Paradigmas Fundamentales de Programación

Semántica del lenguaje Núcleo

Juan Francisco Díaz Frias

Maestría en Ingeniería, énfasis en Ingeniería de Sistemas y Computación
Universidad del Valle, Cali, Colombia
Escuela de Ingeniería de Sistemas y Computación,
home page: http://eisc.univalle.edu.co/
juanfco.diaz@correounivalle.edu.co

Modelos y Paradigmas de Programación

Universidad del Valle

Plan

Plan

Plan

Modelos y Paradigmas de Programación

Universidad del Valle

Plan

Plan

Plan

Identificadores de variables y alcance léxico

Alcance léxico

- El significado de `X` es determinado por la declaración `local` más interna que declara a `X`.
- El área del programa donde `X` conserva ese significado se llama el alcance de `X`.
- Dos tipos de alcance: léxico y dinámico. El más usado es el alcance léxico.
- El significado de un identificador se puede determinar mirando sólo un pedazo del programa.

Ejemplo

```
local X in
   X=1
   local X in
      X=2
      {Browse X}
   end
   {Browse X}
end
```

- ¿Qué se ve en el browser?

Identificadores de variables y alcance léxico

Alcance léxico

- El significado de `X` es determinado por la declaración `local` más interna que declara a `X`.
- El área del programa donde `X` conserva ese significado se llama el alcance de `X`.
- Dos tipos de alcance: léxico y dinámico. El más usado es el alcance léxico.
- El significado de un identificador se puede determinar mirando sólo un pedazo del programa.

Ejemplo

```
local X in
   X=1
   local X in
      X=2
      {Browse X}
   end
   {Browse X}
end
```

- ¿Qué se ve en el browser?

Identificadores de variables y alcance léxico

Alcance léxico

- El significado de X es determinado por la declaración **local** más interna que declara a X.
- El área del programa donde X conserva ese significado se llama el alcance de X.
- Dos tipos de alcance: léxico y dinámico. El más usado es el alcance léxico.
- El significado de un identificador se puede determinar mirando sólo un pedazo del programa.

Ejemplo

```
local X in
   X=1
   local X in
      X=2
      {Browse X}
   end
   {Browse X}
end
```

- ¿Qué se ve en el browser?
- Hay un solo identificador, pero en diferentes puntos de la ejecución, referencia diferentes variables.

Identificadores de variables y alcance léxico

Alcance léxico

- El significado de `X` es determinado por la declaración **`local`** más interna que declara a `X`.
- El área del programa donde `X` conserva ese significado se llama el alcance de `X`.
- Dos tipos de alcance: léxico y dinámico. El más usado es el alcance léxico.
- El significado de un identificador se puede determinar mirando sólo un pedazo del programa.

Ejemplo

```
local X in
   X=1
   local X in
      X=2
      {Browse X}
   end
   {Browse X}
end
```

- ¿Qué se ve en el browser?
- Hay un solo identificador, pero en diferentes puntos de la ejecución, referencia diferentes variables.

Identificadores de variables y alcance léxico

Alcance léxico

- El significado de `X` es determinado por la declaración **`local`** más interna que declara a `X`.
- El área del programa donde `X` conserva ese significado se llama el alcance de `X`.
- Dos tipos de alcance: léxico y dinámico. El más usado es el alcance léxico.
- El significado de un identificador se puede determinar mirando sólo un pedazo del programa.

Ejemplo

```
local X in
   X=1
   local X in
      X=2
      {Browse X}
   end
   {Browse X}
end
```

- ¿Qué se ve en el browser?
- Hay un solo identificador, pero en diferentes puntos de la ejecución, referencia diferentes variables.

Identificadores de variables y alcance léxico

Alcance léxico

- El significado de `X` es determinado por la declaración **`local`** más interna que declara a `X`.
- El área del programa donde `X` conserva ese significado se llama el alcance de `X`.
- Dos tipos de alcance: léxico y dinámico. El más usado es el alcance léxico.
- El significado de un identificador se puede determinar mirando sólo un pedazo del programa.

Ejemplo

```
local X in
   X=1
   local X in
      X=2
      {Browse X}
   end
   {Browse X}
end
```

- ¿Qué se ve en el browser?
- Hay un solo identificador, pero en diferentes puntos de la ejecución, referencia diferentes variables.

Procedimientos

Sin referencias externas

```
proc {Max X Y ?Z}
   if X>=Y then Z=X
           else Z=Y end
end
```

- "?" es solo un comentario.
- {Max 3 5 C} liga C a 5. ¿Cómo?
- X, Y, y Z se ligan a 3, 5, y a la variable no-ligada referenciada por C.
- Paso de parámetros por referencia.

Con referencias externas

- Considere

```
if X>=Y then Z=X
        else Z=Y end
```

¡no puede ser ejecutada! X, Y, y Z, son libres.

Procedimientos

Sin referencias externas

```
proc {Max X Y ?Z}
   if X>=Y then Z=X
           else Z=Y end
end
```

- "?" es solo un comentario.
- {Max 3 5 C} liga C a 5. ¿Cómo?
- X, Y, y Z se ligan a 3, 5, y a la variable no-ligada referenciada por C.
- Paso de parámetros por referencia.

Con referencias externas

- Considere

```
if X>=Y then Z=X
        else Z=Y end
```

¡no puede ser ejecutada! X, Y, y Z, son libres.

Procedimientos

Sin referencias externas

```
proc {Max X Y ?Z}
   if X>=Y then Z=X
           else Z=Y end
end
```

- "?" es solo un comentario.
- {Max 3 5 C} liga C a 5. ¿Cómo?
- X, Y, y Z se ligan a 3, 5, y a la variable no-ligada referenciada por C.
- Paso de parámetros por referencia.

Con referencias externas

- Considere

```
if X>=Y then Z=X
        else Z=Y end
```

¡no puede ser ejecutada! X, Y, y Z, son libres.

¿Con que valor queda ligada Z?

Procedimientos

Sin referencias externas

```
proc {Max X Y ?Z}
   if X>=Y then Z=X
           else Z=Y end
end
```

- "?" es solo un comentario.
- `{Max 3 5 C}` liga `C` a 5. ¿Cómo?
- `X`, `Y`, y `Z` se ligan a 3, 5, y a la variable no-ligada referenciada por `C`.
- Paso de parámetros por referencia.

Con referencias externas

- Considere

```
if X>=Y then Z=X
        else Z=Y end
```

¡no puede ser ejecutada! `X`, `Y`, y `Z`, son libres.

- ¿Con qué valor queda ligado `Z`?

```
local Y=10 LB in
   proc {LB X ?Z}
      if X>=Y then Z=X
              else Z=Y
      end
   end
   local Y=15 Z in
      {LB 5 Z}
   end
end
```

Procedimientos

Sin referencias externas

```
proc {Max X Y ?Z}
   if X>=Y then Z=X
           else Z=Y end
end
```

- "?" es solo un comentario.
- `{Max 3 5 C}` liga `C` a 5. ¿Cómo?
- `X`, `Y`, y `Z` se ligan a 3, 5, y a la variable no-ligada referenciada por `C`.
- Paso de parámetros por referencia.

Con referencias externas

- Considere

```
if X>=Y then Z=X
        else Z=Y end
```

¡no puede ser ejecutada! `X`, `Y`, y `Z`, son libres.

- ¿Con qué valor queda ligado `Z` ?

```
local Y=10 LB in
   proc {LB X ?Z}
      if X>=Y then Z=X
              else Z=Y
      end
   end
   local Y=15 Z in
      {LB 5 Z}
   end
end
```

Procedimientos

Sin referencias externas

```
proc {Max X Y ?Z}
   if X>=Y then Z=X
           else Z=Y end
end
```

- "?" es solo un comentario.
- `{Max 3 5 C}` liga `C` a 5. ¿Cómo?
- `X`, `Y`, y `Z` se ligan a 3, 5, y a la variable no-ligada referenciada por `C`.
- Paso de parámetros por referencia.

Con referencias externas

- Considere

```
if X>=Y then Z=X
        else Z=Y end
```

¡no puede ser ejecutada! `X`, `Y`, y `Z`, son libres.

- ¿Con qué valor queda ligado `Z` ?

```
local Y=10 LB in
   proc {LB X ?Z}
      if X>=Y then Z=X
              else Z=Y
      end
   end
   local Y=15 Z in
      {LB 5 Z}
   end
end
```

Plan

Alcance dinámico vs. Alcance estático

Ejemplo

- Considere el programa:

```
local P Q in
   proc {Q X}
    {Browse estat(X)}
   end
   proc {P X} {Q X} end
   local Q in
      proc {Q X}
       {Browse din(X)}
      end
      {P hola}
   end
end
```

¿Qué debería mostrar en pantalla?

- Alcance estático: `estat(hola)`
- Alcance dinámico: `din(hola)`

¿Cuál es el comportamiento correcto por defecto?

- Estático: porque el procedimiento funciona independientemente del ambiente donde haya sido definido.

Alcance dinámico vs. Alcance estático

Ejemplo

- Considere el programa:

```
local P Q in
   proc {Q X}
    {Browse estat(X)}
   end
   proc {P X} {Q X} end
   local Q in
      proc {Q X}
       {Browse din(X)}
      end
      {P hola}
   end
end
```

¿Qué debería mostrar en pantalla?

- Alcance estático: `estat(hola)`
- Alcance dinámico: `din(hola)`

¿Cuál es el comportamiento correcto por defecto?

- Estático: porque el procedimiento funciona independientemente del ambiente donde haya sido definido.

Alcance dinámico vs. Alcance estático

Ejemplo

- Considere el programa:

```
local P Q in
   proc {Q X}
    {Browse estat(X)}
   end
   proc {P X} {Q X} end
   local Q in
      proc {Q X}
       {Browse din(X)}
      end
      {P hola}
   end
end
```

¿Qué debería mostrar en pantalla?

- Alcance estático: `estat(hola)`
- Alcance dinámico: `din(hola)`

¿Cuál es el comportamiento correcto por defecto?

- Estático: porque el procedimiento funciona independientemente del ambiente donde haya sido definido.
- Cuando usar Dinámico: procedimientos que se ejecutan en máquinas remotas.

Alcance dinámico vs. Alcance estático

Ejemplo

- Considere el programa:

```
local P Q in
   proc {Q X}
    {Browse estat(X)}
   end
   proc {P X} {Q X} end
   local Q in
      proc {Q X}
       {Browse din(X)}
      end
      {P hola}
   end
end
```

¿Qué debería mostrar en pantalla?

