Recursivitat 2

ETSEIB/GIE

17 de novembre de 2023

Exercici: Factorial

- **Definició:**
  1. $n! = \prod_{k=1}^{n} k$
     o bé
     $n! = n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$
  2. $n! = \begin{cases} 1 & \text{si } n = 0, \\ n \cdot (n-1)! & \text{si } n > 0. \end{cases}$

- **Definició:**
  1. $n! = \prod_{k=1}^{n} k$
     o bé
     $n! = n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$
  2. $n! = \begin{cases} 1 & \text{si } n = 0, \\ n \cdot (n-1)! & \text{si } n > 0. \end{cases}$

Iteratiu: Factorial

```
def factorial(n):
    f = 1
    for i in range(1,n+1):
        f = f * i
    return f
```

Iteratiu: Factorial

```
def factorial(n):
  if n==0:
      return 1
  else:
      return n*factorial(n-1)
```

execució en pythontutor

Exercici: Potència

- **Definició:**
  1. $x^n = \underbrace{x \cdot x \cdot x \ldots x \cdot x \cdot x}_{\text{n cops}}$
  2. $x^n = \begin{cases} 1 & \text{si } n = 0, \\ x \cdot x^{n-1} & \text{si } n > 0. \end{cases}$

- **Definició:**
  1. $x^n = \underbrace{x \cdot x \cdot x \ldots x \cdot x \cdot x}_{\text{n cops}}$
  2. $x^n = \begin{cases} 1 & \text{si } n = 0, \\ x \cdot x^{n-1} & \text{si } n > 0. \end{cases}$

Iteratiu: Potència

```
def potencia(x, n):
    p=1
    for i in range(n):
        p=p*x
    return p
```

Recursiu: Potència

```
def potencia(x,n):
    if n==0:
        return 1
    else:
        return x*potencia(x,n-1)
```

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a, b) = mcd(b, a)$
  2. $mcd(a, b) = mcd(a + b, a) = mcd(a, b + a)$
  3. Si $a > b \to mcd(a, b) = mcd(a - b, b)$.
     Si $b > a \to mcd(a, b) = mcd(a, b - a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a, b) = mcd(b, a)$ ✓
  3. Si $a > b \to mcd(a, b) = mcd(a - b, b)$ ✓
  4. $mcd(a, a) = a$ ✓
- $mcd(a_1, b_1) = mcd(a_2, b_2) = \ldots mcd(a_n, b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a, b) = mcd(b, a)$
  2. $mcd(a, b) = mcd(a + b, a) = mcd(a, b + a)$
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$.
     Si $b > a \rightarrow mcd(a, b) = mcd(a, b - a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a, b) = mcd(b, a)$ ✓
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$ ✓
  4. $mcd(a, a) = a$ ✓
- $mcd(a_1, b_1) = mcd(a_2, b_2) = \ldots mcd(a_n, b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a, b) = mcd(b, a)$
  2. $mcd(a, b) = mcd(a + b, a) = mcd(a, b + a)$
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$.
     Si $b > a \rightarrow mcd(a, b) = mcd(a, b - a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a, b) = mcd(b, a)$ ✓
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$ ✓
  4. $mcd(a, a) = a$ ✓
- $mcd(a_1, b_1) = mcd(a_2, b_2) = \ldots mcd(a_n, b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a, b) = mcd(b, a)$
  2. $mcd(a, b) = mcd(a + b, a) = mcd(a, b + a)$
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$.
     Si $b > a \rightarrow mcd(a, b) = mcd(a, b - a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a, b) = mcd(b, a)$ ✓
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$ ✓
  4. $mcd(a, a) = a$ ✓
- $mcd(a_1, b_1) = mcd(a_2, b_2) = \ldots mcd(a_n, b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a, b) = mcd(b, a)$
  2. $mcd(a, b) = mcd(a + b, a) = mcd(a, b + a)$
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$.
     Si $b > a \rightarrow mcd(a, b) = mcd(a, b - a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a, b) = mcd(b, a)$ ✓
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$ ✓
  4. $mcd(a, a) = a$ ✓
- $mcd(a_1, b_1) = mcd(a_2, b_2) = \ldots mcd(a_n, b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a, b) = mcd(b, a)$
  2. $mcd(a, b) = mcd(a + b, a) = mcd(a, b + a)$
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$.
     Si $b > a \rightarrow mcd(a, b) = mcd(a, b - a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a, b) = mcd(b, a)$ ✓
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$ ✓
  4. $mcd(a, a) = a$ ✓
- $mcd(a_1, b_1) = mcd(a_2, b_2) = \ldots mcd(a_n, b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a,b) = mcd(b,a)$
  2. $mcd(a,b) = mcd(a+b,a) = mcd(a,b+a)$
  3. Si $a > b \rightarrow mcd(a,b) = mcd(a-b,b)$.
     Si $b > a \rightarrow mcd(a,b) = mcd(a,b-a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a,b) = mcd(b,a)$ ✓
  3. Si $a > b \rightarrow mcd(a,b) = mcd(a-b,b)$ ✓
  4. $mcd(a,a) = a$ ✓
- $mcd(a_1,b_1) = mcd(a_2,b_2) = \ldots mcd(a_n,b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

