ALGORITMIA

PRÁCTICAS – SESIÓN 2.1

ESTRUCTURA GENERAL DE UNA FUNCIÓN RECURSIVA

El siguiente esquema describe el MODELO GENERAL en el que deben encuadrarse las funciones recursivas:

{ Q($\bar{x}$) }
función f($\bar{x}$: T_1) retorna ($\bar{y}$: T_2)
  caso
    B_t ($\bar{x}$) $\rightarrow$ triv($\bar{x}$)
    B_{nt}($\bar{x}$) $\rightarrow$ c(f(s($\bar{x}$)), $\bar{x}$)
  fcaso
ffunción
{ R($\bar{x}$,$\bar{y}$) }

donde los parámetros formales $\bar{x}$ e $\bar{y}$ han de entenderse como tuplas { x_1, x_2, ..., x_n } e {y_1, y_2, ..., y_m }, respectivamente

TIPOS DE FUNCIONES RECURSIVAS

Según que el **elemento sucesor s($\bar{x}$) sea único o no**, pueden presentarse los dos tipos de funciones recursivas siguientes:

- **Función recursiva lineal o simple**: cuando la función recursiva genera A LO SUMO UNA LLAMADA INTERNA por cada llamada externa.

- **Función recursiva no lineal o múltiple**: cuando genera DOS O MÁS LLAMADAS INTERNAS por cada llamada externa.

Nota.- Si aparecieran varias llamadas recursivas, cada una en una alternativa diferente de una instrucción condicional, la recursividad seguiría siendo lineal ya que, en tiempo de ejecución, las alternativas son mutuamente excluyentes y a lo sumo se produciría una invocación.

EJEMPLO DE FUNCIÓN RECURSIVA LINEAL O SIMPLE

$\{ a \geq 0 \wedge n \geq 0 \}$

Funcion POTENCIA (a, n : entero) retorna (p : entero)

caso

$n = 0 \rightarrow 1$

$n > 0 \rightarrow$ POTENCIA(a, n-1) * a

fcaso

ffunción

$\{ p = a^n \}$

LINEAL O SIMPLE: DESCENSO CADENA DE LLAMADAS RECURSIVAS

$\{ a \geq 0 \land n \geq 0 \}$

Funcion POTENCIA (a, n : entero) retorna (p : entero)

caso

$n = 0 \rightarrow 1$

$n > 0 \rightarrow$ POTENCIA(a, n-1) * a

fcaso

ffunción

$\{ p = a^n \}$

s(a,n) = (a,n-1)

$$\boxed{\begin{matrix} a=2 \\ n=3 \end{matrix}} \xrightarrow{n=n-1} \boxed{\begin{matrix} a=2 \\ n=2 \end{matrix}} \xrightarrow{n=n-1} \boxed{\begin{matrix} a=2 \\ n=1 \end{matrix}} \xrightarrow{n=n-1} \boxed{\begin{matrix} a=2 \\ n=0 \end{matrix}}$$

EJEMPLO DE FUNCIÓN RECURSIVA NO LINEAL O MÚLTIPLE

$\{ n \geq 0 \}$

Funcion POTENCIA3 (n : entero) retorna (f : entero)

caso

$n = 0 \rightarrow 1$

$n = 1 \rightarrow 3$

$n > 1 \rightarrow 2 * \text{POTENCIA3}(n-1) + 3 * \text{POTENCIA3}(n-2)$

fcaso

ffunción

$\{ f = 3^n \}$

NO LINEAL O MÚLTIPLE: DESCENSO CADENA DE LLAMADAS RECURSIVAS

$\{ n \geq 0 \}$

Funcion POTENCIA3 (n : entero) retorna (f : entero)

caso

$n = 0 \rightarrow 1$

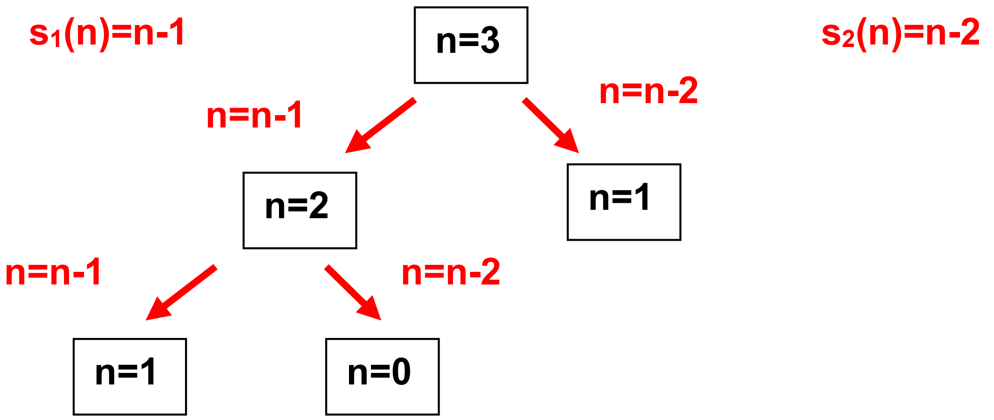
$n = 1 \rightarrow 3$

$n > 1 \rightarrow 2 * \text{POTENCIA3}(n\text{-}1) + 3 * \text{POTENCIA3}(n\text{-}2)$

fcaso

ffunción

$\{ f = 3^n \}$



EJEMPLO DE FUNCIÓN RECURSIVA LINEAL O SIMPLE

{ a > 0 ∧ b > 0 }

```
Funcion MCD (a, b : entero) retorna (g : entero)
   caso
      a = b → a
      a > b → MCD(a-b, b)
      a < b → MCD(a, b-a)
   fcaso
ffunción
```

