Estructura de Datos y Algoritmos

Recursividad

Agenda

- Estructura de datos
- Listas
- Tipos de Listas
- Operaciones con Listas

Recursividad

Técnica de programación que nos permite que un bloque de instrucciones se ejecute un cierto número de veces en función de ella misma.

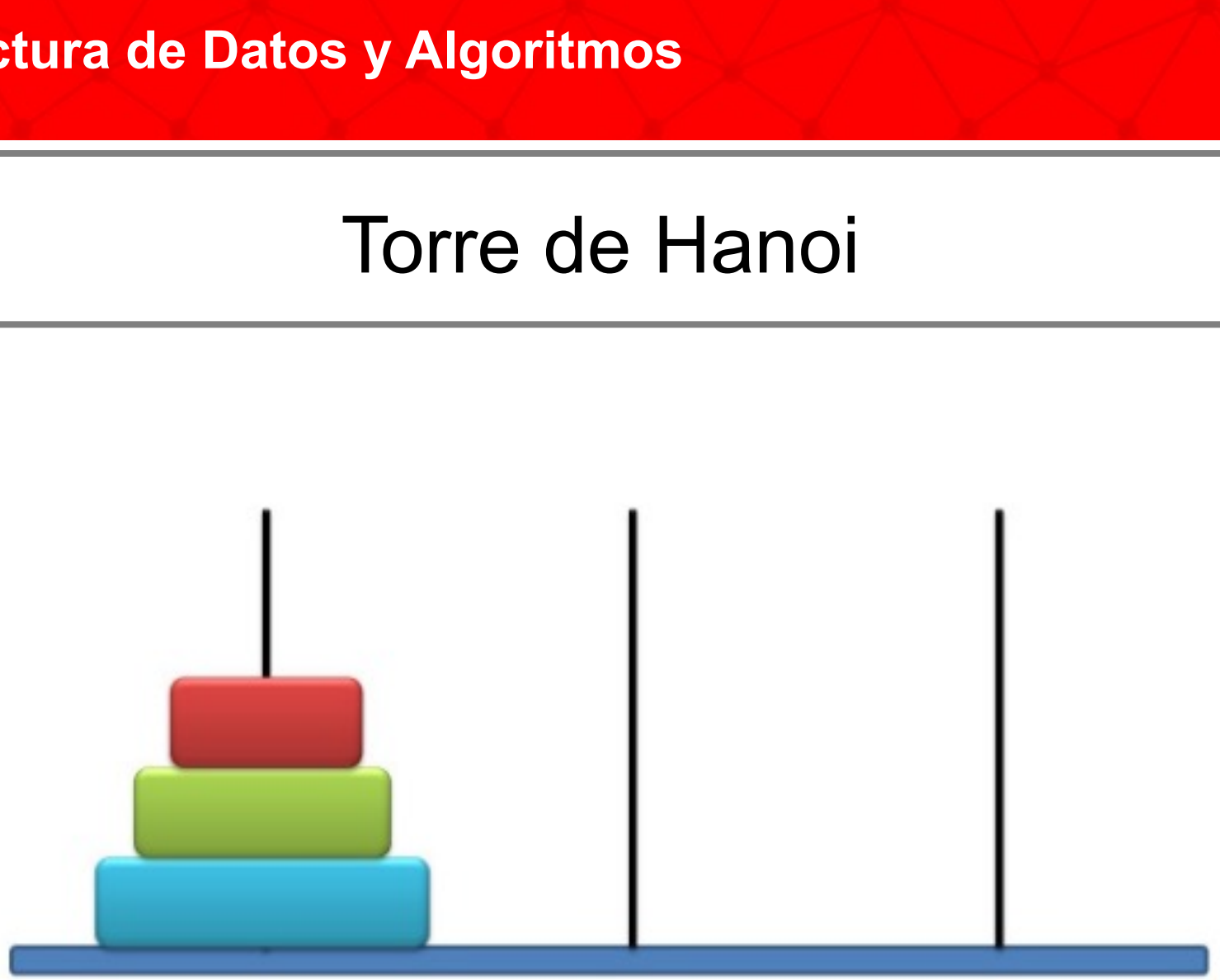
Es una **alternativa** a las estructuras repetitivas.

No todas las funciones pueden llamarse a sí misma.