- Alcance estático: `estat(hola)`
- Alcance dinámico: `din(hola)`

¿Cuál es el comportamiento correcto por defecto?

- Estático: porque el procedimiento funciona independientemente del ambiente donde haya sido definido.
- Cuando usar Dinámico: procedimientos que se ejecutan en máquinas remotas.

Alcance dinámico vs. Alcance estático

Ejemplo

- Considere el programa:

```
local P Q in
   proc {Q X}
    {Browse estat(X)}
   end
   proc {P X} {Q X} end
   local Q in
      proc {Q X}
       {Browse din(X)}
      end
      {P hola}
   end
end
```

¿Qué debería mostrar en pantalla?

- Alcance estático: `estat(hola)`
- Alcance dinámico: `din(hola)`

¿Cuál es el comportamiento correcto por defecto?

- Estático: porque el procedimiento funciona independientemente del ambiente donde haya sido definido.
- Cuando usar Dinámico: procedimientos que se ejecutan en máquinas remotas.

Plan

La máquina abstracta: componentes

Almacén de asignación única

- σ es un conjunto de variables del almacén.
- $\sigma = \{x_1, x_2 = x_3, x_4 = \mathtt{a} \mid x_2\}$

Ambiente

- E es una asociación de identificadores de variable en entidades de σ.
- $E = \{\mathtt{x} \rightarrow x, \mathtt{y} \rightarrow y \ldots\}$,

Declaración semántica

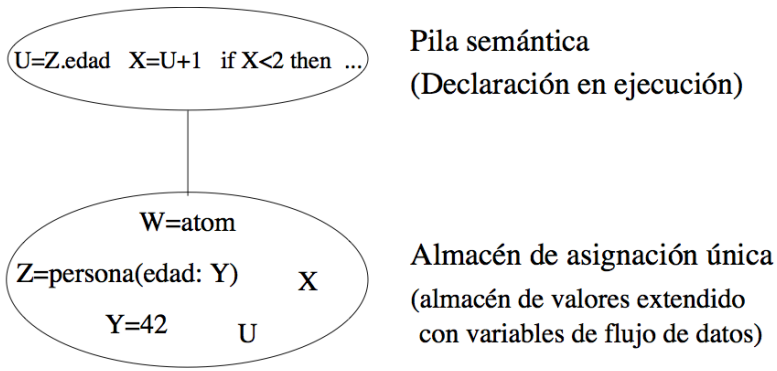
$(\langle \mathrm{d} \rangle, E)$

Estado de ejecución

(ST, σ) donde ST es una pila de declaraciones semánticas.

Computación

secuencia de estados de ejecución, a partir de un estado inicial: $(ST_0, \sigma_0) \rightarrow (ST_1, \sigma_1) \rightarrow (ST_2, \sigma_2) \rightarrow \ldots$.



La máquina abstracta: componentes

Almacén de asignación única

- σ es un conjunto de variables del almacén.
- $\sigma = \{x_1, x_2 = x_3, x_4 = \mathtt{a} \mid x_2\}$

Ambiente

- E es una asociación de identificadores de variable en entidades de σ.
- $E = \{\mathtt{X} \rightarrow x, \mathtt{Y} \rightarrow y \ldots\}$,

Declaración semántica

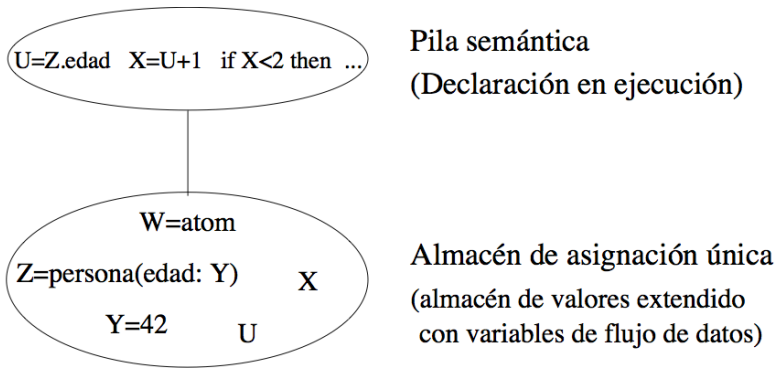
$(\langle \mathrm{d} \rangle, E)$

Estado de ejecución

(ST, σ) donde ST es una pila de declaraciones semánticas.

Computación

secuencia de estados de ejecución, a partir de un estado inicial: $(ST_0, \sigma_0) \rightarrow (ST_1, \sigma_1) \rightarrow (ST_2, \sigma_2) \rightarrow \ldots$.



La máquina abstracta: componentes

Almacén de asignación única

- σ es un conjunto de variables del almacén.
- $\sigma = \{x_1, x_2 = x_3, x_4 = \mathtt{a} \mid x_2\}$

Ambiente

- E es una asociación de identificadores de variable en entidades de σ.
- $E = \{\mathtt{X} \rightarrow x, \mathtt{Y} \rightarrow y \ldots\}$,

Declaración semántica

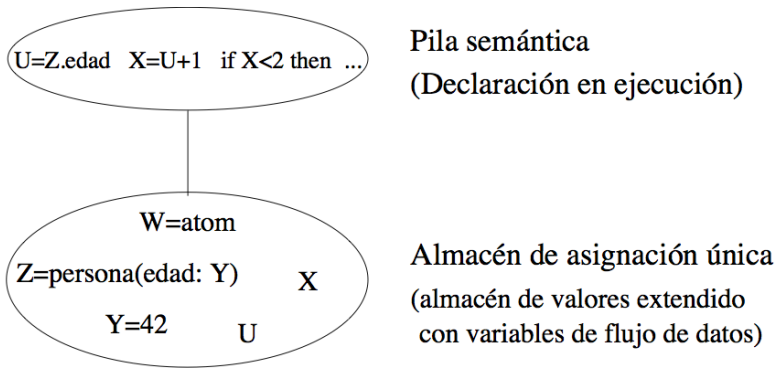
$(\langle \mathrm{d} \rangle, E)$

Estado de ejecución

(ST, σ) donde ST es una pila de declaraciones semánticas.

Computación

secuencia de estados de ejecución, a partir de un estado inicial: $(ST_0, \sigma_0) \rightarrow (ST_1, \sigma_1) \rightarrow (ST_2, \sigma_2) \rightarrow \cdots$.



La máquina abstracta: componentes

Almacén de asignación única

- σ es un conjunto de variables del almacén.
- $\sigma = \{x_1, x_2 = x_3, x_4 = \mathtt{a} \mid x_2\}$

Ambiente

- E es una asociación de identificadores de variable en entidades de σ.
- $E = \{\mathtt{X} \rightarrow x, \mathtt{Y} \rightarrow y \ldots\}$,

Declaración semántica

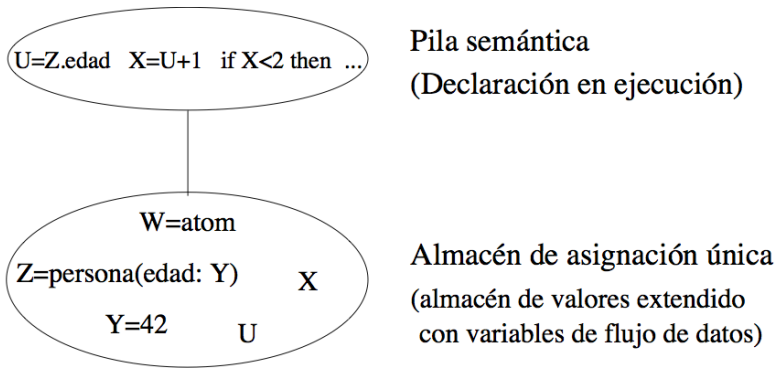
$(\langle \mathrm{d} \rangle, E)$

Estado de ejecución

(ST, σ) donde ST es una pila de declaraciones semánticas.

Computación

secuencia de estados de ejecución, a partir de un estado inicial: $(ST_0, \sigma_0) \rightarrow (ST_1, \sigma_1) \rightarrow (ST_2, \sigma_2) \rightarrow \cdots$.



La máquina abstracta: componentes

Almacén de asignación única

- σ es un conjunto de variables del almacén.
- $\sigma = \{x_1, x_2 = x_3, x_4 = \mathtt{a} \mid x_2\}$

Ambiente

- E es una asociación de identificadores de variable en entidades de σ.
- $E = \{\mathtt{X} \rightarrow x, \mathtt{Y} \rightarrow y \ldots\}$,

Declaración semántica

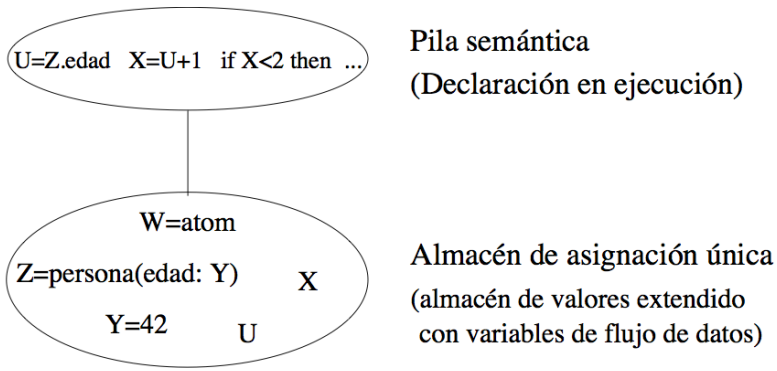
$(\langle \mathrm{d} \rangle, E)$

Estado de ejecución

(ST, σ) donde ST es una pila de declaraciones semánticas.

Computación

secuencia de estados de ejecución, a partir de un estado inicial: $(ST_0, \sigma_0) \rightarrow (ST_1, \sigma_1) \rightarrow (ST_2, \sigma_2) \rightarrow \cdots$.



Ejecución

Ejecutar una declaración $\langle d \rangle$

- Estado inicial:

$$([(\langle d \rangle, \emptyset)], \emptyset)$$

- Se toma el primer elemento de *ST* y se procede con su ejecución.
- Estado final: pila semántica vacía.

Estados de la pila semántica

Ejecución

Ejecutar una declaración $\langle d \rangle$

- Estado inicial:

$$([(\langle d \rangle, \emptyset)], \emptyset)$$

- Se toma el primer elemento de *ST* y se procede con su ejecución.
- Estado final: pila semántica vacía.