- **Objectiu:** Generar parelles de números que comparteixen el mateix màxim comú divisor
  1. $mcd(a, b) = mcd(b, a)$
  2. $mcd(a, b) = mcd(a + b, a) = mcd(a, b + a)$
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$.
     Si $b > a \rightarrow mcd(a, b) = mcd(a, b - a)$. Per propietat 1
- Forçar que s'arribi a una parella de valors iguals. D'aquesta manera sabrem el màxim comú divisor.
  1. $mcd(a, b) = mcd(b, a)$ ✓
  3. Si $a > b \rightarrow mcd(a, b) = mcd(a - b, b)$ ✓
  4. $mcd(a, a) = a$ ✓
- $mcd(a_1, b_1) = mcd(a_2, b_2) = \ldots mcd(a_n, b_n)$
  i per $a > 0, b > 0$ tenim
  $a_1 + b_1 > a_2 + b_2 > \ldots a_n + b_n > 1$

Iteratiu: Màxim Comú Divisor

```
def mcd(a, b):
    while a != b:
        if a > b:
            a = a - b
        else:   # b>a
            b = b - a
    return a
```

Recursivitat: Màxim Comú Divisor

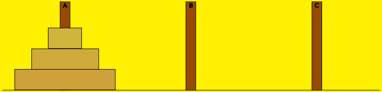
```
def mcd(a, b):
    if a > b:
        return mcd(a-b, b)
    elif a < b:
        return mcd(a, b-a)
    else:  # a==b
        return a
```

Esquema habitual de Recursivitat

```
if esCasBase:
    calculaCasBaseSenseRecursivitat()
else:
    trencaElProblemaEnSubproblemesMesPetitsIgualsEnForm
    SolucionaCadaSubproblemaCridantAquestaFuncioRecursi
    APartirDeLesSolucionsDelsSubproblemesObtenimSolucio
```

Torres d'Hanoi

Es tenen tres pals clavats al terra que anomenarem A, B, i C respectivament, i n discs, tots ells de diametres diferents. Els discs tenen un forat de manera que es poden posar en els pals, formant torres. A l'inici, tots els discs estan en un pal (per exemple pal A) ordenats pel seu diàmetre de manera que el disc d'amunt té el diàmetre inferior i el que està més avall té el diàmetre més gran.



Es tenen tres pals clavats al terra que anomenarem A, B, i C respectivament, i n discs, tots ells de diametres diferents. Els discs tenen un forat de manera que es poden posar en els pals, formant torres. A l'inici, tots els discs estan en un pal (per exemple pal A) ordenats pel seu diàmetre de manera que el disc d'amunt té el diàmetre inferior i el que està més avall té el diàmetre més gran. Per passar un disc d'un pal a un altre s'han de tenir en compte les següents restriccions:

- *En cada pas només es pot moure un disc d'un pal a un altre.*
- *El diàmetre d'un disc en un pal ha de ser sempre inferior al diàmetre del disc que té a sota.*

Dissenyar la funció mouDiscos(nDiscos, origen, desti, suport) que escriu al canal estàndard de sortida tots els moviments de discs que calen per passar els discs inicials en el pal origen al pal desti seguint les regles anteriors

exemple

```
>>> mouDiscos(3,'A','B','C')
mou disc del pal A al pal B
mou disc del pal A al pal C
mou disc del pal B al pal C
mou disc del pal A al pal B
mou disc del pal C al pal A
mou disc del pal C al pal B
mou disc del pal A al pal B
```

