UNIVERSIDAD DE A CORUÑA
FACULTAD DE INFORMÁTICA
DEPARTAMENTO DE COMPUTACIÓN

Tecnología de la Programación

Ingeniería Técnica en Informática de Sistemas

Elena Mª Hernández Pereira
Óscar Fontenla Romero

Bloque didáctico II: Semántica de programas

Tema 6

- Título: El lenguaje GCL, *Guarded Command Language*
- Unidades de contenido
  - Los comandos skip, abort y composición
  - El comando asignación
  - La estructura alternativa
  - La estructura iterativa
  - Funciones y procedimientos

Tema 6: GCL
Introducción

- GCL: *Guarded Command Language* [Dijkstra, 1976]
  - *Sintaxis simple y muy reducida*
  - *Más cómodo para realizar demostraciones formales*
  - *Permite asignaciones simultáneas y no determinismo*
- Semántica operacional
  - Funcionamiento de cada instrucción
  - Relación ⟨estado, instrucción⟩ → estado
  - Ej: ⟨ { (x,1), (y,2) }, "x := 0" ⟩ → { (x,0), (y,2) }
- Semántica axiomática
  - Caracterización (definción)
  - En términos del predicado *wp*

Tema 6: GCL
Sintaxis

- Instrucciones
  1. *skip*
  2. *abort*
  3. Composición secuencial $S_1;S_2$
  4. Asignación simple $x_1 := \alpha_1$
  5. Asignación múltiple $x_1, x_2, \ldots, x_m := \alpha_1, \alpha_2, \ldots, \alpha_m$
  6. Instrucción condicional if $B_1 \rightarrow S_1 | \ldots B_m \rightarrow S_m$ fi
  7. Instrucción iterativa do $B_1 \rightarrow S_1 | \ldots B_m \rightarrow S_m$ od

donde $m \geq 0$, x_i variables o referencias a arrays, α_i expresiones, B_i fórmulas (sin cuantificadores) y S_i instrucciones

Tema 6: GCL
skip y *abort*

- *skip*
  - Transformación identidad
  - $\langle s, \text{“skip”} \rangle \rightarrow s$
  - wp(skip,R) = R
- *abort*
  - $\langle s, \text{“abort”} \rangle \not\rightarrow s'$ para cualquier s y s'
  - wp(abort,R) = F
  - No existe ningún estado de partida que garantice que dicho comando termina en un estado definido y útil

Tema 6: GCL
Composición secuencial

- $S_1;S_2$
  - $\langle s, (S_1;S_2) \rangle \rightarrow s' \Leftrightarrow \langle s, S_1 \rangle \rightarrow s''$ y $\langle s'', S_2 \rangle \rightarrow s'$
  - *wp(“$S_1;S_2$”,R) = wp(S_1,wp(S_2,R))*
  - *Es asociativa*
    - *wp(“$S_1;(S_2;S_3)$”,R) = wp(“$(S_1;S_2);S_3$”,R)*
- *Ejemplos:*
  - *wp(“skip;skip”,R)*
    = wp(skip,wp(skip,R)) = wp(skip,R) = R
  - *wp(“S;abort”,R) = F*

Tema 6: GCL
Comando asignación

- *Asignación simple a variables simples*
- *Asignación múltiple a variables simples*
- *Asignación a un elemento de un array*
- *Asignación múltiple general*

Tema 6: GCL
Asignación simple a variables simples

- *x := α*
  - x variable simple, α expresión, ambas del mismo tipo
  - *El identificador x toma el valor de la expresión α*
  - ⟨s, "x := α"⟩ → (s; x:s(α)) siempre que s(α) ∈ rango(x)
  - wp("x:= α",R) = dominio(α) cand R^x_α
- Ejemplos
  - Dados x,y:integer y s={ (x,1), (y,2) }:
  - ⟨s, "y := x-1"⟩ → { (x,1), (y,0) }
  - ⟨s, "x := y; skip; y := y-1"⟩ → { (x,2), (y,1) }
  - ⟨s, "skip; abort"⟩ → no existe estado siguiente