Estados de la pila semántica

- Ejecutable: *ST* puede realizar una etapa de computación.
- Terminado: *ST* está vacía.
- Estado inválido.

Ejecución

Ejecutar una declaración $\langle d \rangle$

- Estado inicial:

$$([(\langle d \rangle, \emptyset)], \emptyset)$$

- Se toma el primer elemento de *ST* y se procede con su ejecución.
- Estado final: pila semántica vacía.

Estados de la pila semántica

- Ejecutable: *ST* puede realizar una etapa de computación.
- Terminado: *ST* está vacía.
- Suspendido: *ST* no esta vacía, pero tampoco puede realizar una etapa de computación.

Ejecución

Ejecutar una declaración $\langle d \rangle$

- Estado inicial:

$$([(\langle d \rangle, \emptyset)], \emptyset)$$

- Se toma el primer elemento de *ST* y se procede con su ejecución.
- Estado final: pila semántica vacía.

Estados de la pila semántica

- Ejecutable: *ST* puede realizar una etapa de computación.
- Terminado: *ST* está vacía.
- Suspendido: *ST* no está vacía, pero tampoco puede realizar una etapa de computación.

Ejecución

Ejecutar una declaración $\langle d \rangle$

- Estado inicial:

$$([(\langle d \rangle, \emptyset)], \emptyset)$$

- Se toma el primer elemento de *ST* y se procede con su ejecución.
- Estado final: pila semántica vacía.

Estados de la pila semántica

- Ejecutable: *ST* puede realizar una etapa de computación.
- Terminado: *ST* está vacía.
- Suspendido: *ST* no está vacía, pero tampoco puede realizar una etapa de computación.

Ejecución

Ejecutar una declaración $\langle d \rangle$

- Estado inicial:

$$([(\langle d \rangle, \emptyset)], \emptyset)$$

- Se toma el primer elemento de *ST* y se procede con su ejecución.
- Estado final: pila semántica vacía.

Estados de la pila semántica

- Ejecutable: *ST* puede realizar una etapa de computación.
- Terminado: *ST* está vacía.
- Suspendido: *ST* no está vacía, pero tampoco puede realizar una etapa de computación.

Plan

Calculando con ambientes

Recordando qué es un ambiente

- E es una función que asocia identificadores de variables $\langle x \rangle$ con entidades del almacén.
- $E(\langle x \rangle)$ representa la entidad del almacén asociada con el identificador $\langle x \rangle$ en el ambiente E.

Restricción

$$E|_{\{\langle x \rangle_1, \ldots, \langle x \rangle_n\}}$$

denota un ambiente nuevo E' tal que
$\mathrm{dom}(E') = \mathrm{dom}(E) \cap \{\langle x \rangle_1, \ldots, \langle x \rangle_n\}$
y $E'(\langle x \rangle) = E(\langle x \rangle)$ para todo $\langle x \rangle \in \mathrm{dom}(E')$.

Extensión

$$E' = E + \{\langle x \rangle \to x\}$$

- $E'(\langle x \rangle) = x$, y $E'(\langle y \rangle) = E(\langle y \rangle)$ para $\langle y \rangle \neq \langle x \rangle$.
- $E + \{\langle x \rangle_1 \to x_1, \ldots, \langle x \rangle_n \to x_n\}$.

Calculando con ambientes

Recordando qué es un ambiente

- E es una función que asocia identificadores de variables $\langle x \rangle$ con entidades del almacén.
- $E(\langle x \rangle)$ representa la entidad del almacén asociada con el identificador $\langle x \rangle$ en el ambiente E.

Restricción

$$E|_{\{\langle x \rangle_1, \ldots, \langle x \rangle_n\}}$$

denota un ambiente nuevo E' tal que
$\mathrm{dom}(E') = \mathrm{dom}(E) \cap \{\langle x \rangle_1, \ldots, \langle x \rangle_n\}$
y $E'(\langle x \rangle) = E(\langle x \rangle)$ para todo $\langle x \rangle \in \mathrm{dom}(E')$.

Extensión

- $$E' = E + \{\langle x \rangle \to x\}$$
- $E'(\langle x \rangle) = x$, y $E'(\langle y \rangle) = E(\langle y \rangle)$ para $\langle y \rangle \neq \langle x \rangle$.
- $E + \{\langle x \rangle_1 \to x_1, \ldots, \langle x \rangle_n \to x_n\}$.

Calculando con ambientes

Recordando qué es un ambiente

- E es una función que asocia identificadores de variables $\langle x \rangle$ con entidades del almacén.
- $E(\langle x \rangle)$ representa la entidad del almacén asociada con el identificador $\langle x \rangle$ en el ambiente E.

Restricción

$$E|_{\{\langle x \rangle_1, \ldots, \langle x \rangle_n\}}$$

denota un ambiente nuevo E' tal que
$\text{dom}(E') = \text{dom}(E) \cap \{\langle x \rangle_1, \ldots, \langle x \rangle_n\}$
y $E'(\langle x \rangle) = E(\langle x \rangle)$ para todo $\langle x \rangle \in \text{dom}(E')$.

Extensión

-

$$E' = E + \{\langle x \rangle \to x\}$$

- $E'(\langle x \rangle) = x$, y $E'(\langle y \rangle) = E(\langle y \rangle)$ para $\langle y \rangle \neq \langle x \rangle$.
- $E + \{\langle x \rangle_1 \to x_1, \ldots, \langle x \rangle_n \to x_n\}$.

Plan

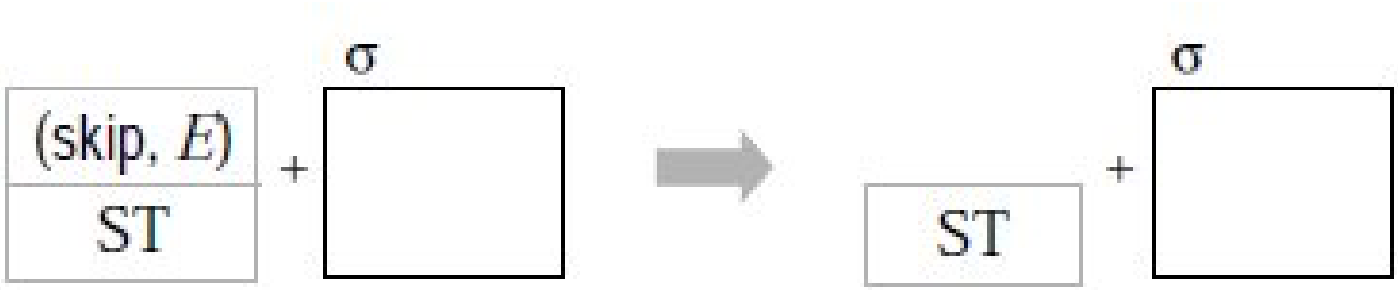
La declaración `skip`

La declaración semántica es:

$$(\texttt{skip}, E)$$

La ejecución se completa con sólo tomar la pareja de la pila semántica.

Gráficamente



La Composición Secuencial

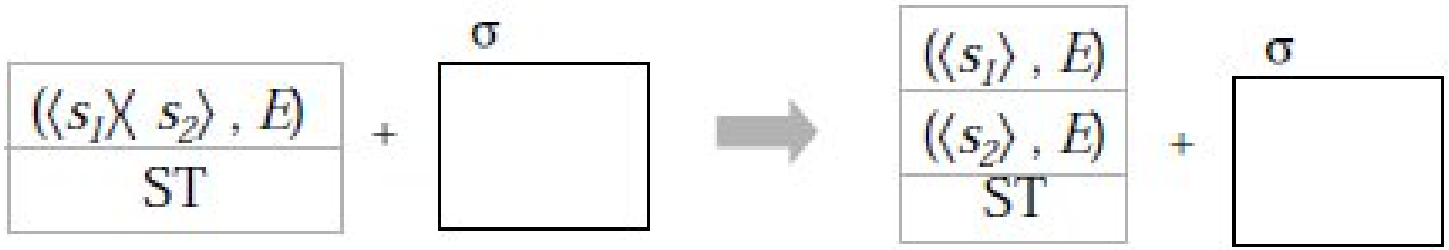
La declaración semántica es:

$$(\langle d \rangle_1 \langle d \rangle_2, E)$$

La ejecución consiste de las acciones siguientes:

- Colocar $(\langle d \rangle_2, E)$ en la pila.
- Colocar $(\langle d \rangle_1, E)$ en la pila.

Gráficamente



Declaración de variable (la declaración `local`) (1)

La declaración semántica es:

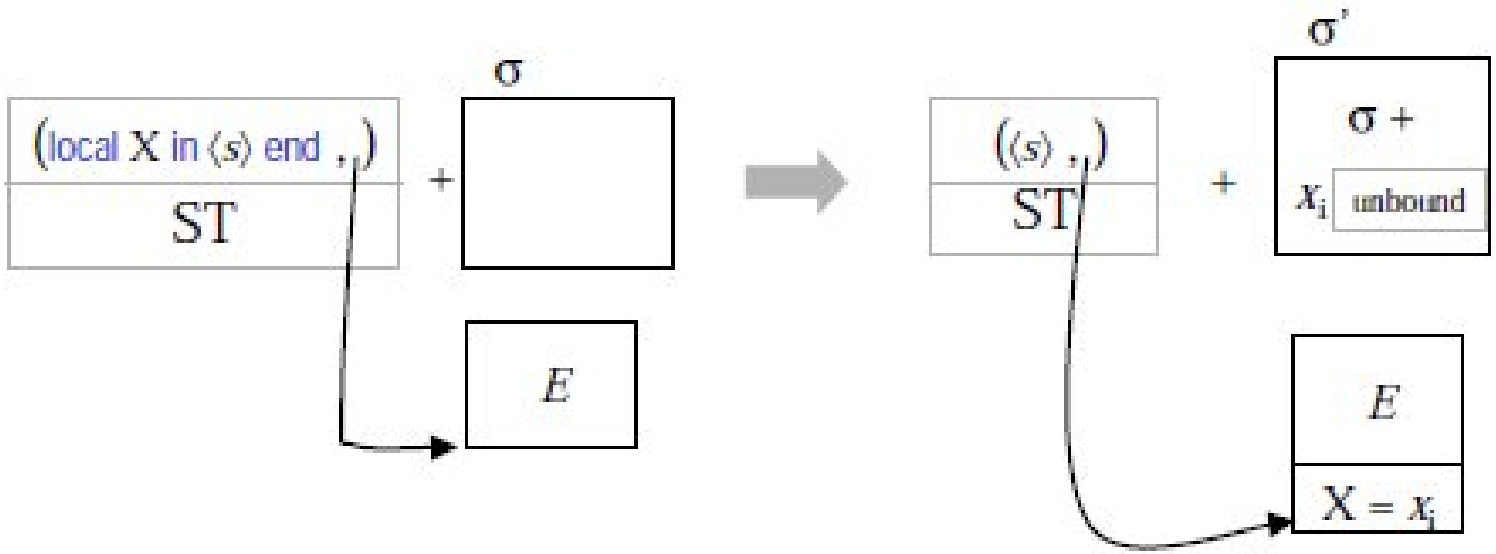
$$(\texttt{local}\ \langle x \rangle\ \texttt{in}\ \langle d \rangle\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Crear una variable nueva, x, en el almacén.
- Calcule E' como $E + \{\langle \mathrm{x} \rangle \rightarrow x\}$,
- Coloque $(\langle \mathrm{d} \rangle, E')$ en la pila.