Recursividad

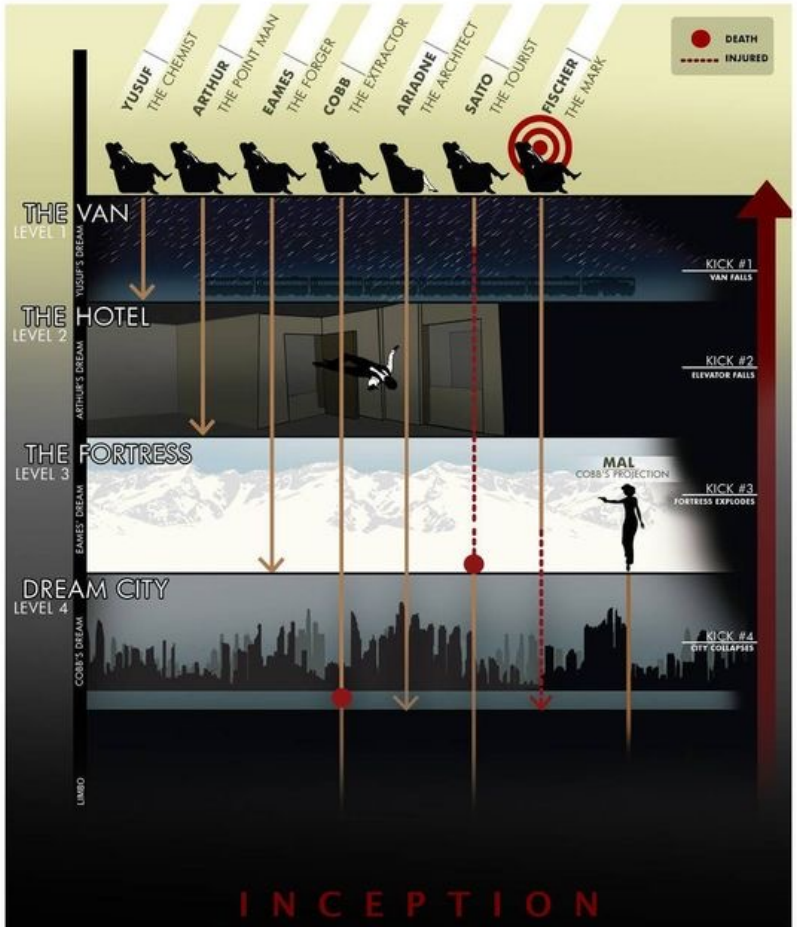
Condiciones

- El cálculo de la solución se obtiene invocándose a sí misma.
- Se debe dividir la tarea en subtareas similares a la tarea principal pero de menor tamaño (caso base).



Torre de Hanoi

Inception (El Origen)



Inception (El Origen)

Tipos

- Directa
  - Cuando una subrutina se llama a sí misma
- Indirecta
  - Cuando se tienen varias subrutinas y estas se llaman unas a otras formando ciclos.

Factorial

```ruby
def programaPrincipal
    #factorial
    n = 5

    numero = 1
    while n > 1
        numero = numero * n
        n = n - 1
    end

    puts numero

end
programaPrincipal

```

Factorial

```ruby
def factorial(numero)
    if numero < 0
        return 0
    elsif numero == 1
        return 1
    else
        return numero * factorial(numero-1)
    end
end

def programaPrincipal
    #factorial
    n = 5

    numero = factorial(n)

    puts numero
end
programaPrincipal

```

Suma de n primeros números

```
1   def programaPrincipal
2       n = 5
3
4       #primeros numeros enteros
5       suma = 0
6       while n>0
7           suma = suma + n
8           n = n-1
9       end
10      puts suma
11
12  end
13  programaPrincipal
14
```

Suma de n primeros números

```ruby
def par(numero)
    return numero + impar(numero-1)
end

def impar(numero)
    if numero == 1
        return 1
    else
        return numero + par(numero-1)
    end
end

def suma(numero)
    if numero <= 0
        return 0
    else
        if numero % 2 == 0 #par
            return numero + impar(numero-1)
        else #impar
            return numero + par(numero-1)
        end
    end
end

def programaPrincipal
    n = 5
    numero = suma(n)
    puts numero
end
programaPrincipal
```

Recursividad

- Consideraciones
  - Definir correctamente el algoritmo de la repetición y luego el de la recursión.
  - Primero definir el caso base, luego los casos contrarios al caso base.
  - Verificar si es necesario validar los casos que no deben entrar a la recursión.
  - Algunos algoritmos recursivos son ineficientes por el uso de la memoria.

Recursividad

- Ejercicios
  - Desarrollar un programa recursivo que permita invertir un número.
  - Desarrollar un programa recursivo que permita calcular el máximo común divisor de dos números.
  - Desarrollar un programa recursivo que permita calcule el valor de un número en la serie fibonacci

Gracias…