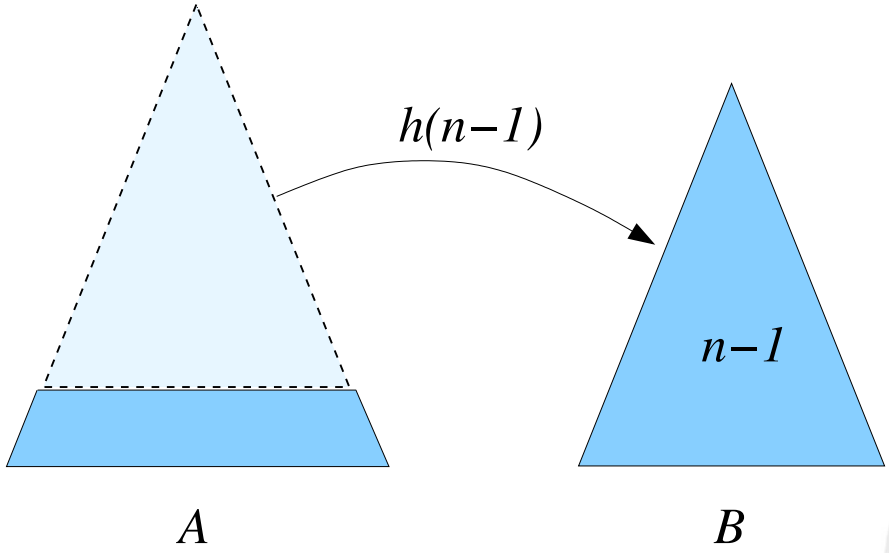
Vi måste alltså börja med att lösa problemet att flytta *n*-1 brickor från A till B:



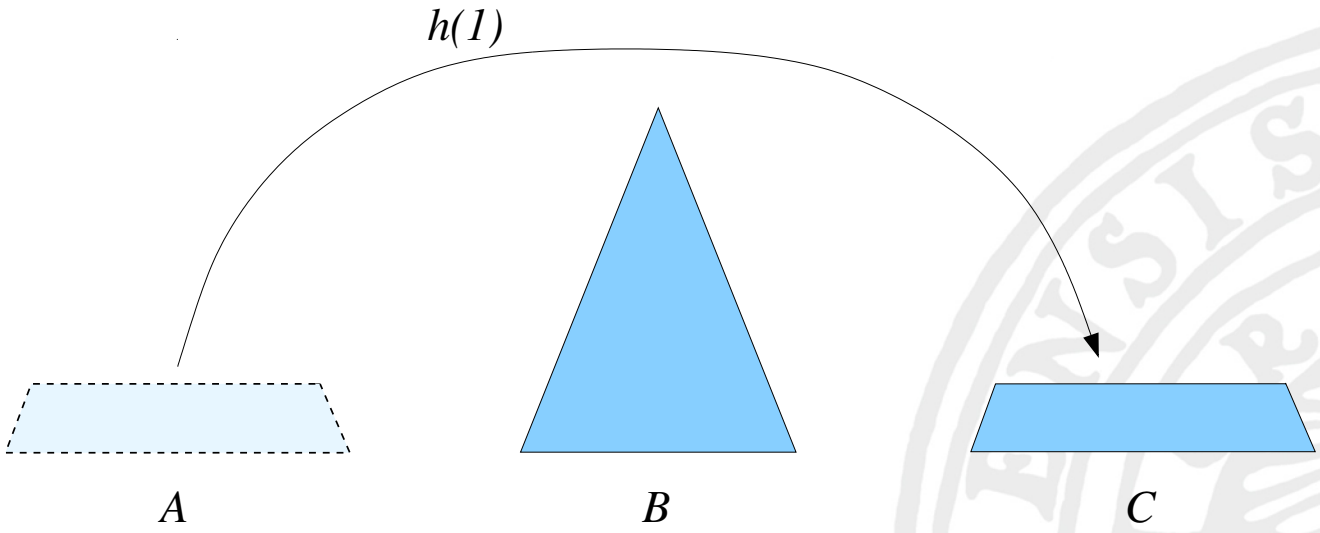
1. Flytta *n*-1 brickor från *A* till *B* med hjälp av *C*

.

Hanois torn



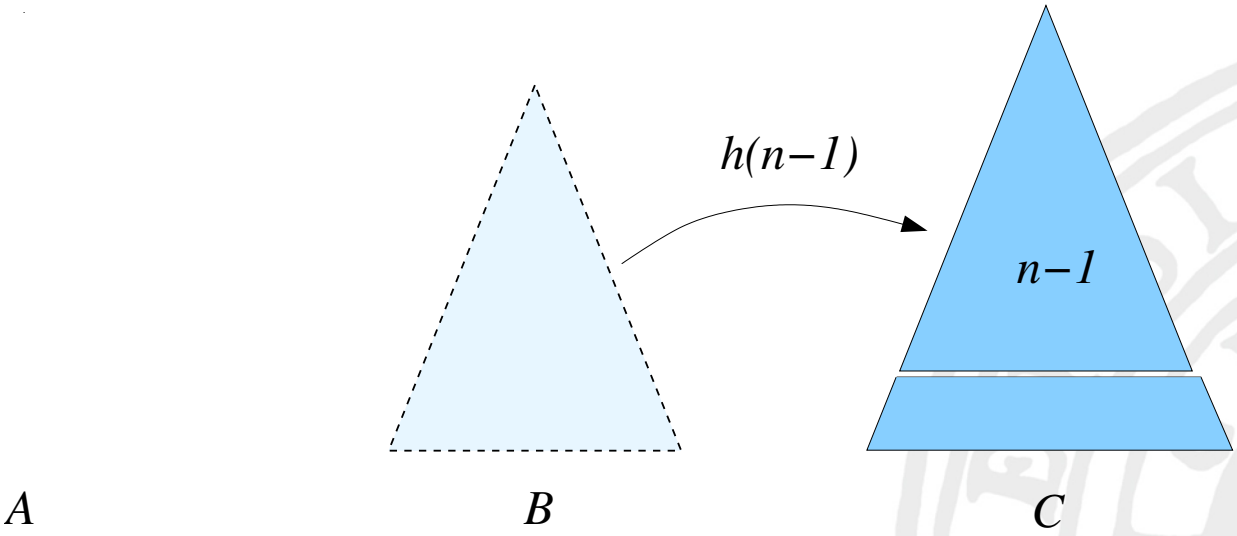
Hanois torn



2. Flytta 1 bricka från *A* till *C.*

.