Declaración de variable (la declaración `local`)(2)

Gráficamente



Ligadura variable-variable, variable-valor

Variable-Variable

La declaración semántica es:

$$(\langle x \rangle_1 = \langle x \rangle_2, E)$$

La ejecución consiste de la acción siguiente:

- Ligue $E(\langle x \rangle_1)$ y $E(\langle x \rangle_2)$ en el almacén.

Variable-Valor

La declaración semántica es:

$$(\langle x \rangle = \langle v \rangle, E)$$

donde $\langle v \rangle$ es un valor parcialmente construido de tipo registro, número, o procedimiento. La ejecución consiste de las acciones siguientes:

- Crear una variable nueva, x, en el almacén.
- Construir el valor representado por $\langle v \rangle$ en el almacén y hacer que x lo referencie. Todos los identificadores de $\langle v \rangle$ se reemplazan por sus contenidos en el almacén, de acuerdo al ambiente E.
- Ligar $E(\langle x \rangle)$ y x en el almacén.

Construcción de valores

Números y registros: claro!

¿Y los procedimientos?

Ligadura variable-variable, variable-valor

Variable-Variable

La declaración semántica es:

$$(\langle x \rangle_1 = \langle x \rangle_2, E)$$

La ejecución consiste de la acción siguiente:

- Ligue $E(\langle x \rangle_1)$ y $E(\langle x \rangle_2)$ en el almacén.

Construcción de valores
Números y registros: claro!
¿Y los procedimientos?

Variable-Valor

La declaración semántica es:

$$(\langle x \rangle = \langle v \rangle, E)$$

donde $\langle v \rangle$ es un valor parcialmente construido de tipo registro, número, o procedimiento. La ejecución consiste de las acciones siguientes:

- Crear una variable nueva, x, en el almacén.
- Construir el valor representado por $\langle v \rangle$ en el almacén y hacer que x lo referencie. Todos los identificadores de $\langle v \rangle$ se reemplazan por sus contenidos en el almacén, de acuerdo al ambiente E.
- Ligar $E(\langle x \rangle)$ y x en el almacén.

Ligadura variable-variable, variable-valor

Variable-Variable

La declaración semántica es:

$$(\langle x \rangle_1 = \langle x \rangle_2, E)$$

La ejecución consiste de la acción siguiente:

- Ligue $E(\langle x \rangle_1)$ y $E(\langle x \rangle_2)$ en el almacén.

Construcción de valores
Números y registros: claro!
¿Y los procedimientos?

Variable-Valor

La declaración semántica es:

$$(\langle x \rangle = \langle v \rangle, E)$$

donde $\langle v \rangle$ es un valor parcialmente construido de tipo registro, número, o procedimiento. La ejecución consiste de las acciones siguientes:

- Crear una variable nueva, x, en el almacén.
- Construir el valor representado por $\langle v \rangle$ en el almacén y hacer que x lo referencie. Todos los identificadores de $\langle v \rangle$ se reemplazan por sus contenidos en el almacén, de acuerdo al ambiente E.
- Ligar $E(\langle x \rangle)$ y x en el almacén.

Ligadura variable-variable, variable-valor

Variable-Variable

La declaración semántica es:

$$(\langle x \rangle_1 = \langle x \rangle_2, E)$$

La ejecución consiste de la acción siguiente:

- Ligue $E(\langle x \rangle_1)$ y $E(\langle x \rangle_2)$ en el almacén.

Construcción de valores

Números y registros: claro!

¿Y los procedimientos?

Variable-Valor

La declaración semántica es:

$$(\langle x \rangle = \langle v \rangle, E)$$

donde $\langle v \rangle$ es un valor parcialmente construido de tipo registro, número, o procedimiento. La ejecución consiste de las acciones siguientes:

- Crear una variable nueva, x, en el almacén.
- Construir el valor representado por $\langle v \rangle$ en el almacén y hacer que x lo referencie. Todos los identificadores de $\langle v \rangle$ se reemplazan por sus contenidos en el almacén, de acuerdo al ambiente E.
- Ligar $E(\langle x \rangle)$ y x en el almacén.

Modelos y Paradigmas de Programación

Universidad del Valle

Plan

Representación interna de los procedimientos (1)

Ocurrencias libres y ligadas de identificadores

- Una declaración ⟨d⟩ puede contener múltiples ocurrencias de identificadores de variables.
- Dado un identificador: ¿Dónde fue definido?
- Si la definición está en alguna declaración (dentro o no de ⟨d⟩) que rodea (i.e. encierra) textualmente la ocurrencia, entonces decimos que la definición obedece el alcance léxico (o alcance estático).
- Una ocurrencia de un identificador x está ligada con respecto a ⟨d⟩ si x está definido dentro de ⟨d⟩ (en una declaración `local`, en el patrón de una declaración `case`, o como argumento de una declaración de un procedimiento).
- Una ocurrencia de un identificador x está libre si no está ligada.
- En un programa ejecutable, toda ocurrencia de un identificador está ligada con respecto al programa completo.

Representación interna de los procedimientos (1)

Ocurrencias libres y ligadas de identificadores

- Una declaración ⟨d⟩ puede contener múltiples ocurrencias de identificadores de variables.
- Dado un identificador: ¿Dónde fue definido?
- Si la definición está en alguna declaración (dentro o no de ⟨d⟩) que rodea (i.e. encierra) textualmente la ocurrencia, entonces decimos que la definición obedece el alcance léxico (o alcance estático).
- Una ocurrencia de un identificador x está ligada con respecto a ⟨d⟩ si x está definido dentro de ⟨d⟩ (en una declaración `local`, en el patrón de una declaración `case`, o como argumento de una declaración de un procedimiento).
- Una ocurrencia de un identificador x está libre si no está ligada.
- En un programa ejecutable, toda ocurrencia de un identificador está ligada con respecto al programa completo.

Representación interna de los procedimientos (1)

Ocurrencias libres y ligadas de identificadores

- Una declaración ⟨d⟩ puede contener múltiples ocurrencias de identificadores de variables.
- Dado un identificador: ¿Dónde fue definido?
- Si la definición está en alguna declaración (dentro o no de ⟨d⟩) que rodea (i.e. encierra) textualmente la ocurrencia, entonces decimos que la definición obedece el alcance léxico (o alcance estático).
- Una ocurrencia de un identificador x está ligada con respecto a ⟨d⟩ si x está definido dentro de ⟨d⟩ (en una declaración `local`, en el patrón de una declaración `case`, o como argumento de una declaración de un procedimiento).
- Una ocurrencia de un identificador x está libre si no está ligada.
- En un programa ejecutable, toda ocurrencia de un identificador está ligada con respecto al programa completo.

Representación interna de los procedimientos (1)

Ocurrencias libres y ligadas de identificadores

- Una declaración ⟨d⟩ puede contener múltiples ocurrencias de identificadores de variables.
- Dado un identificador: ¿Dónde fue definido?
- Si la definición está en alguna declaración (dentro o no de ⟨d⟩) que rodea (i.e. encierra) textualmente la ocurrencia, entonces decimos que la definición obedece el alcance léxico (o alcance estático).
- Una ocurrencia de un identificador x está ligada con respecto a ⟨d⟩ si x está definido dentro de ⟨d⟩ (en una declaración `local`, en el patrón de una declaración `case`, o como argumento de una declaración de un procedimiento).
- Una ocurrencia de un identificador x está libre si no está ligada.
- En un programa ejecutable, toda ocurrencia de un identificador está ligada con respecto al programa completo.

Representación interna de los procedimientos (1)

Ocurrencias libres y ligadas de identificadores

- Una declaración ⟨d⟩ puede contener múltiples ocurrencias de identificadores de variables.
- Dado un identificador: ¿Dónde fue definido?
- Si la definición está en alguna declaración (dentro o no de ⟨d⟩) que rodea (i.e. encierra) textualmente la ocurrencia, entonces decimos que la definición obedece el alcance léxico (o alcance estático).
- Una ocurrencia de un identificador x está ligada con respecto a ⟨d⟩ si x está definido dentro de ⟨d⟩ (en una declaración `local`, en el patrón de una declaración `case`, o como argumento de una declaración de un procedimiento).
- Una ocurrencia de un identificador x está libre si no está ligada.
- En un programa ejecutable, toda ocurrencia de un identificador está ligada con respecto al programa completo.

Representación interna de los procedimientos (1)

Ocurrencias libres y ligadas de identificadores

- Una declaración ⟨d⟩ puede contener múltiples ocurrencias de identificadores de variables.
- Dado un identificador: ¿Dónde fue definido?
- Si la definición está en alguna declaración (dentro o no de ⟨d⟩) que rodea (i.e. encierra) textualmente la ocurrencia, entonces decimos que la definición obedece el alcance léxico (o alcance estático).
- Una ocurrencia de un identificador x está ligada con respecto a ⟨d⟩ si x está definido dentro de ⟨d⟩ (en una declaración `local`, en el patrón de una declaración `case`, o como argumento de una declaración de un procedimiento).
- Una ocurrencia de un identificador x está libre si no está ligada.
- En un programa ejecutable, toda ocurrencia de un identificador está ligada con respecto al programa completo.

Representación interna de los procedimientos (2)

Declaración no ejecutable

```
local Arg1 Arg2 in
   Arg1=111*111
   Arg2=999*999
   Res=Arg1+Arg2
end
```

Todas las ocurrencias de `Arg1` y `Arg2` están ligadas y `Res` está libre.