{ g = mcd(a,b) }

LINEAL O SIMPLE: DESCENSO CADENA DE LLAMADAS RECURSIVAS

$\{\, a > 0 \wedge b > 0 \,\}$

Funcion MCD (a, b : entero) retorna (g : entero)

caso

$a = b \rightarrow a$

$a > b \rightarrow$ MCD(a-b, b)

$a < b \rightarrow$ MCD(a, b-a)

fcaso

ffunción

$\{\, g = mcd(a,b) \,\}$

s(a,b)=si a > b entonces a = a - b sino b = b - a fsi

$$\begin{matrix} a=48 \\ b=60 \end{matrix} \xrightarrow{b=b-a} \begin{matrix} a=48 \\ b=12 \end{matrix} \xrightarrow{a=a-b} \begin{matrix} a=36 \\ b=12 \end{matrix} \xrightarrow{a=a-b} \begin{matrix} a=24 \\ b=12 \end{matrix} \xrightarrow{a=a-b} \begin{matrix} a=12 \\ b=12 \end{matrix}$$

TIPOS DE FUNCIONES RECURSIVAS

Según que el comportamiento de la **función c** (función de combinación) pueden presentarse los dos tipos de funciones recursivas siguientes:

- **Función recursiva no final o no de cola**: cuando la función c es necesaria, esto es,

$$f(\bar{x}) = c(f(s(\bar{x})), \bar{x})$$

- **Función recursiva final o de cola**: cuando la función c no es necesaria, esto es,

$$f(\bar{x}) = f(s(\bar{x}))$$

FUNCIÓN RECURSIVA NO FINAL O NO DE COLA

Cuando la función c es necesaria, esto es, $f(\bar{x}) = c(f(s(\bar{x})), \bar{x})$

$\{ Q(\bar{x}) \}$

función $f(\bar{x}$: T1) retorna $(\bar{y}$: T2)

  caso

    $Bt(\bar{x}) \rightarrow triv(\bar{x})$

    $Bnt(\bar{x}) \rightarrow c(f(s(\bar{x})), \bar{x})$

  fcaso

ffunción

$\{ R(\bar{x},\bar{y}) \}$

FUNCIÓN RECURSIVA FINAL O DE COLA

Cuando la función c no es necesaria, esto es, $f(\bar{x}) = f(s(\bar{x}))$

{ $Q(\bar{x})$ }

función $f(\bar{x}$: T1) retorna ($\bar{y}$: T2)

caso

Bt ($\bar{x}$) $\rightarrow$ triv($\bar{x}$)

Bnt($\bar{x}$) $\rightarrow$ $f(s(\bar{x}))$

fcaso

ffunción

{ $R(\bar{x},\bar{y})$ }

EJEMPLO DE FUNCIÓN RECURSIVA NO FINAL O NO DE COLA

{ $a \geq 0 \wedge n \geq 0$ }

Funcion POTENCIA (a, n : entero) retorna (p : entero)

caso

$n = 0 \rightarrow 1$

$n > 0 \rightarrow$ POTENCIA(a,n-1) * a

fcaso

ffunción

{ $p = a^n$ }

NO FINAL O NO DE COLA: SOLUCIÓN CASO TRIVIAL

$\{ a \geq 0 \wedge n \geq 0 \}$

Funcion POTENCIA (a, n : entero) retorna (p : entero)

caso

$n = 0 \rightarrow 1$

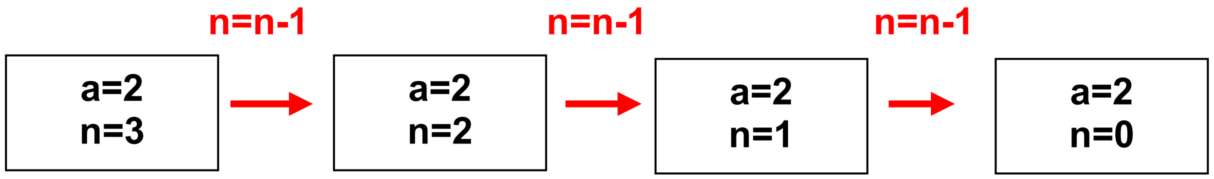
$n > 0 \rightarrow$ POTENCIA(a,n-1) * a

fcaso

ffunción

$\{ p = a^n \}$

s(a,n) = (a,n-1)



NO FINAL O NO DE COLA: SOLUCIÓN CASO TRIVIAL

$\{ a \geq 0 \wedge n \geq 0 \}$

Funcion POTENCIA (a, n : entero) retorna (p : entero)

caso

$n = 0 \rightarrow 1$

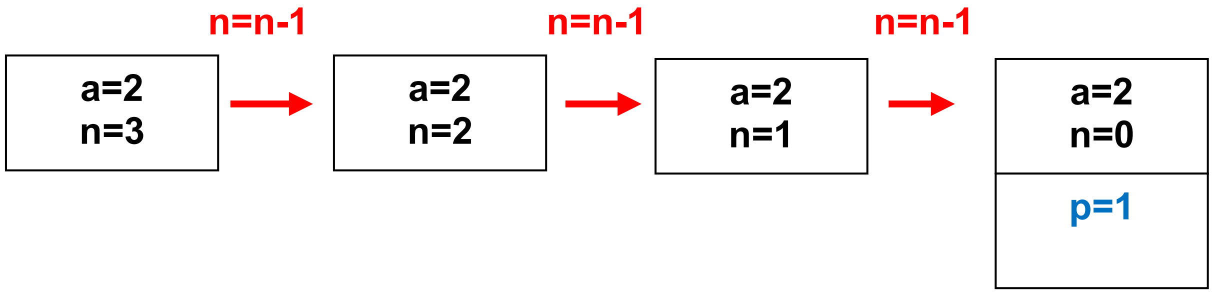
$n > 0 \rightarrow$ POTENCIA(a,n-1) * a

fcaso

ffunción

$\{ p = a^n \}$

s(a,n) = (a,n-1)



Caso base → n=0 → La función retorna 1.