Dissenyar la funció mouDiscos(nDiscos, origen, desti, suport) que escriu al canal estàndard de sortida tots els moviments de discs que calen per passar els discs inicials en el pal origen al pal desti seguint les regles anteriors

exemple

```
>>> mouDiscos(3,'A','B','C')
mou disc del pal A al pal B
mou disc del pal A al pal C
mou disc del pal B al pal C
mou disc del pal A al pal B
mou disc del pal C al pal A
mou disc del pal C al pal B
mou disc del pal A al pal B
```

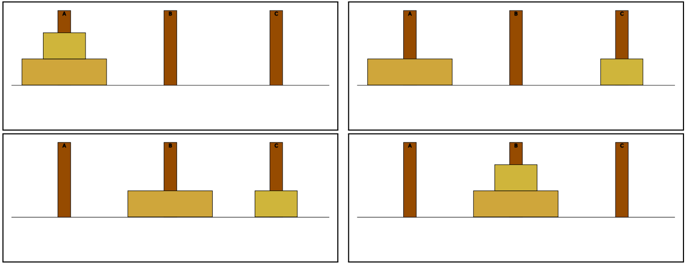
Dissenyar la funció mouDiscos(nDiscos, origen, desti, suport) que escriu al canal estàndard de sortida tots els moviments de discs que calen per passar els discs inicials en el pal origen al pal desti seguint les regles anteriors

exemple

```
>>> mouDiscos(3,'A','B','C')
mou disc del pal A al pal B
mou disc del pal A al pal C
mou disc del pal B al pal C
mou disc del pal A al pal B
mou disc del pal C al pal A
mou disc del pal C al pal B
mou disc del pal A al pal B
```

Torres d'Hanoi: 2 discs

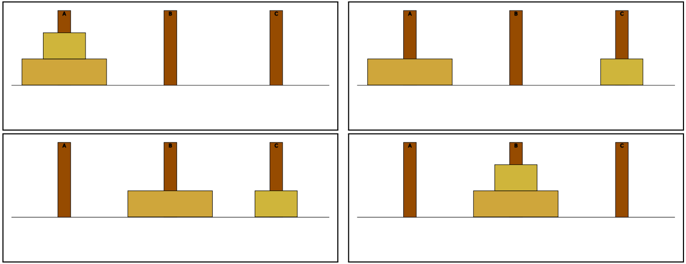
mou 2 discos del pal A cap al pal B usant el pal C
mou 1 disc del pal A cap al pal C
mou 1 disc del pal A cap al pal B
mou 1 disc del pal C cap al pal B



Torres d'Hanoi: 2 discs

$$(2,\ A \rightarrow B)$$

$$(1,\ A \rightarrow C)$$
$$(1,\ A \rightarrow B)$$
$$(1,\ C \rightarrow B)$$



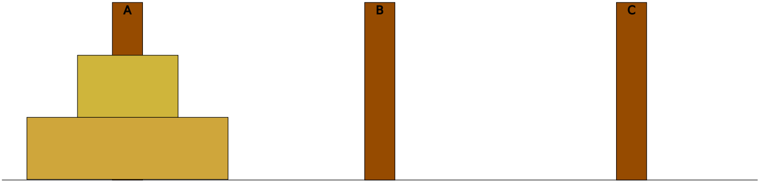
Torres d'Hanoi: 2 discs

$(2, A \rightarrow B)$

$(1, A \rightarrow C)$

$(1, A \rightarrow B)$

$(1, C \rightarrow B)$



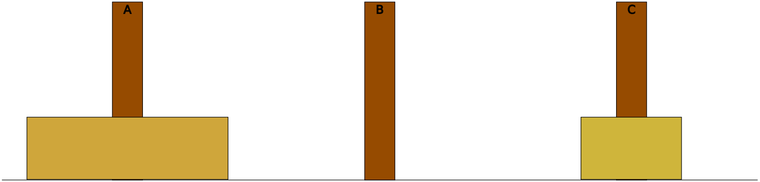
Torres d'Hanoi: 2 discs

$(2, A \rightarrow B)$

$(1, A \rightarrow C)$

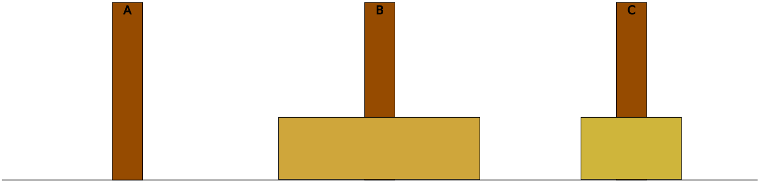
$(1, A \rightarrow B)$

$(1, C \rightarrow B)$



Torres d'Hanoi: 2 discs

(2, A $\to$ B)
(1, A $\to$ C)
(1, A $\to$ B)
(1, C $\to$ B)