Tema 6: GCL
Asignación simple a variables simples

- Ejemplos
  - wp("x:=5",x=5) = dominio(5) cand (5=5) = dominio(5)
  - wp("x:=5",x≠5) = dominio(5) cand (5 ≠ 5) = dominio(5) cand F = F
  - Para cualquier predicado *p* de un argumento wp("x:=a/b",p(x)) = dominio(a/b) cand $(p(x)^{x}_{a/b})$ = ((b ≠ 0) cand p(a/b))
- Ausencia de efectos colaterales
  - Dados x,y distintos y una constante C, wp("x:=e", y=C) = (y=C)

Tema 6: GCL
Asignación simple a variables simples

- Dada una secuencia de instrucciones

  wp("$S_1;S_2;S_3;S_4$",R)
  = wp("$S_1;S_2;S_3$",wp("S_4",R))
  = wp("$S_1;S_2$",wp("S_3",wp("S_4",R)))
  = wp("S_1",wp("S_2",wp("S_3",wp("S_4",R))))
- Lo simplificamos como
  - $\{wp_1\}\ S_1\ \{wp_2\}\ S_2\ \{wp_3\}\ S_3\ \{wp_4\}\ S_4\ \{R\}$
  - $\{wp_1\}\ S_1;S_2;S_3;S_4\ \{R\}$

Tema 6: GCL

Asignación simple a variables simples

- Ejercicios: Determina y simplifica wp(S,R)

	S	R
(a)	$x := 2 * y + 3$	$x = 13$
(b)	$x := x + y$	$x < 2 * y$
(c)	$j := j + 1$	$0 < j \wedge (\forall I \in [0 : j] : b[I] = 5)$
(d)	$all5 := (b[j] = 5)$	$all5 = (\forall I \in [0 : j] : b[I] = 5)$
(e)	$all5 := all5 \wedge (b[j] - 5)$	$all5 - (\forall I \in [0 : j] : b[I] - 5)$
(f)	$x := x * y$	$x * y = c$
(g)	$x := (x - y) * (x + y)$	$x + y^2 \neq 0$

Tema 6: GCL

Asignación múltiple a variables simples

- $x_1, \ldots, x_n := \alpha_1, \ldots, \alpha_n \Rightarrow \bar{x} := \bar{\alpha}$
  - x_i variables simples distintas, α_i expresiones
  - $\langle s, \text{“}x_1, \ldots, x_n := \alpha_1, \ldots, \alpha_n\text{“}\rangle \rightarrow (s; x_1{:}s(\alpha_1); \ldots; x_n{:}s(\alpha_n))$ siempre que $s(\alpha_i) \in$ rango(x_i)
  - wp(“ $\bar{x} = \bar{\alpha}$ “,R) = dominio($\bar{\alpha}$) cand $R^{\bar{x}}_{\bar{\alpha}}$ donde dominio($\bar{\alpha}$) = ($\forall$I : dominio(α_i))
- Ejecución
  - Se evalúan las α_i en cualquier orden para obtener v_i
  - Se asigna v_1 a x_1, …, v_n a x_n en este orden

Tema 6: GCL

Asignación múltiple a variables simples

- Ejemplos
  - Dados x,y:integer y s={ (x,1), (y,2) }:
  - ⟨s, "x := y; y := x"⟩ → { (x,2), (y,2) }
  - ⟨s, "x,y := y,x"⟩ → { (x,2), (y,1) }
  - ⟨s, "x,y,x := y,x,x-1"⟩ → { (x,0), (y,1) }
  - ⟨s, "x,y := 0,y/x"⟩ → { (x,0), (y,2) }
- Ejercicios
  - wp("z,y:=z*x,y-1",((y ≥ 0) ∧ (z*x^y =c)))
  - wp("s,i:=s+b[i],i+1",((i > 0) ∧ s=(∑ J ∈ [0,i ⟩: b[J])))