NO FINAL O NO DE COLA: ASCENSO CADENA DE LLAMADAS RECURSIVAS

$\{ a \geq 0 \wedge n \geq 0 \}$

Funcion POTENCIA (a, n : entero) retorna (p : entero)

caso

$n = 0 \rightarrow 1$

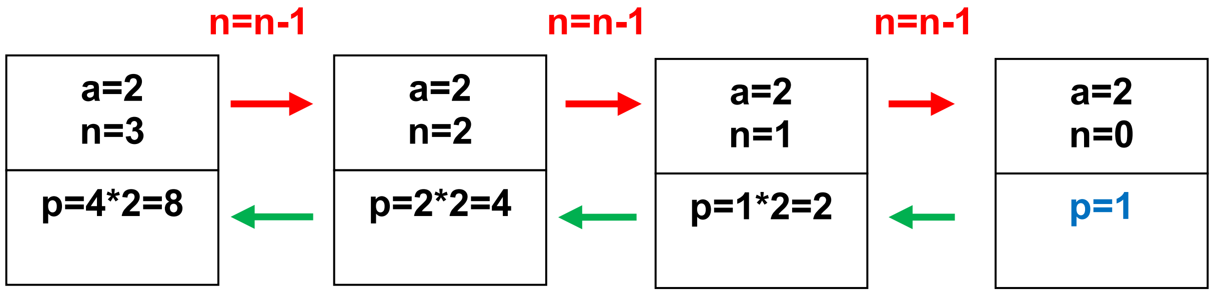
$n > 0 \rightarrow$ POTENCIA(a,n-1) * a

fcaso

ffunción

$\{ p = a^n \}$

s(a,n) = (a,n-1)



p = p´ * a

III

POTENCIA(a,n)=POTENCIA(a,n-1) * a

EJEMPLO DE FUNCIÓN RECURSIVA FINAL O DE COLA

{ $a > 0 \wedge b > 0$ }

Funcion MCD (a, b : entero) retorna (g : entero)

caso

$a = b \rightarrow a$

$a > b \rightarrow$ MCD(a-b, b)

$a < b \rightarrow$ MCD(a, b-a)

fcaso

ffunción

{ g = mcd(a,b) }

FINAL O DE COLA: DESCENSO CADENA DE LLAMADAS RECURSIVAS

$\{ a > 0 \wedge b > 0 \}$

Funcion MCD (a, b : entero) retorna (g : entero)

caso

$a = b \rightarrow a$

$a > b \rightarrow$ MCD(a-b, b)

$a < b \rightarrow$ MCD(a, b-a)

fcaso

ffunción

$\{ g = mcd(a,b) \}$

s(a,b)=si a > b entonces a = a - b sino b = b - a fsi

a=48 b=60 $\xrightarrow{\text{b=b-a}}$ a=48 b=12 $\xrightarrow{\text{a=a-b}}$ a=36 b=12 $\xrightarrow{\text{a=a-b}}$ a=24 b=12 $\xrightarrow{\text{a=a-b}}$ a=12 b=12

FINAL O DE COLA: SOLUCIÓN CASO TRIVIAL

$\{ a > 0 \wedge b > 0 \}$

Funcion MCD (a, b : entero) retorna (g : entero)

caso

$a = b \rightarrow a$

$a > b \rightarrow$ MCD(a-b, b)

$a < b \rightarrow$ MCD(a, b-a)

fcaso

ffunción

$\{ g = mcd(a,b) \}$

s(a,b)=si a > b entonces a = a - b sino b = b - a fsi

a=48 b=60 $\xrightarrow{b=b-a}$ a=48 b=12 $\xrightarrow{a=a-b}$ a=36 b=12 $\xrightarrow{a=a-b}$ a=24 b=12 $\xrightarrow{a=a-b}$ a=12 b=12

g=12

Caso base → a=b → La función retorna a, en nuestro ejemplo, 12.

FINAL O DE COLA: ASCENSO CADENA DE LLAMADAS RECURSIVAS

{ $a > 0 \wedge b > 0$ }

Funcion MCD (a, b : entero) retorna (g : entero)