Hanois torn



3. Flytta n – 1 brickor från *B* till *C* med hjälp av *A*

Hanois torn

Algoritm:

1. Flytta *n* - 1 brickor från *A* till *B* med hjälp av *C.*
2. Flytta 1 bricka från *A* till *C.*
3. Flytta n – 1 brickor från *B* till *C* med hjälp av *A.*

Observera att det är två rekursiva anrop samt att uppgifterna om från, till och med hjälp av varierar. Platserna *A*, *B* och *C* har byter alltså roller.

Tips 1: Enklast kod får man om man använder 0 som basfall.

Tips 2: Det räcker med 5 rader kod inklusive `def`, `if` och `else` för att lösa uppgiften!

UPPSALA UNIVERSITET

Växlingsproblemet

UPPSALA UNIVERSITET

Växlingsproblemet

Antag att vi har mynt/sedlar med följande värden: 1, 5, 10, 50, 100.
På hur många olika sätt kan vi växla ett givet belopp?

Exempel: Beloppet 12 kan växlas på 4 olika sätt:

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

Vi vill skriva en `funktion`

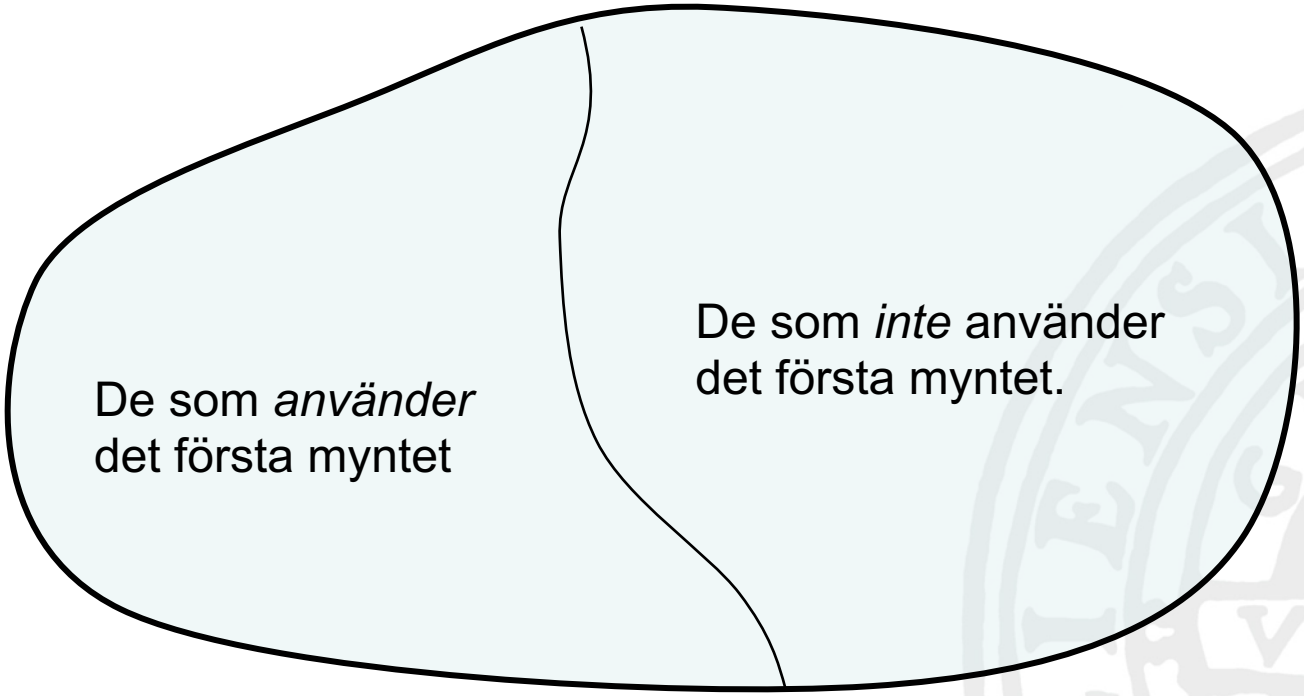
```
exchange(a, coins)
```

som returnerar antalet sätt som man kan växla beloppet a med de valörer som finns i `coins`.

Anropet `exchange(12, [1, 5, 10, 50, 100])` ska alltså returnera 4.

Växlingsproblemet

Mängden möjliga växlingar:



UPPSALA UNIVERSITET

Växlingsproblemet

Det ger följande kod:

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

Basfall?

- a = 0 Lyckad. Räkna 1
- a < 0 Misslyckad. Räkna 0
- coins tom Misslyckad. Räkna 0

UPPSALA UNIVERSITET

Växlingsproblemet

Slutlig 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)
```

Två utmatningar (frivilliga):

1. Skriv en funktion som listar de gjorda växlingarna.
2. Lista möjliga växlingar givet ett begränsat antal av varje mynt.

UPPSALA UNIVERSITET

The end

.