Tema 6: GCL

Asignación múltiple a variables simples

- Ejercicios: Determina y simplifica wp(S,R)

	S	R
(a)	$z, x, y := 1, c, d$	$z * x^y = c^d$
(b)	$i, s := 1, b[0]$	$(1 \leq i < n) \wedge s = b[0] + ... + b[i-1]$
(c)	$a, n := 0, 1$	$a^2 < n \wedge ((a+1)^2 \geq n)$
(d)	$i, s := i+1, s+b[i]$	$(0 < i < n) \wedge s = b[0] + ... + b[i-1]$
(e)	$i := i+1; j := j+i$	$i = j$
(f)	$j := j+i; i := i+1$	$i = j$
(g)	$i, j := i+1, j+i$	$i = j$

Tema 6: GCL

Asignación múltiple a variables simples

- Utilidad: Derivación de programas
- Dado *b*:array, *i*,*m*,*p*:integer con $i \leq m < i+p$
  - *i*, *(i+p-1)* límites de una partición de *p*
  - *m* índice de esa partición
  - Se pretende hacer más pequeña la partición haciendo *i = m+1* sin cambiar *p*
  - ¿Qué valor se asigna a *p*?
- Dado: { T } a:= a+1; b:= x { a= b }
  - Encontrar un valor para *x*

Tema 6: GCL

Asignación múltiple a variables simples

- Ejercicios: Determina y simplifica wp(S,R)

(a) $\{T\}a, b := a+1, x\{b = a+1\}$
(b) $\{T\}a := a+1; b := x\{b = a+1\}$
(c) $\{T\}b := x; a := a+1\{b = a+1\}$
(d) $\{i = j\}i, j := i+1, x\{i = j\}$
(e) $\{i = j\}i := i+1; j := x\{i = j\}$
(f) $\{i - j\}j := x; i := i+1\{i - j\}$
(g) $\{z + a * b = c\}z, a := z+b, x\{z + a * b = c\}$
(h) $\{even(a) \wedge z + a * b = c\}a, b := a/2, x\{z + a * b = c\}$
(i) $\{even(a) \wedge z + a * b = c\}a := a/2; b := x\{z + a * b = c\}$
(j) $\{T\}i, s := 0, x\{s = (\sum J : 0 \leq J \leq i : b[J])\}$
(k) $\{T\}i, s := 0, x\{s = (\sum J : 0 \leq J < i : b[J])\}$
(l) $\{i > 0 \wedge s = (\sum J : 0 \leq J < i : b[J])\}i, s := i+1, x\{s = (\sum J : 0 \leq J < i : b[J])\}$

Tema 6: GCL

Asignación a un elemento de un array

- Recordamos
  - Array: variable simple que contiene una función
  - b[i] $\Rightarrow$ aplicación de la función b sobre el argumento i
  - (b; i : α) $\Rightarrow$ función igual a b excepto en el argumento i que produce el valor α
  - b[i] := $\alpha \equiv$ b := (b; i : α)

- *b := (b; i : α)*

Tema 6: GCL

Asignación a un elemento de un array

- *b := (b; i : α)*
  - Dado un selector $\sigma = [e_1][e_2]\ldots[e_n]$ donde cada e_i son expresiones, definimos su evaluación en *s*
    - $s(\sigma) = [s(e_1)]\,[s(e_2)]\ldots\,[s(e_n)]$
  - La ejecución de bσ := α produce el efecto:
  - $\langle$ s, "bσ := α" $\rangle \rightarrow$ (s; b: (s(b); s(σ):s(α)))
    siempre que $e_i \in$ dominio$_i$(b) y $\alpha \in$ rango(b)
  - Tanto α como σ se evalúan en el estado *s*

Tema 6: GCL

Asignación a un elemento de un array

- Ejemplos:
  - Dado b[0:2]:integer y s = { (i,1), (b, (0,1,20)) }
  - ⟨ s, "b[i+1] := b[b[i]] + 6" ⟩ → { (i,1), (b, (0,1,7)) }
  - ⟨ s, "b[b[i-1]] := 7" ⟩ → { (i,1), (b, (7,1,20)) }
  - ⟨ s, "b[i+2] := 7" ⟩ → no hay estado siguiente