caso

$a = b \rightarrow a$

$a > b \rightarrow MCD(a-b, b)$

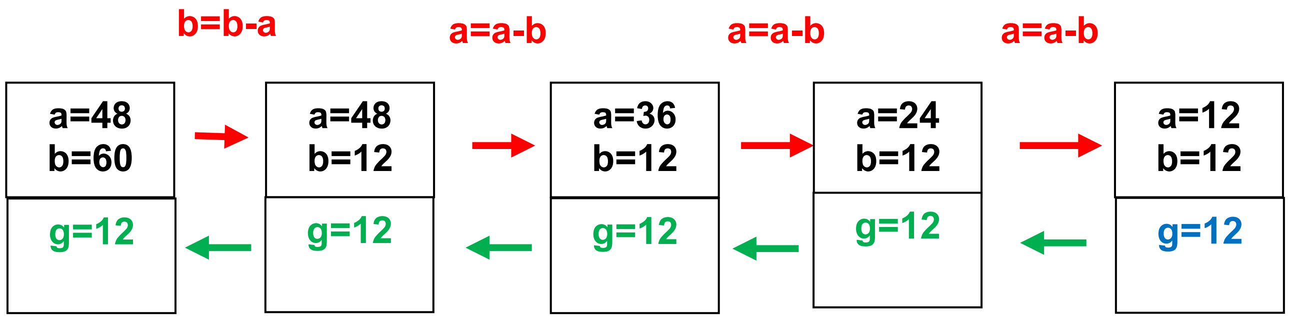
$a < b \rightarrow MCD(a, b-a)$

fcaso

ffunción

{ g = mcd(a,b) }

s(a,b)=si a > b entonces a = a - b sino b = b - a fsi



g = g´

III

MCD(a,b)=MCD(a-b,b) si a>b

MCD(a,b)=MCD(a,b-a) si a<b

EJEMPLO DE FUNCIÓN RECURSIVA NO FINAL O NO DE COLA

$\{ n \geq 0 \}$

```
Funcion POTENCIA3 (n : entero) retorna (f : entero)
   caso
      n = 0 → 1
      n = 1 → 3
      n > 1 → 2 * POTENCIA3(n-1) + 3 * POTENCIA3(n-2)
   fcaso
ffunción
```

$\{ f = 3^n \}$

NO FINAL O NO DE COLA: DESCENSO CADENA DE LLAMADAS RECURSIVAS

$\{ n \geq 0 \}$

Funcion POTENCIA3 (n : entero) retorna (f : entero)

caso

$n = 0 \rightarrow 1$

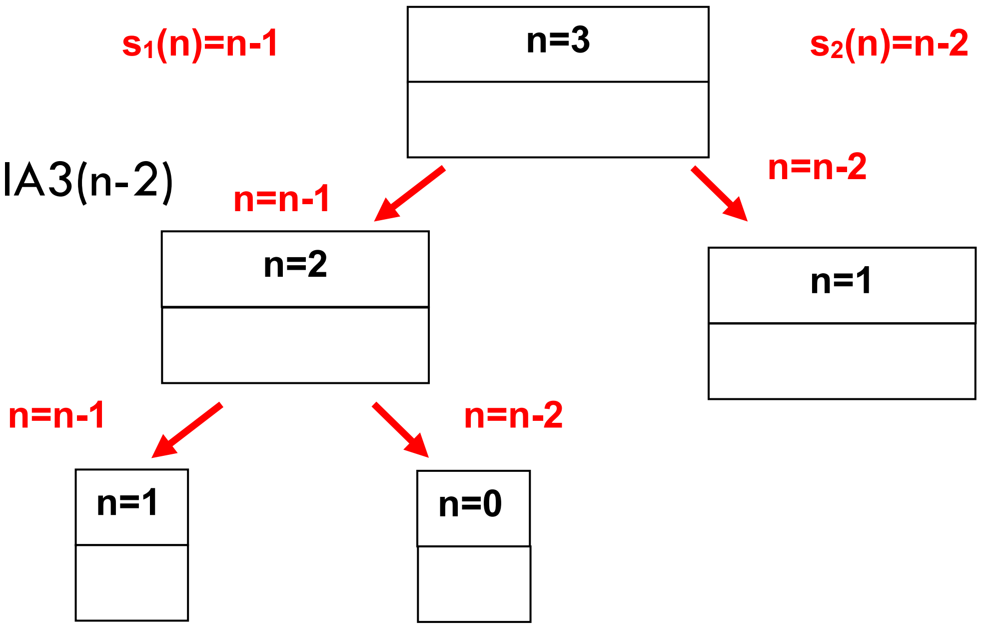
$n = 1 \rightarrow 3$

$n > 1 \rightarrow 2$ * POTENCIA3(n-1) + 3 * POTENCIA3(n-2)

fcaso

ffunción

$\{ f = 3^n \}$



NO FINAL O NO DE COLA: SOLUCIÓN CASOS TRIVIALES

$\{ n \geq 0 \}$

Funcion POTENCIA3 (n : entero) retorna (f : entero)

caso

$n = 0 \rightarrow 1$

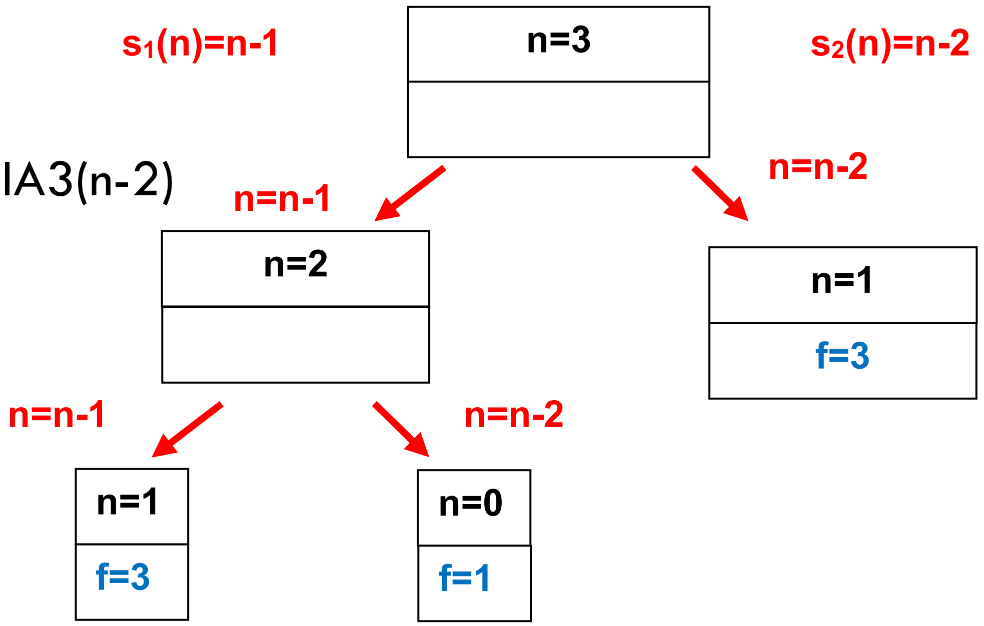
$n = 1 \rightarrow 3$

$n > 1 \rightarrow 2 * \text{POTENCIA3}(n-1) + 3 * \text{POTENCIA3}(n-2)$

fcaso

ffunción

$\{ f = 3^n \}$



NO FINAL O NO DE COLA: ASCENSO CADENA DE LLAMADAS RECURSIVAS

$\{ n \geq 0 \}$

Funcion POTENCIA3 (n : entero) retorna (f : entero)