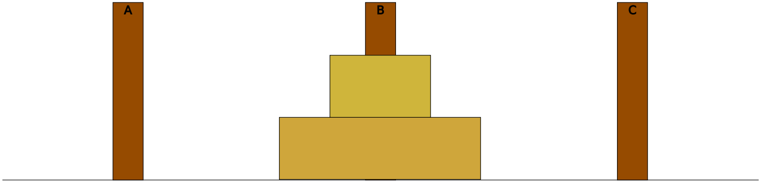
Torres d'Hanoi: 2 discs

$(2,\ A \rightarrow B)$

$(1,\ A \rightarrow C)$

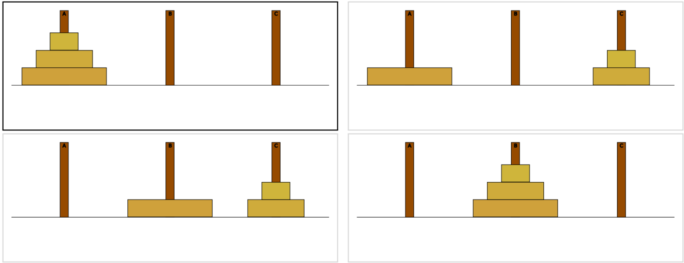
$(1,\ A \rightarrow B)$

$(1,\ C \rightarrow B)$



Torres d'Hanoi: 3 discs

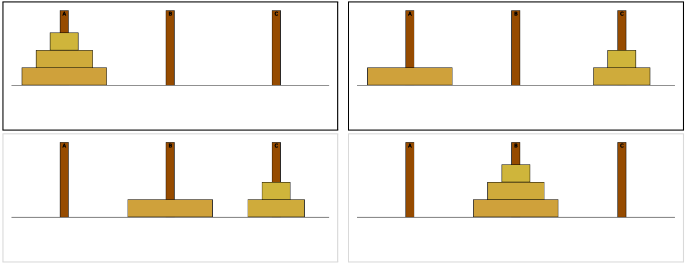
mou 3 discos del pal A cap al pal B usant el pal C
mou 2 discos del pal A cap al pal C usant el pal B
mou 1 disc del pal A cap al pal B
mou 2 discos del pal C cap al pal B usant el pal A



Torres d'Hanoi: 3 discs

$$(3,\ A \rightarrow B)$$

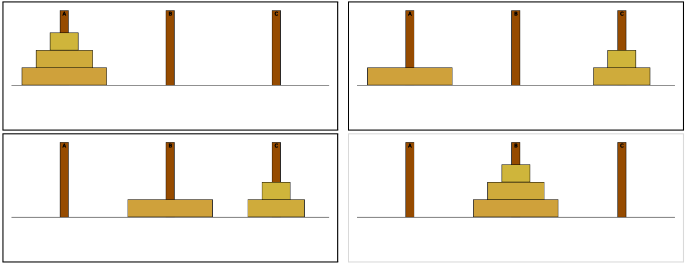
$$(2,\ A \rightarrow C)$$
$$(1,\ A \rightarrow B)$$
$$(2,\ C \rightarrow B)$$



Torres d'Hanoi: 3 discs

$(3,\ A \rightarrow B)$

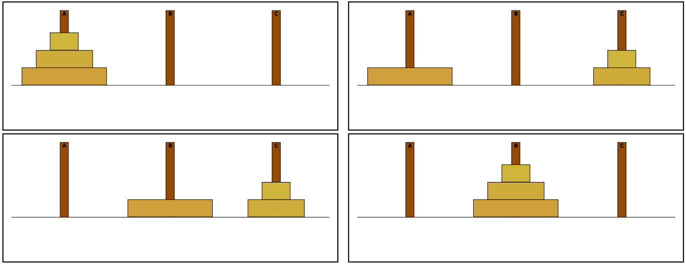
$(2,\ A \rightarrow C)$
$(1,\ A \rightarrow B)$
$(2,\ C \rightarrow B)$