Tema 6: GCL

Asignación a un elemento de un array

- *b := (b; i : α)*
  - wp("b[i] := α",R) = wp("b := (b; i:α)",R)
    = dominio((b; i:α)) cand $R^b_{(b;\ i:\alpha)}$
    = enrango(i,b) cand dominio(α) cand $R^b_{(b;\ i:\alpha)}$
- Ejemplos:
  - wp("b[i] := 5", b[i]=5)
  - wp("b[i] := 5", b[i]=b[j])
  - wp("b[b[i]] := i", b[i]=i)
  - wp("b[n] := x", ordenado(b[1:n]))

Tema 6: GCL

Asignación a un elemento de un array

- Ejercicios: Determina y simplifica wp(S,R)

(a) $wp("b[i] := i", b[b[i]] = i)$
(b) $wp("b[i] := 5", (\exists J : i \leq J < n : b[i] \leq b[J])$
(c) $wp("b[i] := 5", (\exists J : i \leq J < n : b[i] < b[J])$
(d) $wp("b[i] := 5", b[0 : n-1] = B[0 : n-1])$
(e) $wp("b[i] := b[i-1] + b[i]", b[i] = (\sum J : 1 \leq J < i : b[J])$
(f) $wp("t := b[i]; b[i] := b[j]; b[j] := t", b[i] = x \wedge b[j] = y)$
(g) $wp("t := b[i]; b[i] := b[j]; b[j] := t", k \neq i \wedge k \neq j \wedge b[k] = C)$

Tema 6: GCL

Asignación múltiple general

- $x_1\sigma_1,\ldots, x_n\sigma_n := \alpha_1,\ldots,\alpha_n \Rightarrow \quad \overline{x\sigma} = \overline{\alpha}$
  - x_i identificadores, σ_i selectores y α_i expresiones del mismo tipo que $x_i\sigma_i$
  - $\langle$ s, "$x_1\sigma_1,\ldots,x_n\sigma_n := \alpha_1,\ldots,\alpha_n$" $\rangle \rightarrow$ (s; x_1: (s(x_1); s(σ_1):s(α_1)); …; x_n: (s(x_n); s(σ_n):s(α_n))) siempre que $\sigma_i \in$ dominio$_i$(x) y $\alpha_i \in$ rango(x_i)
  - wp(" $\overline{x\sigma} = \overline{\alpha}$,R)= dominio($\overline{\alpha}$) cand $R^{x\sigma}_{\alpha}$ = $\boxed{R^{x\sigma}_{\alpha}}$
- Ejecución
  1. Se evalúan todos los σ_i en s
  2. Se evalúan todas las α_i en s
  3. Se asigna de izquierda a derecha

Tema 6: GCL

Asignación múltiple general

- Problema:
  - Redefinir la sustitución textual ya que $\bar{x}\sigma$ no son sólo identificadores
- Dos casos:
  - *$b_1\sigma_1,\ldots, b_n\sigma_n := \alpha_1,\ldots,\alpha_n$ asigna primero α_1 a $b_1\sigma_1$, después $\alpha_2,\ldots$,a $b_2\sigma_2$ y así sucesivamente, por lo tanto es equivalente a b := (b; σ_1: α_1; …;σ_n: α_n)*
  - *Sean b,c identificadores distintos y σ, ρ dos selectores, entonces bσ, cρ := α,β debe tener el mismo efecto que cρ, bσ :=β,α*

Tema 6: GCL

Asignación múltiple general

- Definición P^{x}_{α}

1. *$\bar{x}$, lista de identificadores distintos y todos los selectores son ε, entonces tenemos la sustitución textual ya definida*
2. *Parejas adyacentes de ref-expr se pueden permutar siempre que comiencen por identificadores distintos*

$$R^{\bar{x},b\circ\sigma,c\circ\rho,\bar{y}}_{\bar{\alpha},f,h,\bar{\beta}} = R^{\bar{x},c\circ\rho,b\circ\sigma,\bar{y}}_{\bar{\alpha},h,f,\bar{\beta}}$$