Declaración ejecutable

```
local Res in
   local Arg1 Arg2 in
      Arg1=111*111
      Arg2=999*999
      Res=Arg1+Arg2
   end
   {Browse Res}
end
```

¿La ocurrencia de `Browse` es libre?

Representación interna de los procedimientos (2)

Declaración no ejecutable

```
local Arg1 Arg2 in
   Arg1=111*111
   Arg2=999*999
   Res=Arg1+Arg2
end
```

Todas las ocurrencias de `Arg1` y `Arg2` están ligadas y `Res` está libre.

Declaración ejecutable

```
local Res in
   local Arg1 Arg2 in
      Arg1=111*111
      Arg2=999*999
      Res=Arg1+Arg2
   end
   {Browse Res}
end
```

¿La ocurrencia de `Browse` es libre?

Representación interna de los procedimientos (3)

¿Qué hacer con las referencias libres en los procedimientos?

```
proc {LowerBound X ?Z}
   if X>=Y then Z=X else Z=Y end
end
```

- `if` tiene seis ocurrencias libres: `X` (dos veces), `Y` (dos veces), y `Z` (dos veces).
- `X` y `Z`, son parámetros formales de `LowerBound`. Su valor estará definido por la invocación.
- Pero, ¿`Y`? Su valor debe definirse al momento de la creacción del procedimiento.

Representación interna de los procedimientos (3)

¿Qué hacer con las referencias libres en los procedimientos?

```
proc {LowerBound X ?Z}
   if X>=Y then Z=X else Z=Y end
end
```

- `if` tiene seis ocurrencias libres: `X` (dos veces), `Y` (dos veces), y `Z` (dos veces).
- `X` y `Z`, son parámetros formales de `LowerBound`. Su valor estará definido por la invocación.
- Pero, ¿`Y`? Su valor debe definirse al momento de la creacción del procedimiento.

Representación interna de los procedimientos (3)

¿Qué hacer con las referencias libres en los procedimientos?

```
proc {LowerBound X ?Z}
   if X>=Y then Z=X else Z=Y end
end
```

- `if` tiene seis ocurrencias libres: `X` (dos veces), `Y` (dos veces), y `Z` (dos veces).
- `X` y `Z`, son parámetros formales de `LowerBound`. Su valor estará definido por la invocación.
- Pero, ¿`Y`? Su valor debe definirse al momento de la creacción del procedimiento.

Representación interna de los procedimientos (4)

Creación de procedimientos

Por medio de la expresión:

$$\texttt{proc}\ \{\ \$\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \}\ \langle d \rangle\ \texttt{end}$$

Observaciones

- $\langle d \rangle$ puede tener ocurrencias libres de identificadores de variables.
- Si corresponde a parámetros formales: su valor se define cada vez que se invoca el procedimiento. Las llamamos $\{\langle y \rangle_1, \ldots, \langle y \rangle_n\}$.
- Las otras, referencias externas, se definen una sola vez y para siempre, en el momento en que se define el procedimiento. Las llamamos $\{\langle z \rangle_1, \ldots, \langle z \rangle_k\}$.

¿Qué es un valor de tipo procedimiento?

Es una pareja (clausura):

$$(\texttt{proc}\ \{\ \$\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \}\ \langle d \rangle\ \texttt{end},\ CE)$$

CE (ambiente contextual) es

$E|_{\{\langle z \rangle_1, \ldots, \langle z \rangle_n\}}$, donde E es el ambiente en el momento en que el procedimiento es definido.

Representación interna de los procedimientos (4)

Observaciones

- $\langle$d$\rangle$ puede tener ocurrencias libres de identificadores de variables.
- Si corresponde a parámetros formales: su valor se define cada vez que se invoca el procedimiento. Las llamamos $\{\langle y \rangle_1, \ldots, \langle y \rangle_n\}$.
- Las otras, referencias externas, se definen una sola vez y para siempre, en el momento en que se define el procedimiento. Las llamamos $\{\langle z \rangle_1, \ldots, \langle z \rangle_k\}$.

Creación de procedimientos

Por medio de la expresión:

$$\texttt{proc}\ \{\ \$\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \}\ \langle d \rangle\ \texttt{end}$$

¿Qué es un valor de tipo procedimiento?

Es una pareja (clausura):

$$(\texttt{proc}\ \{\ \$\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \}\ \langle d \rangle\ \texttt{end},\ CE)$$

CE (ambiente contextual) es $E|_{\{\langle z \rangle_1, \ldots, \langle z \rangle_n\}}$, donde E es el ambiente en el momento en que el procedimiento es definido.

Representación interna de los procedimientos (4)

Observaciones

- $\langle$d$\rangle$ puede tener ocurrencias libres de identificadores de variables.
- Si corresponde a parámetros formales: su valor se define cada vez que se invoca el procedimiento. Las llamamos $\{\langle y\rangle_1, \ldots, \langle y\rangle_n\}$.
- Las otras, referencias externas, se definen una sola vez y para siempre, en el momento en que se define el procedimiento. Las llamamos $\{\langle z\rangle_1, \ldots, \langle z\rangle_k\}$.

Creación de procedimientos

Por medio de la expresión:

$$\texttt{proc}\ \{\ \$\ \langle y\rangle_1 \cdots \langle y\rangle_n\ \}\ \langle d\rangle\ \texttt{end}$$

¿Qué es un valor de tipo procedimiento?

Es una pareja (clausura):

$$(\texttt{proc}\ \{\ \$\ \langle y\rangle_1 \cdots \langle y\rangle_n\ \}\ \langle d\rangle\ \texttt{end},\ CE)$$

CE (ambiente contextual) es $E|_{\{\langle z\rangle_1, \ldots, \langle z\rangle_n\}}$, donde E es el ambiente en el momento en que el procedimiento es definido.

Representación interna de los procedimientos (4)

Observaciones

- $\langle$d$\rangle$ puede tener ocurrencias libres de identificadores de variables.
- Si corresponde a parámetros formales: su valor se define cada vez que se invoca el procedimiento. Las llamamos $\{\langle y\rangle_1, \ldots, \langle y\rangle_n\}$.
- Las otras, referencias externas, se definen una sola vez y para siempre, en el momento en que se define el procedimiento. Las llamamos $\{\langle z\rangle_1, \ldots, \langle z\rangle_k\}$.

Creación de procedimientos

Por medio de la expresión:

$$\texttt{proc}\ \{\ \$\ \langle y\rangle_1 \cdots \langle y\rangle_n\ \}\ \langle d\rangle\ \texttt{end}$$

¿Qué es un valor de tipo procedimiento?

Es una pareja (clausura):

$$(\texttt{proc}\ \{\ \$\ \langle y\rangle_1 \cdots \langle y\rangle_n\ \}\ \langle d\rangle\ \texttt{end},\ CE)$$

CE (ambiente contextual) es $E|_{\{\langle z\rangle_1, \ldots, \langle z\rangle_n\}}$, donde *E* es el ambiente en el momento en que el procedimiento es definido.

Representación interna de los procedimientos (4)

Observaciones

- $\langle d \rangle$ puede tener ocurrencias libres de identificadores de variables.
- Si corresponde a parámetros formales: su valor se define cada vez que se invoca el procedimiento. Las llamamos $\{\langle y \rangle_1, \ldots, \langle y \rangle_n\}$.
- Las otras, referencias externas, se definen una sola vez y para siempre, en el momento en que se define el procedimiento. Las llamamos $\{\langle z \rangle_1, \ldots, \langle z \rangle_k\}$.

Creación de procedimientos

Por medio de la expresión:

$$\texttt{proc } \{\ \$\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \}\ \langle d \rangle\ \texttt{end}$$

¿Qué es un valor de tipo procedimiento?

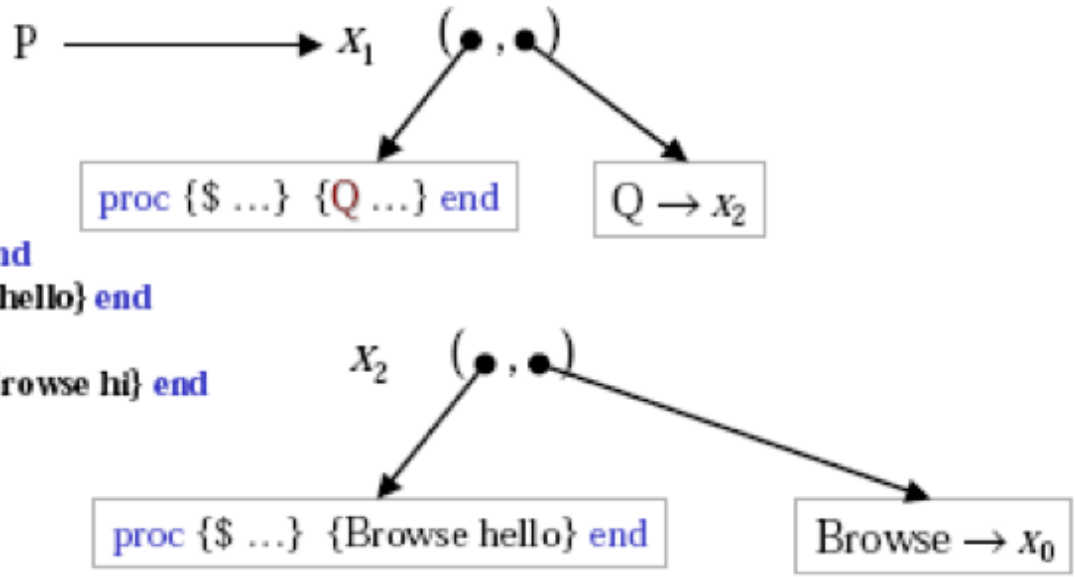
Es una pareja (clausura):

$$(\texttt{proc } \{\ \$\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \}\ \langle d \rangle\ \texttt{end},\ CE\ )$$

CE (ambiente contextual) es $E|_{\{\langle z \rangle_1, \ldots, \langle z \rangle_n\}}$, donde *E* es el ambiente en el momento en que el procedimiento es definido.