caso

$n = 0 \rightarrow 1$

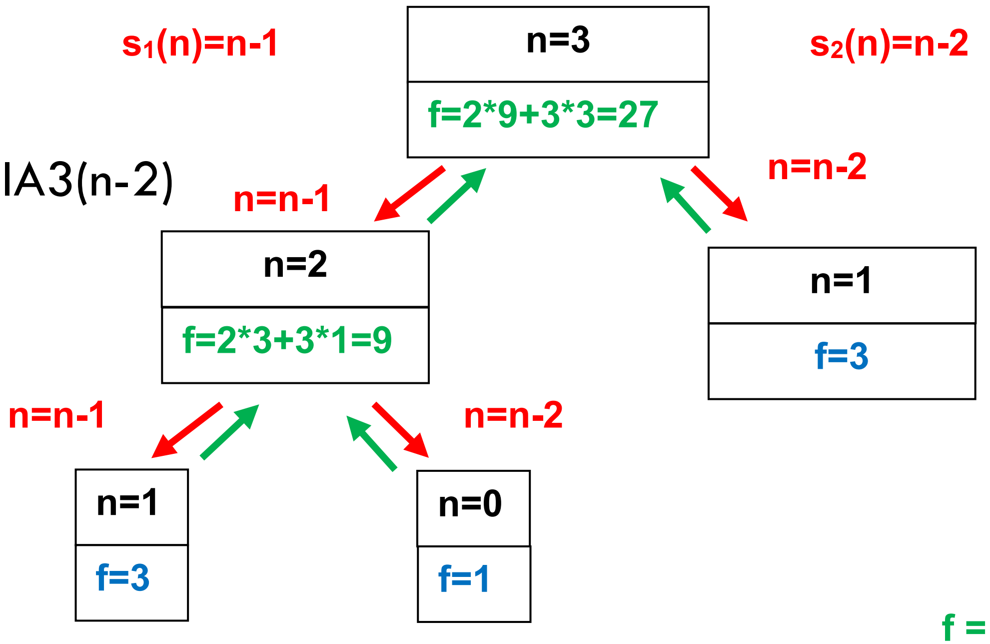
$n = 1 \rightarrow 3$

$n > 1 \rightarrow 2 * \text{POTENCIA3}(n-1) + 3 * \text{POTENCIA3}(n-2)$

fcaso

ffunción

$\{ f = 3^n \}$



f = 2*f1+3*f2

III

F(n)=2*F(n-1)+3*F(n-2)

TAREAS PARA EL ALUMNO

- Revisar que la postcondición se cumple en cada uno de los resultados (intermedios y finales) que figuran en los ejemplos de las diapositivas 16, 20 Y 24.
- Al alumno se le facilita un código fuente (sesion_2_1_practicas_recursion_alumno_2022_2023.c) que incluye la implementación "directa" del algoritmo que figura en la diapositiva 4 (función POTENCIA). En dicho código fuente aparece otra función, POTENCIA_entresijos, que añade sentencias de escritura a la función POTENCIA. Dichas sentencias tienen como objetivo mostrar por pantalla los valores de los parámetros correspondientes a cada invocación a la función, así como sus resultados.
- Renombrar el código fuente proporcionado incluyendo nombre y apellidos del alumno.
- Añadir a dicho código fuente la implementación de los algoritmos de las diapositivas 6 y 8 (POTENCIA3 y MCD). Proceder de forma similar al ejemplo expuesto anteriormente (implementación "directa" e implementación con sentencias de escritura). Ejecutar el programa, observando la evolución de los datos de las diferentes invocaciones a las funciones (POTENCIA, POTENCIA3 y MCD) así como los resultados.

TAREAS PARA EL ALUMNO

- Añadir a dicho código fuente la implementación de las funciones recursivas vistas hasta ahora, o propuestas para su realización, en clases expositivas: factorial, número_cifras, suma_cifras, semifactorial y fibonacci.

- Añadir a dicho código fuente la implementación de las dos funciones recursivas que resuelvan los siguientes problemas:

  - Dado un vector V[1..n] de enteros, con $n>0$, se pide diseñar una función recursiva que contabilice cuántos elementos del vector V son pares

  - Dada una matriz M[1..n][1..n] de enteros, con $n>0$, se pide diseñar una función recursiva que contabilice cuántas veces ocurre que $M[i][j]$ es igual a $M[j][i]$ con $1 \leq i, j \leq n$

TAREAS PARA EL ALUMNO

- [OPCIONAL] Añadir a dicho código fuente la implementación de dos funciones recursivas vistas en clase expositiva:
  - Dado un vector V[1..n] de enteros, con n>0, se pide diseñar una función recursiva que sume los elementos del vector V
  - Dada una matriz M[1..n][1..n] de enteros, con n>0, se pide diseñar una función recursiva que determine si dicha matriz M es simétrica o no.
- Al finalizar la sesión de prácticas, entregar a través del Campus Virtual el fichero fuente final reuniendo las tareas realizadas por el alumno.