3. *Asignaciones múltiples a partes de un objeto b pueden verse como una única asignación a b*

$$R^{b_1\circ s_1,\ldots,b_m\circ s_m,\bar{x}}_{\alpha_1,\ldots,\alpha_m,\bar{g}} = R^{b,\bar{x}}_{(b;s_1:\alpha_1;\ldots;s_m:\alpha_m),\bar{g}}$$

Tema 6: GCL

Asignación múltiple general

- Ejercicios: Determina y simplifica wp(S,R)

(a) $wp(\text{"}b[i], b[2] := 3, 4\text{"}, b[i] = 3)$
(b) $wp(\text{"}b[i], b[2] := 4, 4\text{"}, b[i] = 3)$
(c) $wp(\text{"}p, b[p] := b[p], p\text{"}, p = b[p])$
(d) $wp(\text{"}i, b[i] := i + 1, 0\text{"}, 0 < i \wedge (\forall j : 0 \leq j < i : b[j] = 0)$
(e) $wp(\text{"}i, b[i] := i + 1, 0\text{"}, 0 < i \wedge b[0 : i - 1] = 0)$
(f) $wp(\text{"}p, b[p], b[q] := b[p], b[q], p\text{"}, p = b[q])$
(g) $wp(\text{"}p, b[p], b[b[p]] := b[p], b[b[p]], p\text{"}, p = b[b[p]])$
(h) $wp(\text{"}p, b[p], b[b[p]] := b[p], b[b[p]], p\text{"}, p \neq b[b[p]])$

Tema 6: GCL
Estructura alternativa

- Instrucción condicional
- If $B_1 \rightarrow S_1 \mid B_2 \rightarrow S_2 \mid \ldots \mid B_n \rightarrow S_n$ fi
  - $n \geq 0$
  - S_i instrucciones
  - B_i predicados booleanos, condiciones o custodias (guardianes)
- Abreviaturas:
  - **IF**
  - $BB \equiv B_1 \vee \ldots \vee B_n$

Tema 6: GCL
Estructura alternativa

- $\langle$ s, IF $\rangle \rightarrow$ s' se da siempre que:
  - $\neg \exists$ B_i tal que $s(B_i) = U$
  - s' es un estado tal que para una rama i, $s(B_i) = T$ y $\langle s, S_i \rangle \rightarrow s'$
- Es decir,
  - Si $\exists$ B_i tal que $s(B_i) = U$, aborta
  - Si $\neg \exists$ B_i tal que $s(B_i) = T$, aborta
  - Toma un B_i cualquiera que sea cierto y ejecuta S_i

Tema 6: GCL
Estructura alternativa

- Ejemplo
  - S_1 = { (a,1), (b,0), (x,0) }
  - S_2 = { (a,0), (b,1), (x,0) }
  - S_3 = { (a,2), (b,1), (x,0) }
  - S_4 = { (a,1), (b,1), (x,0) }

```
if a = 1   → x:=1
|  a/b > 0 → x:=2
|  a/b < 0 → x:=3
fi
```

- Sirve de if-then-else y de case
- No tiene rama default ni sentencia if-then
- Ejemplo:
  - if x<0 then x:=abs(x);

```
if x < 0 → x:=abs(x)
|  x ≥ 0 → skip
fi
```

Tema 6: GCL
Estructura alternativa

- wp(IF, R) = dominio (BB) $\wedge$ BB
  $\wedge (B_1 \Rightarrow wp(S_1, R)) \wedge \ldots$
  $\wedge (B_n \Rightarrow wp(S_n, R))$
- wp(IF, R) = $(\exists\ I \in [1,n]: B_1)$
  $\wedge (\forall\ I \in [1,n]: B_I \Rightarrow wp(S_I, R))$
- Ejemplos:
  - Programa que deja en x el valor absoluto de z
  - Programa que cuenta el número de valores positivos de un array b[0:n-1]