Representación interna de los procedimientos (5)

```
local P Q in
    P = proc {$ ...} {Q ...} end
    Q = proc {$ ...} {Browse hello} end
    local Q in
        Q = proc {$ ...} {Browse hi} end
    end
        {P ...}
    end
end
```



Modelos y Paradigmas de Programación

Plan

Declaraciones que se suspenden

$$
\begin{aligned}
\langle d \rangle ::= & \ldots \\
& \mid \quad \texttt{if } \langle x \rangle \texttt{ then } \langle d \rangle_1 \texttt{ else } \langle d \rangle_2 \texttt{ end} \\
& \mid \quad \texttt{case } \langle x \rangle \texttt{ of } \langle \textit{patrón} \rangle \texttt{ then } \langle d \rangle_1 \texttt{ else } \langle d \rangle_2 \texttt{ end} \\
& \mid \quad \texttt{'\{' } \langle x \rangle \ \langle y \rangle_1 \cdots \langle y \rangle_n \texttt{ '\}'}
\end{aligned}
$$

- ¿Qué debería suceder con estas declaraciones si $\langle x \rangle$ es no-ligada?
- Condición de activación: $E(\langle x \rangle)$ debe estar ligada a un número, registro, o procedimiento.
- En el modelo declarativo una vez una declaración se suspende, no continuará nunca.

Declaraciones que se suspenden

$$
\begin{aligned}
\langle d \rangle ::= & \ldots \\
& \mid \quad \texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end} \\
& \mid \quad \texttt{case}\ \langle x \rangle\ \texttt{of}\ \langle \textit{patrón} \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end} \\
& \mid \quad \texttt{'\{'}\ \langle x \rangle\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \texttt{'\}'}
\end{aligned}
$$

- ¿Qué debería suceder con estas declaraciones si $\langle x \rangle$ es no-ligada?
- Condición de activación: $E(\langle x \rangle)$ debe estar ligada a un número, registro, o procedimiento.
- En el modelo declarativo una vez una declaración se suspende, no continuará nunca.

Declaraciones que se suspenden

$$
\begin{aligned}
\langle d \rangle ::= & \ldots \\
& \mid \quad \texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end} \\
& \mid \quad \texttt{case}\ \langle x \rangle\ \texttt{of}\ \langle \textit{patrón} \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end} \\
& \mid \quad \texttt{'\{'}\ \langle x \rangle\ \langle y \rangle_1 \cdots \langle y \rangle_n\ \texttt{'\}'}
\end{aligned}
$$

- ¿Qué debería suceder con estas declaraciones si $\langle x \rangle$ es no-ligada?
- Condición de activación: $E(\langle x \rangle)$ debe estar ligada a un número, registro, o procedimiento.
- En el modelo declarativo una vez una declaración se suspende, no continuará nunca.

Declaraciones que se suspenden

$$
\begin{aligned}
\langle d \rangle ::= & \ldots \\
\mid \quad & \texttt{if } \langle x \rangle \texttt{ then } \langle d \rangle_1 \texttt{ else } \langle d \rangle_2 \texttt{ end} \\
\mid \quad & \texttt{case } \langle x \rangle \texttt{ of } \langle \textit{patrón} \rangle \texttt{ then } \langle d \rangle_1 \texttt{ else } \langle d \rangle_2 \texttt{ end} \\
\mid \quad & \texttt{'\{'} \ \langle x \rangle \ \langle y \rangle_1 \cdots \langle y \rangle_n \ \texttt{'\}'}
\end{aligned}
$$

- ¿Qué debería suceder con estas declaraciones si $\langle \text{x} \rangle$ es no-ligada?
- Condición de activación: $E(\langle \text{x} \rangle)$ debe estar ligada a un número, registro, o procedimiento.
- En el modelo declarativo una vez una declaración se suspende, no continuará nunca.

El Condicional (la declaración `if`)

La declaración semántica es:

$$(\texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle x \rangle)$ está determinada, entonces:
  - Si $E(\langle x \rangle)$ no es un booleano (`true` o `false`) entonces lance una condición de error.
  - Si $E(\langle x \rangle)$ es `true`, entonces coloque $(\langle d \rangle_1, E)$ en la pila.
- Si la condición de activación es falsa, entonces la ejecución se suspende.

El Condicional (la declaración `if`)

La declaración semántica es:

$$(\texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x} \rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x} \rangle)$ no es un booleano (`true` o `false`) entonces lance una condición de error.
  - Si $E(\langle \mathrm{x} \rangle)$ es `true`, entonces coloque $(\langle \mathrm{d} \rangle_1, E)$ en la pila.
  - Si $E(\langle \mathrm{x} \rangle)$ es `false`, entonces coloque $(\langle \mathrm{d} \rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces la ejecución se suspende.

El Condicional (la declaración `if`)

La declaración semántica es:

$$(\texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x} \rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x} \rangle)$ no es un booleano (`true` o `false`) entonces lance una condición de error.
  - Si $E(\langle \mathrm{x} \rangle)$ es `true`, entonces coloque $(\langle \mathrm{d} \rangle_1, E)$ en la pila.
  - Si $E(\langle \mathrm{x} \rangle)$ es `false`, entonces coloque $(\langle \mathrm{d} \rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces la ejecución se suspende.

El Condicional (la declaración `if`)

La declaración semántica es:

$$(\texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x} \rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x} \rangle)$ no es un booleano (`true` o `false`) entonces lance una condición de error.
  - Si $E(\langle \mathrm{x} \rangle)$ es `true`, entonces coloque $(\langle \mathrm{d} \rangle_1, E)$ en la pila.
  - Si $E(\langle \mathrm{x} \rangle)$ es `false`, entonces coloque $(\langle \mathrm{d} \rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces la ejecución se suspende.

El Condicional (la declaración `if`)

La declaración semántica es:

$$(\texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x} \rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x} \rangle)$ no es un booleano (**`true`** o **`false`**) entonces lance una condición de error.
  - Si $E(\langle \mathrm{x} \rangle)$ es **`true`**, entonces coloque $(\langle \mathrm{d} \rangle_1, E)$ en la pila.
  - Si $E(\langle \mathrm{x} \rangle)$ es **`false`**, entonces coloque $(\langle \mathrm{d} \rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces la ejecución se suspende.

El Condicional (la declaración `if`)

La declaración semántica es:

$$(\texttt{if}\ \langle x \rangle\ \texttt{then}\ \langle d \rangle_1\ \texttt{else}\ \langle d \rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \text{x} \rangle)$ está determinada, entonces:
  - Si $E(\langle \text{x} \rangle)$ no es un booleano (`true` o `false`) entonces lance una condición de error.
  - Si $E(\langle \text{x} \rangle)$ es `true`, entonces coloque $(\langle \text{d} \rangle_1, E)$ en la pila.
  - Si $E(\langle \text{x} \rangle)$ es `false`, entonces coloque $(\langle \text{d} \rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces la ejecución se suspende.

La Invocación de procedimiento

La declaración semántica es:

$$\left(\{\langle x\rangle\ \langle y\rangle_1 \cdots \langle y\rangle_n\}, E\right)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle x\rangle)$ está determinada, entonces:
  - Si $E(\langle x\rangle)$ no es un valor de tipo procedimiento o es un procedimiento con un número de argumentos diferente de n, entonces lance una condición de error.
  - Si $E(\langle x\rangle)$ tiene la forma $(\text{proc}\ \{\$\ \langle z\rangle_1 \cdots \langle z\rangle_n\}\ \langle s\rangle\ \text{end}, CE)$ entonces coloque

    $(\langle s\rangle, CE + \{\langle z\rangle_1 \to E(\langle y\rangle_1), \ldots, \langle z\rangle_n \to E(\langle y\rangle_n)\})$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

La Invocación de procedimiento

La declaración semántica es:

$$(\{\langle x\rangle\ \langle y\rangle_1 \cdots \langle y\rangle_n\}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x}\rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x}\rangle)$ no es un valor de tipo procedimiento o es un procedimiento con un número de argumentos diferente de n, entonces lance una condición de error.
  - Si $E(\langle \mathrm{x}\rangle)$ tiene la forma $(\texttt{proc}\ \{\ \$\ \langle \mathrm{z}\rangle_1 \cdots \langle \mathrm{z}\rangle_n\}\ \langle \mathrm{d}\rangle\ \texttt{end}, CE)$ entonces coloque $(\langle \mathrm{d}\rangle, CE + \{\langle \mathrm{z}\rangle_1 \to E(\langle \mathrm{y}\rangle_1), \ldots, \langle \mathrm{z}\rangle_n \to E(\langle \mathrm{y}\rangle_n)\})$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

La Invocación de procedimiento

La declaración semántica es:

$$\left(\{\langle x\rangle\ \langle y\rangle_1 \cdots \langle y\rangle_n\}, E\right)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x}\rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x}\rangle)$ no es un valor de tipo procedimiento o es un procedimiento con un número de argumentos diferente de n, entonces lance una condición de error.
  - Si $E(\langle \mathrm{x}\rangle)$ tiene la forma $(\texttt{proc}\ \{\$\ \langle \mathrm{z}\rangle_1 \cdots \langle \mathrm{z}\rangle_n\}\ \langle \mathrm{d}\rangle\ \texttt{end}, CE)$ entonces coloque $(\langle \mathrm{d}\rangle, CE + \{\langle \mathrm{z}\rangle_1 \to E(\langle \mathrm{y}\rangle_1), \ldots, \langle \mathrm{z}\rangle_n \to E(\langle \mathrm{y}\rangle_n)\})$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

La Invocación de procedimiento

La declaración semántica es:

$$(\{\langle x\rangle\ \langle y\rangle_1 \cdots \langle y\rangle_n\}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x}\rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x}\rangle)$ no es un valor de tipo procedimiento o es un procedimiento con un número de argumentos diferente de n, entonces lance una condición de error.
  - Si $E(\langle \mathrm{x}\rangle)$ tiene la forma $(\texttt{proc}\ \{\ \$\ \langle \mathrm{z}\rangle_1 \cdots \langle \mathrm{z}\rangle_n\}\ \langle \mathrm{d}\rangle\ \texttt{end}, CE)$ entonces coloque $(\langle \mathrm{d}\rangle, CE + \{\langle \mathrm{z}\rangle_1 \to E(\langle \mathrm{y}\rangle_1), \ldots, \langle \mathrm{z}\rangle_n \to E(\langle \mathrm{y}\rangle_n)\})$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