ALGORITMIA

PRÁCTICAS – SESIÓN 2.2

- Da solución a problemas de forma natural, sencilla, comprensible y con un esfuerzo de razonamiento menor
- Da lugar a algoritmos más compactos
- Presenta facilidad para verificar formalmente que la solución es correcta
- En general, las soluciones recursivas son más ineficientes en tiempo y en espacio que las versiones iterativas, esto se debe al mecanismo de llamadas continuas a la función y al paso de parámetros, al uso de una pila donde cada uno de sus elementos reserva espacio para una activación de la función recursiva (direcciones de parámetros y variables locales).

En esta sección se muestran las **versiones iterativas** de los esquemas genéricos **recursivo simple no final** y **recursivo simple final** vistos en la sesión de prácticas anterior. Recordemos que:

- **función recursiva simple** es cuando la función recursiva genera a lo sumo una llamada interna por cada llamada externa

- **función recursiva no final** es cuando la función c (función de combinación) es necesaria, esto es,

$$f(\bar{x}) = c(f(s(\bar{x})), \bar{x})$$

- **función recursiva final** es cuando la función c (función de combinación) no es necesaria, esto es,

$$f(\bar{x}) = f(s(\bar{x}))$$

TRANSFORMACIÓN A ITERATIVO DE UNA RECURSIÓN SIMPLE Y NO FINAL

- La transformación da lugar a **dos bucles**.
- El **primero** corresponde al **descenso en la cadena de llamadas recursivas**, transformando los parámetros $\bar{x}$ de la llamada en curso en los parámetros s($\bar{x}$) de la llamada sucesora, hasta encontrar el valor $\bar{x}$ correspondiente al caso trivial. Por ello se aplica repetidamente la función sucesor, función s.
- La asignación posterior al primer bucle calcula el primer resultado $\bar{y}$, que corresponde al caso trivial.
- El **segundo bucle** representa el **ascenso en la cadena de llamadas**, aplicando reiteradamente la función c (función de combinación) para calcular los resultados de la llamada en curso en función de los de la llamada sucesora. Antes de aplicar la función c es necesario recuperar los parámetros $\bar{x}$ de la llamada en curso a partir de los de la llamada sucesora. Para ello, se aplica la función $s^{-1}(\bar{x})$, eso es, la inversa de la función sucesor (función s).

A continuación se muestra el esquema general de una función recursiva no final y de su versión iterativa.

TRANSFORMACIÓN A ITERATIVO DE UNA RECURSIÓN SIMPLE Y NO FINAL

{ Q($\bar{x}$) }

función f($\bar{x}$: T_1) retorna ($\bar{y}$: T_2)

caso

Bt ($\bar{x}$) $\rightarrow$ triv($\bar{x}$)

Bnt ($\bar{x}$) $\rightarrow$ c(f(s($\bar{x}$)), $\bar{x}$)

fcaso

ffunción

{ R($\bar{x}$,$\bar{y}$) }

Función recursiva simple y no final

TRANSFORMACIÓN A ITERATIVO DE UNA RECURSIÓN SIMPLE Y NO FINAL

{ Q($\bar{x}$) }

función f($\bar{x}$: T_1) retorna ($\bar{y}$: T_2)

  caso

    Bt ($\bar{x}$) $\rightarrow$ triv($\bar{x}$)

    Bnt ($\bar{x}$) $\rightarrow$ c(f(s($\bar{x}$)), $\bar{x}$)

  fcaso

ffunción

{ R($\bar{x}$,$\bar{y}$) }

Función recursiva simple y no final

{ Q($\bar{x}_{inicial}$) }

función f($\bar{x}_{inicial}$: T_1) retorna ($\bar{y}$: T_2)

  var $\bar{x}$: T_1 fvar

  $\bar{x} = \bar{x}_{inicial}$

  mientras Bnt($\bar{x}$) hacer

    $\bar{x}$ = s($\bar{x}$)

  fmientras

  $\bar{y}$ = triv($\bar{x}$)

  mientras $\bar{x} \neq \bar{x}_{inicial}$ hacer

    $\bar{x}$ = s^{-1}($\bar{x}$)

    $\bar{y}$ = c ($\bar{y}$, $\bar{x}$)

  fmientras

  retorna $\bar{y}$

ffunción

{ R($\bar{x}_{inicial}$, $\bar{y}$) }

Función iterativa equivalente

EJEMPLO FUNCIÓN RECURSIVA SIMPLE Y NO FINAL: POTENCIA

$Q \equiv \{ a \geq 0 \wedge n \geq 0 \}$

Funcion POTENCIA (a, n:entero) retorna (p:entero)

caso

$n = 0 \rightarrow 1$

$n > 0 \rightarrow$ POTENCIA(a,n-1) * a

fcaso

ffunción

$R \equiv \{ p = a^n \}$

Función recursiva simple y no final

$Q \equiv \{ a \geq 0 \wedge n_{inicial} \geq 0 \}$

Funcion POTENCIA (a, $n_{inicial}$:entero) retorna (p:entero)

var n: entero fvar

$n = n_{inicial}$

mientras $n > 0$ hacer

$n = n - 1$

fmientras

$p = 1$

mientras $n \neq n_{inicial}$ hacer

$n = n + 1$

$p = p * a$

fmientras

retorna p

ffunción

$R \equiv \{ p = a^{n_{inicial}} \}$

Función iterativa equivalente

EJEMPLO FUNCIÓN RECURSIVA SIMPLE Y NO FINAL: POTENCIA (mejora)

$Q \equiv \{ a \geq 0 \wedge n \geq 0 \}$

Funcion POTENCIA (a, n:entero) retorna (p:entero)

caso

$n = 0 \rightarrow 1$

$n > 0 \rightarrow$ POTENCIA(a,n-1) * a

fcaso

ffunción

$R \equiv \{ p = a^n \}$

Función recursiva simple y no final

$Q \equiv \{ a \geq 0 \wedge n_{inicial} \geq 0 \}$

Funcion POTENCIA (a, $n_{inicial}$:entero) retorna (p:entero)

var n: entero fvar

~~$n = n_{inicial}$~~

~~mientras n > 0 hacer~~

~~n = n – 1~~

~~fmientras~~

$\Big\} \; n = 0$

p = 1

mientras $n \neq n_{inicial}$ hacer

n = n + 1

p = p * a

fmientras

retorna p

ffunción

$R \equiv \{ p = a^{n_{inicial}} \}$

Función iterativa equivalente

TRANSFORMACIÓN A ITERATIVO DE UNA RECURSIÓN SIMPLE Y NO FINAL CUANDO NO EXISTE s^{-1}

- No siempre existe la inversa de la función s (sucesor).