Tema 6: GCL
Estructura alternativa

- Teorema:
- Probar que $Q \Rightarrow$ wp(IF, R) equivale a probar:
  - $Q \Rightarrow$ dominio (BB)
  - $Q \Rightarrow$ BB
  - $Q \wedge B_I \Rightarrow wp(S_I, R)\ \ \forall i \in [1:n]$

Tema 6: GCL
Estructura repetitiva

- Instrucción iterativa
- do $B_1 \rightarrow S_1 \mid B_2 \rightarrow S_2 \mid \ldots \mid B_n \rightarrow S_n$ od
  - $n \geq 0$
  - S_i instrucciones
  - B_i predicados booleanos, condiciones o custodias (guardianes)
- Abreviaturas:
  - **DO**
  - $BB \equiv B_1 \vee \ldots \vee B_n$

Tema 6: GCL
Estructura repetitiva

- $\langle s, DO \rangle \rightarrow s'$ se da siempre que:
  - $\neg \exists B_i$ tal que $s(B_i) = U$
  - s' es un estado tal que
    - $\exists i, s(B_i) = T$ y $\langle s, (S_i:DO) \rangle \rightarrow s'$
    - $\forall i, s(B_i) = F$ y $s' = s$
- Es decir,
  - Si $\exists B_i$ tal que $s(B_i) = U$, aborta
  - Si $\neg \exists B_i$ tal que $s(B_i) = T$, finaliza en el estado *s*
  - Toma un B_i cualquiera que sea cierto y ejecuta S_i y en el estado resultante vuelve a ejecutar DO

Tema 6: GCL
Estructura repetitiva

- DO en función de IF
  - do BB $\rightarrow$ if $B_1 \rightarrow S_1$ | … | $B_n \rightarrow S_n$ fi od
- Ejemplo:
  - En s = { (i,6) }, s' = { (i,3) } ejecutar los siguientes programas

```
do i/abs(i)=  1 → i:=i-2
|  i/abs(i)= -1 → i:=i+2
od
```

```
do i ≠ 0 cand i/abs(i)=  1 → i:=i-2
|  i ≠ 0 cand i/abs(i)= -1 → i:=i+2
od
```


Tema 6: GCL
Estructura repetitiva

- wp(DO, R)
  - Acabar en 0 iteraciones
    - $H_0(R) = \neg BB \wedge R$
  - Acabar en k > 0 iteraciones
    - $H_k(R) = wp(IF, H_{k-1}(R))$
  - Acabar en cualquier número de iteraciones
    - $wp(DO,R) = (\exists\ k: 0 \leq k: wp(IF, H_k(R)))$
- Problema:
  - No es aplicable en la práctica
  - El desarrollo de wp(DO,R) es recursivo y puede ser que nunca terminemos de elaborarlo

Tema 6: GCL
Estructura repetitiva

- **Técnica:**

(1) Para demostrar corrección parcial

- Buscamos una fórmula, llamada *invariante*, cierta antes y después de toda iteración del bucle (incluso al terminar)

(2) Para demostrar que el bucle finaliza

- Buscamos una función t, llamada *función cota*, que en cada iteración sea positiva y decreciente

Tema 6: GCL
Estructura repetitiva

- Ejemplo:

```
i, s := 1, b[0];
do i < 11 → i, s :=i+1, s+ b[i] od
{R: s = (∑K ∈[0,11⟩: b[K])}
```

- Ejemplo:

```
{ b ≥ 0}
x, y, z:= a, b, 0;
do y > 0 ∧ par(y) → y, x:= y/2, x+x
  |   impar(y)    → y, z:= y-1, z+x
od
{R: z = a*b}
```

Tema 6: GCL
Estructura repetitiva

- **Teorema**

Dados DO, P y t tales que:

(1) $P \wedge B_i \Rightarrow wp(S_i,P) \quad \forall i \in [1,n]$

(2) $P \wedge BB \Rightarrow t > 0$

(3) $P \wedge B_i \Rightarrow wp(\text{“}t_1:=t;\ S\text{”},t < t_1) \quad \forall i \in [1,n]$

Entonces $P \Rightarrow wp(DO, P \wedge \neg BB)$

Tema 6: GCL
Estructura repetitiva

```
{ Q: … }
{ inv P: … }
{ cota t: … }
do B1 → S1
| …
| Bn  → Sn
od
{R: …}
```

- **Método de prueba**

(1) P es cierto justo antes de comenzar el bucle

(2) $\{P \wedge B_i\}\ S_i\ \{P\}$ para cada rama

(3) $P \wedge \neg BB \Rightarrow R$

(4) $P \wedge BB \Rightarrow t > 0$

(5) $\{P \wedge B_i\}\ t_1:=t;\ S_i\ \{t < t_1\}$

Tema 6: GCL
Funciones y procedimientos

- $\text{func } f(\overbrace{v_1 : t_1;...;v_n : t_n}^{parametros}) \text{ ret } \overbrace{(r : t)}^{resultado}$

- **Ejemplos**

```
func abs(x:integer) ret (a:integer)
if x ≤ 0 → a:= -x
|  x > 0 → a:= x
fi
```

```
func fact(i:integer) ret (f:integer)
if i = 0 → f:= 1
|  i > 0 → f:= fact(i-1)*i
fi
```


Tema 6: GCL
Funciones y procedimientos

- $\text{proc } p(\overbrace{v_1 : t_1;...;v_n : t_n}^{valor}; \text{var } \overbrace{v_1' : t_1';...;v_m' : t_m'}^{referencia})$

- **Ejemplos**

```
proc swap(var i,j:integer)
i,j := j,i
```

```
proc div(n,d:integer; var c,r:integer)
c,r := 0,n;
do r ≥ d → c,r := c+1,r-d
od
```

Tema 6: GCL
Funciones y procedimientos

- **Ejercicios**
  - Dado un número *x*, devolver la primera posición *i* en un array *b[0:n-1]* tal que *b[i]* $\geq x$.
  - Insertar un elemento *x* en un array *b[0:n-1]* ordenado desde *0* a *n-2*.
  - Programar el procedimiento *div(n,d,c,r)* de modo recursivo.
  - Escribir una función recursiva para el máximo común divisor *mcd(a,b)*.

Tema 6: GCL
Funciones y procedimientos

- **Funciones**

```
func f( X) ret r
{Q} S {R}
```

- Por simplicidad, supondremos que
  - En Q sólo aparecen libres las $\overline{X}$
  - En R sólo aparecen libres las $\overline{X}$ y r
  - Las variables internas de f no se usan en el resto del programa

Tema 6: GCL
Funciones

- **Ejemplo**

```
func abs(X) ret r
{Q: T}
if X ≥ 0 → r:= X
|  x ≤ 0 → r:= -X
fi
{R: (X ≥ 0 ∧ r=X) ∨ (X ≤ 0 ∧ r=-X)}
```

- Las llamadas tienen la forma yσ := α(f($\bar{e}$)) donde:
  - $\bar{e}$ son expresiones del mismo tipo que $\bar{X}$
  - yσ coincide en tipo con r
  - α(f($\bar{e}$)) es una expresión que contiene a f($\bar{e}$) una sóla vez
  - f no aparece en σ ni en $\bar{e}$

Tema 6: GCL
Funciones

- Ejemplo de llamadas posibles
  - `y := abs(10 - a*2);`
  - `b[0] := abs(b[1] - b[2])`
  - `y := abs(a) + 1`
- Se descartan llamadas del tipo
  - `z := abs( abs(x) – z)`
  - `b[abs(i - j)] := 0`
- Que pueden ser sustituidas por asignaciones intermedias
  - aux_0 `:= abs(a); z := abs(`aux_0 `– z)`
  - aux_1 `:=abs(i – j); b[`aux_1`] := 0;`