La Invocación de procedimiento

La declaración semántica es:

$$(\{\langle x\rangle\ \langle y\rangle_1 \cdots \langle y\rangle_n\}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x}\rangle)$ está determinada, entonces:
  - Si $E(\langle \mathrm{x}\rangle)$ no es un valor de tipo procedimiento o es un procedimiento con un número de argumentos diferente de n, entonces lance una condición de error.
  - Si $E(\langle \mathrm{x}\rangle)$ tiene la forma $(\texttt{proc}\ \{\ \$\ \langle \mathrm{z}\rangle_1 \cdots \langle \mathrm{z}\rangle_n\}\ \langle \mathrm{d}\rangle\ \texttt{end}, CE)$ entonces coloque $(\langle \mathrm{d}\rangle, CE + \{\langle \mathrm{z}\rangle_1 \to E(\langle \mathrm{y}\rangle_1), \ldots, \langle \mathrm{z}\rangle_n \to E(\langle \mathrm{y}\rangle_n)\})$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

El Reconocimiento de patrones (la declaración `case`)

La declaración semántica es:

$$(\texttt{case}\ \langle x\rangle\ \texttt{of}\ \langle lit\rangle(\langle cmp\rangle_1\texttt{:}\ \langle x\rangle_1 \cdots\ \langle cmp\rangle_n\texttt{:}\ \langle x\rangle_n)\ \texttt{then}\ \langle d\rangle_1\ \texttt{else}\ \langle d\rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle x\rangle)$ está determinada, entonces:
  - Si la etiqueta de $E(\langle x\rangle)$ es $\langle lit\rangle$ y su aridad es $\{\langle cmp\rangle_1, \cdots, \langle cmp\rangle_n\}$, entonces coloque $(\langle d\rangle_1, E + \{\langle x\rangle_1 \mapsto E(\langle x\rangle).\langle cmp\rangle_1, \ldots, \langle x\rangle_n \mapsto E(\langle x\rangle).\langle cmp\rangle_n\})$ en la pila.
  - En cualquier otro caso coloque $(\langle d\rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

El Reconocimiento de patrones (la declaración `case`)

La declaración semántica es:

$$(\texttt{case}\ \langle x\rangle\ \texttt{of}\ \langle lit\rangle(\langle cmp\rangle_1 \langle x\rangle_1 \cdots \langle cmp\rangle_n \langle x\rangle_n)\ \texttt{then}\ \langle d\rangle_1\ \texttt{else}\ \langle d\rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle x\rangle)$ está determinada, entonces:
  - Si la etiqueta de $E(\langle x\rangle)$ es $\langle lit\rangle$ y su aridad es $\{\langle cmp\rangle_1, \cdots, \langle cmp\rangle_n\}$, entonces coloque $(\langle d\rangle_1, E + \{\langle x\rangle_1 \to E(\langle x\rangle).\langle cmp\rangle_1, \ldots, \langle x\rangle_n \to E(\langle x\rangle).\langle cmp\rangle_n\})$ en la pila.
  - En cualquier otro caso coloque $(\langle d\rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

El Reconocimiento de patrones (la declaración `case`)

La declaración semántica es:

$$(\texttt{case}\ \langle x\rangle\ \texttt{of}\ \langle lit\rangle(\langle cmp\rangle_1 \langle x\rangle_1 \cdots \langle cmp\rangle_n \langle x\rangle_n)\ \texttt{then}\ \langle d\rangle_1\ \texttt{else}\ \langle d\rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle x\rangle)$ está determinada, entonces:
  - Si la etiqueta de $E(\langle x\rangle)$ es $\langle lit\rangle$ y su aridad es $\{\langle cmp\rangle_1, \cdots, \langle cmp\rangle_n\}$, entonces coloque $(\langle d\rangle_1, E + \{\langle x\rangle_1 \to E(\langle x\rangle).\langle cmp\rangle_1, \ldots, \langle x\rangle_n \to E(\langle x\rangle).\langle cmp\rangle_n\})$ en la pila.
  - En cualquier otro caso coloque $(\langle d\rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

El Reconocimiento de patrones (la declaración `case`)

La declaración semántica es:

$$(\texttt{case}\ \langle x\rangle\ \texttt{of}\ \langle lit\rangle(\langle cmp\rangle_1 \langle x\rangle_1 \cdots \langle cmp\rangle_n \langle x\rangle_n)\ \texttt{then}\ \langle d\rangle_1\ \texttt{else}\ \langle d\rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x}\rangle)$ está determinada, entonces:
  - Si la etiqueta de $E(\langle \mathrm{x}\rangle)$ es $\langle \mathrm{lit}\rangle$ y su aridad es $\{\langle \mathrm{cmp}\rangle_1, \cdots, \langle \mathrm{cmp}\rangle_n\}$, entonces coloque $(\langle \mathrm{d}\rangle_1, E + \{\langle \mathrm{x}\rangle_1 \to E(\langle \mathrm{x}\rangle).\langle \mathrm{cmp}\rangle_1, \ldots, \langle \mathrm{x}\rangle_n \to E(\langle \mathrm{x}\rangle).\langle \mathrm{cmp}\rangle_n\})$ en la pila.
  - En cualquier otro caso coloque $(\langle \mathrm{d}\rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

El Reconocimiento de patrones (la declaración `case`)

La declaración semántica es:

$$(\texttt{case}\ \langle x\rangle\ \texttt{of}\ \langle lit\rangle(\langle cmp\rangle_1 \langle x\rangle_1 \cdots \langle cmp\rangle_n \langle x\rangle_n)\ \texttt{then}\ \langle d\rangle_1\ \texttt{else}\ \langle d\rangle_2\ \texttt{end}, E)$$

La ejecución consiste de las acciones siguientes:

- Si $E(\langle \mathrm{x}\rangle)$ está determinada, entonces:
  - Si la etiqueta de $E(\langle \mathrm{x}\rangle)$ es $\langle \mathrm{lit}\rangle$ y su aridad es $\{\langle \mathrm{cmp}\rangle_1, \cdots, \langle \mathrm{cmp}\rangle_n\}$, entonces coloque $(\langle \mathrm{d}\rangle_1, E + \{\langle \mathrm{x}\rangle_1 \to E(\langle \mathrm{x}\rangle).\langle \mathrm{cmp}\rangle_1, \ldots, \langle \mathrm{x}\rangle_n \to E(\langle \mathrm{x}\rangle).\langle \mathrm{cmp}\rangle_n\})$ en la pila.
  - En cualquier otro caso coloque $(\langle \mathrm{d}\rangle_2, E)$ en la pila.
- Si la condición de activación es falsa, entonces se suspende la ejecución.

Plan

Identificadores de variables y alcance estático

Recuerde el programa:

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=1} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=2} \\ \quad \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right. \\ \quad \langle d \rangle_2 \equiv \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right.$$

Identificadores de variables y alcance estático

Recuerde el programa:

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=1} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=2} \\ \quad \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right. \\ \quad \langle d \rangle_2 \equiv \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right.$$

Ejecución

- $([(\langle d \rangle, \emptyset)], \emptyset)$
- $([(\langle d \rangle_1 \langle d \rangle_2, \{X \rightarrow x\})], \{y = 1, x = y\} \equiv \{x = 1\})$

Identificadores de variables y alcance estático

Recuerde el programa:

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=1} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=2} \\ \quad \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right. \\ \quad \langle d \rangle_2 \equiv \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right.$$

Ejecución

- $([(\langle d \rangle, \emptyset)], \emptyset)$
- $([(\langle d \rangle_1 \langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{y = 1, x = y\} \equiv \{x = 1\})$

Identificadores de variables y alcance estático

Recuerde el programa:

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=1} \\ \quad \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=2} \\ \quad \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right. \\ \quad \langle d \rangle_1 \equiv \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right. \qquad \langle d \rangle_2 \equiv \texttt{\{Browse X\}}$$

Ejecución

- $([(\langle d \rangle, \emptyset)], \emptyset)$
- $([(\langle d \rangle_1 \langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{y = 1, x = y\} \equiv \{x = 1\})$

Ejecución (cont.)

- $([(\langle d \rangle_1, \{\texttt{X} \rightarrow x\}), (\langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{x = 1\})$
  $\langle d \rangle_1$ y $\langle d \rangle_2$ tienen su ambiente propio.
- Ejecución de $\langle d \rangle_1$. Después de `local`: $([(\texttt{X=2 \{Browse X\}}, \{\texttt{X} \rightarrow x'\}), (\langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{x', x = 1\})$
- Después de `X=2`: $([(\texttt{\{Browse X\}}, \{\texttt{X} \rightarrow x'\}), (\texttt{\{Browse X\}}, \{\texttt{X} \rightarrow x\})], \{x' = 2, x = 1\})$
  Ahora se ejecutan los `Browse`

Identificadores de variables y alcance estático

Recuerde el programa:

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=1} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=2} \\ \quad \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right. \\ \quad \langle d \rangle_2 \equiv \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right.$$

Ejecución

- $([(\langle d \rangle, \emptyset)], \emptyset)$
- $([(\langle d \rangle_1 \langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{y = 1, x = y\} \equiv \{x = 1\})$

Ejecución (cont.)

- $([(\langle d \rangle_1, \{\texttt{X} \rightarrow x\}), (\langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{x = 1\})$
  $\langle d \rangle_1$ y $\langle d \rangle_2$ tienen su ambiente propio.
- Ejecución de $\langle d \rangle_1$. Después de `local`:
  $([(\texttt{X=2 \{Browse X\}}, \{\texttt{X} \rightarrow x'\}), (\langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{x', x = 1\})$
- Después de `X=2`: $([(\texttt{\{Browse X\}}, \{\texttt{X} \rightarrow x'\}), (\texttt{\{Browse X\}}, \{\texttt{X} \rightarrow x\})], \{x' = 2, x = 1\})$
  Ahora se ejecutan los `Browse`

Identificadores de variables y alcance estático

Recuerde el programa:

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=1} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{local X in} \\ \quad \texttt{X=2} \\ \quad \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right. \\ \quad \langle d \rangle_2 \equiv \texttt{\{Browse X\}} \\ \texttt{end} \end{array} \right.$$

Ejecución

- $([(\langle d \rangle, \emptyset)], \emptyset)$
- $([(\langle d \rangle_1 \langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{y = 1, x = y\} \equiv \{x = 1\})$

Ejecución (cont.)