Torres d'Hanoi: 3 discs

$$(3,\ A \rightarrow B)$$

$$(2,\ A \rightarrow C)$$
$$(1,\ A \rightarrow B)$$
$$(2,\ C \rightarrow B)$$



Torres d'Hanoi: 3 discs

(3, A $\rightarrow$ B)

  (2, A $\rightarrow$ C)

    (1, A $\rightarrow$ B)

    (1, A $\rightarrow$ C)

    (1, B $\rightarrow$ C)

  (1, A $\rightarrow$ B)

  (2, C $\rightarrow$ B)

    (1, C $\rightarrow$ A)

    (1, C $\rightarrow$ B)

    (1, A $\rightarrow$ B)

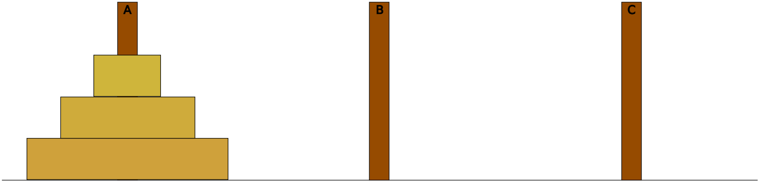
Torres d'Hanoi: 3 discs

$(3,\ A \to B)$

$(2,\ A \to C)$

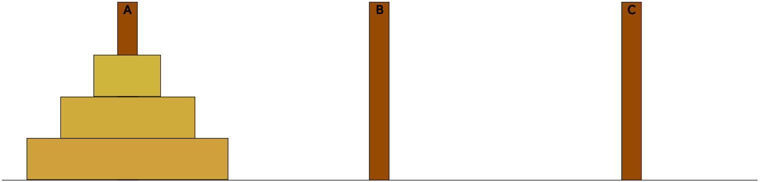
$(1,\ A \to B)$

$(2,\ C \to B)$



Torres d'Hanoi: 3 discs

(3, A → B)
(2, A → C)
(1, A → B)
(1, A → C)
(1, B → C)
(1, A → B)
(2, C → B)



Torres d'Hanoi: 3 discs

$(3, A \rightarrow B)$

$(2, A \rightarrow C)$

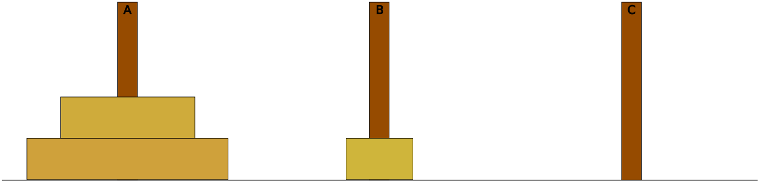
$(1, A \rightarrow B)$

$(1, A \rightarrow C)$

$(1, B \rightarrow C)$

$(1, A \rightarrow B)$

$(2, C \rightarrow B)$



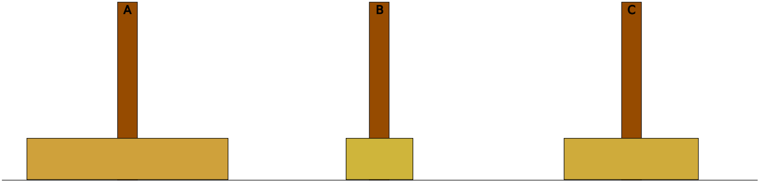
Torres d'Hanoi: 3 discs

$(3,\ A \rightarrow B)$

$(2,\ A \rightarrow C)$

$(1,\ A \rightarrow B)$
$(1,\ A \rightarrow C)$
$(1,\ B \rightarrow C)$

$(1,\ A \rightarrow B)$
$(2,\ C \rightarrow B)$



Torres d'Hanoi: 3 discs

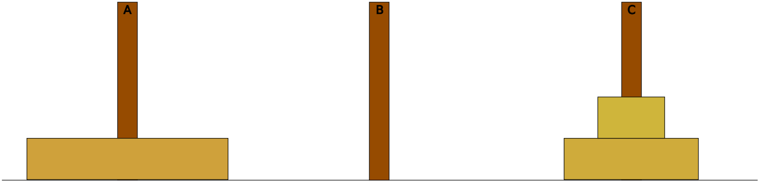
(3, A → B)

(2, A → C)

(1, A → B)
(1, A → C)
(1, B → C)