Tema 6: GCL
Funciones

- La llamada y:=f($\overline{e}$) equivale a ejecutar
  - $\overline{X}$:= $\overline{e}$; S; y := α (r)
- Supongamos que hemos demostrado {Q} S {R}. Para demostrar:
  - {G} y := α(f($\overline{e}$)) {H}
- debemos realizar los siguientes pasos:
  - (1) $G \Rightarrow Q^{\overline{X}}_{\overline{e}}$
  - (2) $G \wedge R^{\overline{X}}_{\overline{e}} \Rightarrow H^{y}_{\alpha(r)}$
- Ejemplo: Demostrar
  - {a > b $\wedge$ c > 0} b:= c * abs(a – b) {b > 0}

Tema 6: GCL
Procedimientos

- **Procedimientos**

```
proc p( X; var r)
{Q} S {R}
```

- Por simplicidad, supondremos que
  - En Q sólo aparecen libres las $\overline{X}$
  - En R sólo aparecen libres las $\overline{X}$ y $\overline{r}$
  - Las variables internas de **p** no se usan en el resto del programa
  - Las variables $\overline{r}$ funcionan por copia local. Es decir, la llamada p($\overline{e}$, $\overline{y}$) equivale a:
    - $\overline{X}$, $\overline{r}$:= $\overline{e}$, $\overline{y}$; S; $\overline{y}$:= $\overline{r}$

Tema 6: GCL
Procedimientos

- **Ejemplo**

```
proc incr(X; var r)
{Q: r = Z}
 r := r + X;
{R: r = Z + X}
```

- Supongamos que hemos demostrado {Q} S {R}. Para demostrar
  - {G} p($\overline{e}$, $\overline{y}$) {H}
- debemos realizar los siguientes pasos:
  - (1) $G \Rightarrow Q^{\overline{x}, \overline{r}}_{\overline{e}, \overline{y}}$
  - (2) $G \wedge R^{\overline{x}}_{\overline{e}} \Rightarrow H^{\overline{y}}_{\overline{r}}$
- Ejemplo: Demostrar {a ≥0 ∧ b=B} incr(a+1,b) {b > B}

Tema 6: GCL
Recursividad

- **Definición**
  - La función (o procedimiento) *f* es recursiva si su ejecución puede producir una nueva llamada a *f*
- Ejemplo

```
func fact(X) ret r
if X = 0 → r := 1
|  X > 0 → r := X * fact(X – 1)
fi
```

- Recursividad directa: la llamada a *f* se produce en el cuerpo de *f*.
  - Ejemplo: *fact*

Tema 6: GCL
Recursividad

- Recursividad indirecta: *f* llama a otra función g cuya ejecución acaba finalmente llamando a *f* de vuelta.
  - Ejemplo:

```
func par(X) ret r:boolean
if X = 0 → r := T
|  X > 0 → r := impar(X – 1)
fi
```

```
func impar(X) ret r:boolean
if X = 0 → r := F
|  X > 0 → r := par(X – 1)
fi
```


Tema 6: GCL
Recursividad directa

- Simple o lineal: cada ejecución del cuerpo de *f* genera como mucho una nueva llamada a *f* (a primer nivel).
  - Ejemplo: *fact*
- Múltiple o no lineal: ejecutar el cuerpo de *f* puede suponer hacer uso de *f* dos o más veces.
  - Ejemplo:

```
func fibo(X) ret r
if X = 0 → r := 0
|  X = 1 → r := 1
|  X > 1 → a := fibo(X – 1);
          b := fibo(X – 2);
          r := a + b
fi
```

Tema 6: GCL
Recursividad directa lineal

- Final: la llamada es la última instrucción ejecutada.
  - Ejemplo: *fact*
- Ejemplo de No final:

```
func max(B[0:N-1],i,d) ret r
if i < d → r := max(B,i+1,d)
|  i = d → r := B[i]
fi;
if B[i] > r → r := B[i]
|  B[i] ≤ r → skip
fi
```