- En ese caso, la implementación de s^{-1} consistirá en recuperar $\overline{x}$ de una estructura de almacenamiento donde, durante el bucle de descenso, se han almacenado los parámetros $\overline{x}$ de todas las llamadas.

EJEMPLO FUNCIÓN RECURSIVA SIMPLE Y NO FINAL CUANDO NO EXISTE s^{-1}

$Q \equiv \{ (a \geq 0) \wedge (b \geq 0) \}$

```
Funcion PRODUCTO (a, b : entero) retorna (p:entero)
   var p´: entero fvar
   si a = 0 entonces retorna 0
   sino
      p´ = PRODUCTO(adiv2, b*2)
      si a % 2 = 0 entonces retorna p´
      sino retorna p´+ b
      fsi
   fsi
ffunción
```

$R \equiv \{ p = a * b \}$

Función recursiva simple y no final

EJEMPLO FUNCIÓN RECURSIVA SIMPLE Y NO FINAL CUANDO NO EXISTE s^{-1}

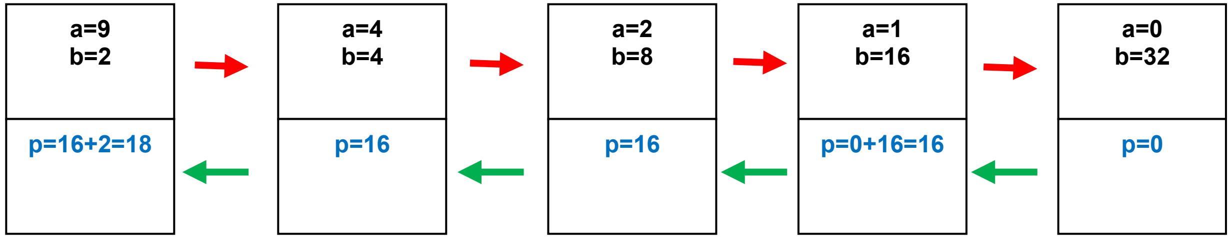
$Q \equiv \{ (a \geq 0) \wedge (b \geq 0) \}$

```
Funcion PRODUCTO (a, b : entero) retorna (p:entero)
   var p´: entero fvar
   si a = 0 entonces retorna 0
   sino
      p´ = PRODUCTO(adiv2, b*2)
      si a % 2 = 0 entonces retorna p´
      sino retorna p´+ b
      fsi
   fsi
ffunción
```

$R \equiv \{ p = a * b \}$

Función recursiva simple y no final

s(a,b) = (adiv2,b*2)




```
si a%2=0 entonces p = p’
      sino p = p’ + b
fsi
```

VERSIÓN ITERATIVA DE PRODUCTO

$Q \equiv \{ (a_{inicial} \geq 0) \wedge (b_{inicial} \geq 0) \}$

función PRODUCTO ($a_{inicial}$, $b_{inicial}$: entero) retorna (p: entero)

var a, b, i, p: entero, vector_as[1..$a_{inicial}$]: vector de enteros fvar

i =1
vector_as[i] = $a_{inicial}$
a = $a_{inicial}$
b = $b_{inicial}$

$\overline{x} = \overline{x}_{inicial}$

```
mientras a ≠ 0 hacer
   a = a / 2
   b = b * 2
   i = i + 1
   vector_as[i] = a
fmientras
```

VERSIÓN ITERATIVA DE PRODUCTO

$Q \equiv \{\ (a_{inicial} \geq 0) \wedge (b_{inicial} \geq 0)\ \}$

función PRODUCTO_iterativo ($a_{inicial}$, $b_{inicial}$: entero) retorna (p: entero)

var a, b, i, p: entero, vector_as[1..$a_{inicial}$]: vector de enteros fvar

i =1
vector_as[i] = $a_{inicial}$
a = $a_{inicial}$
b = $b_{inicial}$

mientras a ≠ 0 hacer
  a = a / 2
  b = b * 2
  i = i + 1
  vector_as[i]=a
fmientras

$\bar{x} = \bar{x}_{inicial}$

mientras $B_{nt}(\bar{x})$ hacer
  $\bar{x} = s(\bar{x})$
fmientras

VERSIÓN ITERATIVA DE PRODUCTO

p = 0	$\bar{y}$ = triv($\bar{x}$)
mientras a ≠ $a_{inicial}$ hacer	mientras $\bar{x} \neq \bar{x}_{inicial}$ hacer
i = i - 1	
a = vector_as[i]	$\bar{x}$ = $s^{-1}(\bar{x})$
b = b / 2	
si a % 2 ≠ 0 entonces p = p + b fsi	$\bar{y}$ = c ($\bar{y}$, $\bar{x}$)
fmientras	fmientras
retorna p	retorna $\bar{y}$
ffuncion	