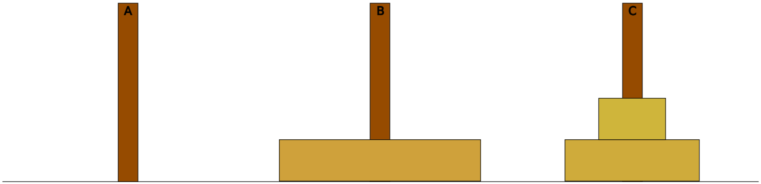
(1, A → B)

(2, C → B)



Torres d'Hanoi: 3 discs

(3, A → B)
(2, A → C)
(1, A → B)
(2, C → B)
(1, C → A)
(1, C → B)
(1, A → B)

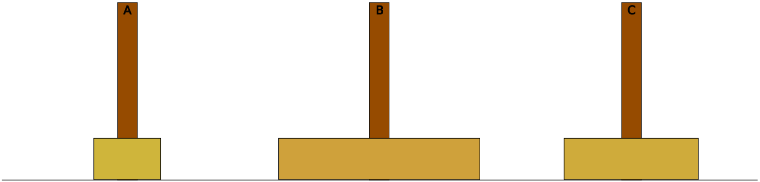


Torres d'Hanoi: 3 discs

$(3,\ A \to B)$

$(2,\ A \to C)$
$(1,\ A \to B)$
$(2,\ C \to B)$

$(1,\ C \to A)$
$(1,\ C \to B)$
$(1,\ A \to B)$



Torres d'Hanoi: 3 discs

$(3,\ A \rightarrow B)$

$(2,\ A \rightarrow C)$

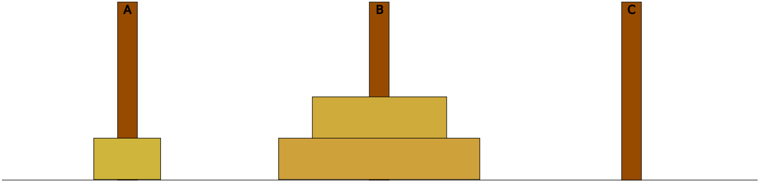
$(1,\ A \rightarrow B)$

$(2,\ C \rightarrow B)$

$(1,\ C \rightarrow A)$

$(1,\ C \rightarrow B)$

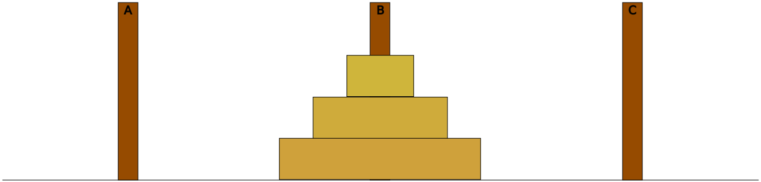
$(1,\ A \rightarrow B)$



Torres d'Hanoi: 3 discs

$(3, A \to B)$

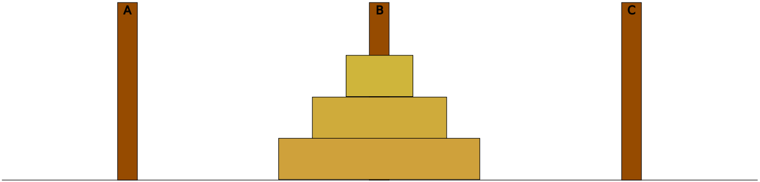
$(2, A \to C)$
$(1, A \to B)$
$(2, C \to B)$



Torres d'Hanoi: 3 discs

$(3,\ \mathrm{A} \rightarrow \mathrm{B})$

$(2,\ \mathrm{A} \rightarrow \mathrm{C})$
$(1,\ \mathrm{A} \rightarrow \mathrm{B})$
$(2,\ \mathrm{C} \rightarrow \mathrm{B})$



```
def mouDisc(o, d):
    print('mou disc del pal {} al pal {}'.format(o, d))
```

```
def mouDiscos(nDiscos, origen, desti, suport):
    if nDiscos != 0:
        mouDiscos(nDiscos-1,
                  origen, suport, desti)
        mouDisc(origen, desti)
        mouDiscos(nDiscos-1,
                  suport, desti, origen)
```

Torres d'Hanoi: Versió iterativa

2	3	4	5	
$A \rightarrow C$	$A \rightarrow B$	$A \rightarrow C$	$A \rightarrow B$	$A \rightarrow B$
$A \rightarrow B$	$A \rightarrow C$	$A \rightarrow B$	$A \rightarrow C$	$C \rightarrow A$
$C \rightarrow B$	$B \rightarrow C$	$C \rightarrow B$	$B \rightarrow C$	$C \rightarrow B$
	$A \rightarrow B$	$A \rightarrow C$	$A \rightarrow B$	$A \rightarrow B$
	$C \rightarrow A$	$B \rightarrow A$	$C \rightarrow A$	$C \rightarrow A$
	$C \rightarrow B$	$B \rightarrow C$	$C \rightarrow B$	$B \rightarrow C$
	$A \rightarrow B$	$A \rightarrow C$	$A \rightarrow B$	$B \rightarrow A$
		$A \rightarrow B$	$A \rightarrow C$	$C \rightarrow A$
		$C \rightarrow B$	$B \rightarrow C$	$C \rightarrow B$
		$C \rightarrow A$	$B \rightarrow A$	$A \rightarrow B$
		$B \rightarrow A$	$C \rightarrow A$	$A \rightarrow C$
		$C \rightarrow B$	$B \rightarrow C$	$B \rightarrow C$
		$A \rightarrow C$	$A \rightarrow B$	$A \rightarrow B$
		$A \rightarrow B$	$A \rightarrow C$	$C \rightarrow A$
		$C \rightarrow B$	$B \rightarrow C$	$C \rightarrow B$
				$A \rightarrow B$

Torres d'Hanoi: Versió iterativa

- Per moure n discos calen $2^n - 1$ moviments.
- Si n és parell, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ C, A $\leftrightarrow$ B, C $\leftrightarrow$ B
- Si n és senar, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ B, A $\leftrightarrow$ C, C $\leftrightarrow$ B
- El sentit és en la direcció de disc petit sobre disc gran

Torres d'Hanoi: Versió iterativa

- Per moure n discos calen $2^n - 1$ moviments.
- Si n és parell, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ C, A $\leftrightarrow$ B, C $\leftrightarrow$ B
- Si n és senar, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ B, A $\leftrightarrow$ C, C $\leftrightarrow$ B
- El sentit és en la direcció de disc petit sobre disc gran

- Per moure n discos calen $2^n - 1$ moviments.
- Si n és parell, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ C, A $\leftrightarrow$ B, C $\leftrightarrow$ B
- Si n és senar, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ B, A $\leftrightarrow$ C, C $\leftrightarrow$ B
- El sentit és en la direcció de disc petit sobre disc gran

Torres d'Hanoi: Versió iterativa

- Per moure n discos calen $2^n - 1$ moviments.
- Si n és parell, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ C, A $\leftrightarrow$ B, C $\leftrightarrow$ B
- Si n és senar, hi ha una seqüència repetitiva entre pals: A $\leftrightarrow$ B, A $\leftrightarrow$ C, C $\leftrightarrow$ B
- El sentit és en la direcció de disc petit sobre disc gran

Torres d'Hanoi: Versió iterativa


```
def mouDiscos(nDiscos, origen, desti, suport):
    if nDiscos % 2 != 0:
        for i in range(1, 2**nDiscos):
            if (i % 3 == 1):
                mouDiscEntre(origen, desti)
            elif (i % 3 == 2):
                mouDiscEntre(origen, suport)
            else: #  (i % 3 == 0):
                mouDiscEntre(desti, suport)
    else:
        for i in range(1, 2**nDiscos):
            if (i % 3 == 1):
                mouDiscEntre(origen, suport)
            elif (i % 3 == 2):
                mouDiscEntre(origen, desti)
            else: # (i % 3 == 0):
                mouDiscEntre(desti, suport)
```

Torres d'Hanoi: Versió iterativa

```
def mouDiscEntre(a, b):
    assert not a.buit() or not b.buit()
    if a.buit():
        o, d = b, a
    elif b.buit():
        o, d = a, b
    else: # not a.buit() and not b.buit()
        if a.cim() < b.cim():
            o, d = a, b
        else:
            o, d = b, a
    d.empila(o.desempila())
    mouDisc(o, d)
```

Torres d'Hanoi: Versió iterativa


```
class Pal:
    def __init__(self, nom, nd):
        self.nom = nom
        self.pila= [n for n in range(nd, 0, -1)]

    def cim(self):
        return self.pila[-1]

    def empila(self, disc):
        self.pila.append(disc)

    def desempila(self):
        return self.pila.pop()

    def buit(self):
        return len(self.pila)==0
```