R ≡ { p = $a_{inicial}$ * $b_{inicial}$ }

VERSIÓN ITERATIVA DE PRODUCTO

p = 0

mientras a ≠ $a_{inicial}$ hacer

i = i - 1

a = vector_as[i]

b = b / 2

si a % 2 ≠ 0 entonces p = p + b fsi

fmientras

retorna p

ffuncion

R ≡ { p = $a_{inicial}$ * $b_{inicial}$ }

$\bar{y}$ = triv($\bar{x}$)

mientras $\bar{x} \neq \bar{x}_{inicial}$ hacer

$\bar{x}$ = $s^{-1}(\bar{x})$

$\bar{y}$ = c ($\bar{y}$, $\bar{x}$)

fmientras

retorna $\bar{y}$

VERSIÓN ITERATIVA DE PRODUCTO

p = 0

mientras a $\neq$ $a_{inicial}$ hacer
   i = i - 1
   a = vector_as[i]
   b = b / 2
   si a % 2 $\neq$ 0 entonces p = p + b fsi
fmientras

retorna p
ffuncion

R $\equiv$ { p = $a_{inicial}$ * $b_{inicial}$ }

$\bar{y}$ = triv($\bar{x}$)

mientras $\bar{x} \neq \bar{x}_{inicial}$ hacer

$\bar{x}$ = s^{-1}($\bar{x}$)

$\bar{y}$ = c ($\bar{y}$, $\bar{x}$)

fmientras

retorna $\bar{y}$

TRANSFORMACIÓN A ITERATIVO DE UNA RECURSIÓN SIMPLE Y FINAL

- La transformación da lugar a **un solo bucle**.
- La variable $\bar{x}$ toma sucesivamente el valor de los parámetros de cada llamada recursiva. La versión iterativa solo necesita espacio para una copia de los mismos.
- A continuación se muestra el esquema general de una función recursiva final y de su versión iterativa.

TRANSFORMACIÓN A ITERATIVO DE UNA RECURSIÓN SIMPLE Y FINAL

{ Q($\bar{x}$) }

función f($\bar{x}$: T_1) retorna ($\bar{y}$: T_2)

caso

Bt ($\bar{x}$) $\longrightarrow$ triv($\bar{x}$)

Bnt ($\bar{x}$) $\longrightarrow$ f(s($\bar{x}$))

fcaso

ffunción

{ R($\bar{x}$,$\bar{y}$) }

Función recursiva simple y final

TRANSFORMACIÓN A ITERATIVO DE UNA RECURSIÓN SIMPLE Y FINAL

{ Q($\bar{x}$) }
función f($\bar{x}$: T_1) retorna ($\bar{y}$: T_2)
  caso
    Bt ($\bar{x}$) $\rightarrow$ triv($\bar{x}$)
    Bnt ($\bar{x}$) $\rightarrow$ f(s($\bar{x}$))
  fcaso
ffunción
{ R($\bar{x}$,$\bar{y}$) }

Función recursiva simple y final

{ Q($\bar{x}_{inicial}$) }
función f($\bar{x}_{inicial}$: T_1) retorna ($\bar{y}$: T_2)
  var $\bar{x}$: T_1 fvar
  $\bar{x} = \bar{x}_{inicial}$
  mientras Bnt($\bar{x}$) hacer
    $\bar{x}$ = s($\bar{x}$)
  fmientras
  retorna triv($\bar{x}$)
ffunción
{ R($\bar{x}_{inicial}$, $\bar{y}$) }

Función iterativa equivalente

EJEMPLO FUNCIÓN RECURSIVA SIMPLE Y FINAL: TABLEROS

$Q \equiv \{ (n \geq m) \wedge (n > 0) \wedge (m > 0) \}$

Funcion TABLEROS (n, m : entero) retorna (h:entero)

caso

$m = 1 \rightarrow n^2$

$m > 1 \rightarrow$ TABLEROS(n-1, m-1)

fcaso

ffunción

$R \equiv \{ h = (n - m + 1)^2 \}$

Función recursiva simple y final

$Q \equiv \{ (n_{inicial} \geq m_{inicial}) \wedge (n_{inicial} > 0) \wedge (m_{inicial} > 0) \}$

Funcion TABLEROS ($n_{inicial}$, $m_{inicial}$: entero) retorna (h:entero)

var n, m: entero fvar

$n = n_{inicial}$

$m = m_{inicial}$

mientras $m > 1$ hacer

$n = n - 1$

$m = m - 1$

fmientras

retorna n^2

ffunción

$R \equiv \{ h = (n_{inicial} - m_{inicial} + 1)^2 \}$

Función iterativa equivalente

TAREAS PARA EL ALUMNO

- Al alumno se le facilita un código fuente (sesion_2_2_practicas_recursion_alumno_2022_2023.c) que incluye la implementación de una serie de funciones recursivas vistas, o planteadas, en clase.

- Renombrar el código fuente proporcionado incluyendo nombre y apellidos del alumno y realizar en dicho código fuente las siguientes tareas:

  1) Implementar la función recursiva TABLEROS que figura en este documento y su correspondiente versión iterativa

2) Implementar las transformaciones a iterativo de las siguientes funciones que figuran en el fichero .c:

- Función MCD_recursiva
- Función iSUMA_VECTOR_recursiva_backward
- Función iSUMA_VECTOR_recursiva_forward
- Función iSIMETRIA_MATRIZ_recursiva
- Función iCAPICUA_VECTOR_recursiva
- Función iMEDIA_ARITMETICA_recursiva

TAREAS PARA EL ALUMNO

3) [OPCIONAL] Implementar las transformaciones a iterativo de las funciones de la sesión 2.1:

- Función iCONTAR_PARES_VECTOR
- Función iCONTAR_COINCIDENCIAS_MATRIZ

- Al finalizar la sesión de prácticas, entregar a través del Campus Virtual el fichero fuente final reuniendo las tareas realizadas por el alumno.