- $([(\langle d \rangle_1, \{\texttt{X} \rightarrow x\}), (\langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{x = 1\})$
  $\langle d \rangle_1$ y $\langle d \rangle_2$ tienen su ambiente propio.
- Ejecución de $\langle d \rangle_1$. Después de `local`:
  $([(\texttt{X=2 \{Browse X\}}, \{\texttt{X} \rightarrow x'\}), (\langle d \rangle_2, \{\texttt{X} \rightarrow x\})], \{x', x = 1\})$
- Después de `X=2`: $([(\texttt{\{Browse X\}}, \{\texttt{X} \rightarrow x'\}), (\texttt{\{Browse X\}}, \{\texttt{X} \rightarrow x\})], \{x' = 2, x = 1\})$
  Ahora se ejecutan los `Browse`

Definición e invocación de procedimientos (1)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

Abreviaciones utilizadas:

- Más de una variable en una declaración **`local`**.
- El uso de valores "en línea" en lugar de variables, e.g., `{P 3}` en lugar de **`local`** `X` **`in`** `X=3 {P X}` **`end`**.
- La utilización de operaciones anidadas, e.g., colocar la operación `X>=Y` en lugar del booleano en la declaración **`if`**.

Definición e invocación de procedimientos (1)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

Abreviaciones utilizadas:

- Más de una variable en una declaración **`local`**.
- El uso de valores “en línea” en lugar de variables, e.g., `{P 3}` en lugar de **`local`** `X` **`in`** `X=3 {P X}` **`end`**.
- La utilización de operaciones anidadas, e.g., colocar la operación `X>=Y` en lugar del booleano en la declaración **`if`**.

Definición e invocación de procedimientos (1)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

Abreviaciones utilizadas:

- Más de una variable en una declaración **`local`**.
- El uso de valores “en línea” en lugar de variables, e.g., `{P 3}` en lugar de **`local`** `X` **`in`** `X=3 {P X}` **`end`**.
- La utilización de operaciones anidadas, e.g., colocar la operación `X>=Y` en lugar del booleano en la declaración **`if`**.

Definición e invocación de procedimientos (2)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

- Estado inicial: $([(\langle d \rangle, \emptyset)], \emptyset)$
- Después de **local**: $([(\langle d \rangle_1, \{\texttt{Max} \rightarrow m, \texttt{C} \rightarrow c\})], \{m, c\})$
- Después de ligar Max:
  $([(\texttt{\{Max 3 5 C\}}, \{\texttt{Max} \rightarrow m, \texttt{C} \rightarrow c\})],$
  $\{m = (\texttt{proc \{\$ X Y Z\}} \langle d \rangle_3 \texttt{ end}, \emptyset), c\})$

Definición e invocación de procedimientos (2)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

- Estado inicial: $([(\langle d \rangle, \emptyset)], \emptyset)$
- Después de **local**:$([(\langle d \rangle_1, \{\texttt{Max} \rightarrow m, \texttt{C} \rightarrow c\})], \{m, c\})$
- Después de ligar Max:
  $([(\texttt{\{Max 3 5 C\}}, \{\texttt{Max} \rightarrow m, \texttt{C} \rightarrow c\})],$
  $\{m = (\texttt{proc \{\$ X Y Z\}} \langle d \rangle_3 \texttt{ end}, \emptyset), c\})$

Definición e invocación de procedimientos (2)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

- Estado inicial: $([(\langle d \rangle, \emptyset)], \emptyset)$
- Después de **local**:$([(\langle d \rangle_1, \{\texttt{Max} \rightarrow m, \texttt{C} \rightarrow c\})], \{m, c\})$
- Después de ligar Max:
  $([(\texttt{\{Max 3 5 C\}}, \{\texttt{Max} \rightarrow m, \texttt{C} \rightarrow c\})],$
  $\{m = (\textbf{proc } \texttt{\{\$ X Y Z\}} \langle d \rangle_3 \textbf{ end}, \emptyset), c\})$

Definición e invocación de procedimientos (3)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

- Después de la invocación del procedimiento:
  $([(\langle d \rangle_3, \{\texttt{X} \rightarrow a, \texttt{Y} \rightarrow b, \texttt{Z} \rightarrow c\})],$
  $\{m = (\textbf{proc } \texttt{\{\$ X Y Z\}} \langle d \rangle_3 \textbf{ end}, \emptyset), a = 3, b = 5, c\})$
- Después de la comparación X>=Y:
  $([(\langle d \rangle_3, \{\texttt{X} \rightarrow a, \texttt{Y} \rightarrow b, \texttt{Z} \rightarrow c, \texttt{T} \rightarrow t\})],$
  $\{m = (\textbf{proc } \texttt{\{\$ X Y Z\}} \langle d \rangle_3 \textbf{ end}, \emptyset), a = 3, b = 5, c, t = \textbf{false}\})$
- Termina después de $\langle d \rangle_3$:
  $([], \{m = (\textbf{proc } \texttt{\{\$ X Y Z\}} \langle d \rangle_3 \textbf{ end}, \emptyset),$
  $a = 3, b = 5, c = 5, t = \textbf{false}\})$

Definición e invocación de procedimientos (3)

$$\langle d \rangle \equiv \left\{ \begin{array}{l} \texttt{local Max C in} \\ \quad \langle d \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=proc \{\$ X Y Z\}} \\ \quad \langle d \rangle_3 \equiv \{ \texttt{if (X>=Y) then Z=X else Z=Y end} \\ \quad \texttt{end} \\ \quad \langle d \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \texttt{end} \end{array} \right.$$

- Después de la invocación del procedimiento:
  $([(\langle d \rangle_3, \{X \rightarrow a, Y \rightarrow b, Z \rightarrow c\})],$
  $\{m = (\textbf{proc } \{\$\ X\ Y\ Z\}\ \langle d \rangle_3\ \textbf{end}, \emptyset), a = 3, b = 5, c\})$
- Después de la comparación X>=Y:
  $([(\langle d \rangle_3, \{X \rightarrow a, Y \rightarrow b, Z \rightarrow c, T \rightarrow t\})],$
  $\{m = (\textbf{proc } \{\$\ X\ Y\ Z\}\ \langle d \rangle_3\ \textbf{end}, \emptyset), a = 3, b = 5, c, t = \textbf{false}\})$
- Termina después de $\langle d \rangle_3$:
  $([], \{m = (\textbf{proc } \{\$\ X\ Y\ Z\}\ \langle d \rangle_3\ \textbf{end}, \emptyset),$
  $a = 3, b = 5, c = 5, t = \textbf{false}\})$

Definición e invocación de procedimientos (3)

$$\langle \mathrm{d} \rangle \equiv \left\{ \begin{array}{l} \textbf{local } \texttt{Max C } \textbf{in} \\ \quad \langle \mathrm{d} \rangle_1 \equiv \left\{ \begin{array}{l} \texttt{Max=}\textbf{proc } \texttt{\{\$ X Y Z\}} \\ \quad \langle \mathrm{d} \rangle_3 \equiv \{ \textbf{ if } \texttt{(X>=Y)} \textbf{ then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end} \\ \qquad\qquad \textbf{end} \\ \quad \langle \mathrm{d} \rangle_2 \equiv \texttt{\{Max 3 5 C\}} \end{array} \right. \\ \textbf{end} \end{array} \right.$$

- Después de la invocación del procedimiento:
  $([(\langle \mathrm{d} \rangle_3, \{\mathtt{X} \to a, \mathtt{Y} \to b, \mathtt{Z} \to c\})],$
  $\{m = (\textbf{proc } \texttt{\{\$ X Y Z\}} \langle \mathrm{d} \rangle_3 \textbf{ end}, \emptyset), a = 3, b = 5, c\})$
- Después de la comparación `X>=Y`:
  $([(\langle \mathrm{d} \rangle_3, \{\mathtt{X} \to a, \mathtt{Y} \to b, \mathtt{Z} \to c, \mathtt{T} \to t\})],$
  $\{m = (\textbf{proc } \texttt{\{\$ X Y Z\}} \langle \mathrm{d} \rangle_3 \textbf{ end}, \emptyset), a = 3, b = 5, c, t = \textbf{false}\})$
- Termina después de $\langle \mathrm{d} \rangle_3$:
  $([], \{m = (\textbf{proc } \texttt{\{\$ X Y Z\}} \langle \mathrm{d} \rangle_3 \textbf{ end}, \emptyset),$
  $a = 3, b = 5, c = 5, t = \textbf{false}\})$

Invocación de procedimiento con referencias externas

```
local LowerBound Y C in
   Y=5
   proc {LowerBound X ?Z}
      if X>=Y then Z=X else Z=Y end
   end
   {LowerBound 3 C}
end
```

- Antes de la invocación:
  $((\{\texttt{LowerBound A C}\}, \{\texttt{Y} \rightarrow y, \texttt{LowerBound} \rightarrow lb, \texttt{A} \rightarrow a, \texttt{C} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y = 5, a = 3, c\})$
- Después de la invocación:
  $([(\textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end}, \{\texttt{Y} \rightarrow y, \texttt{X} \rightarrow a, \texttt{Z} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y = 5, a = 3, c\})$

Invocación de procedimiento con referencias externas

```
local LowerBound Y C in
   Y=5
   proc {LowerBound X ?Z}
      if X>=Y then Z=X else Z=Y end
   end
   {LowerBound 3 C}
end
```

- Antes de la invocación:
  $((\{\texttt{LowerBound A C}\}, \{\texttt{Y} \rightarrow y, \texttt{LowerBound} \rightarrow lb, \texttt{A} \rightarrow a, \texttt{C} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y = 5, a = 3, c\})$
- Después de la invocación:
  $([(\textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end}, \{\texttt{Y} \rightarrow y, \texttt{X} \rightarrow a, \texttt{Z} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y = 5, a = 3, c\})$

Invocación de procedimiento con referencias externas (2)

```
local LowerBound Y C in
   Y=5
   proc {LowerBound X ?Z}
      if X>=Y then Z=X else Z=Y end
   end
   local Y in  Y=10   {LowerBound 3 C}   end
end
```

- Antes de la invocación:
  $([(\{\texttt{LowerBound A C}\}, \{\texttt{Y} \rightarrow y', \texttt{LowerBound} \rightarrow lb, \texttt{A} \rightarrow a, \texttt{C} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y' = 10, y = 5, a = 3, c\})$
- Después de la invocación:
  $([(\textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end}, \{\texttt{Y} \rightarrow y, \texttt{X} \rightarrow a, \texttt{Z} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y' = 10, y = 5, a = 3, c\})$

Invocación de procedimiento con referencias externas (2)

```
local LowerBound Y C in
   Y=5
   proc {LowerBound X ?Z}
      if X>=Y then Z=X else Z=Y end
   end
   local Y in  Y=10   {LowerBound 3 C}   end
end
```

- Antes de la invocación:
  $([(\{\texttt{LowerBound A C}\}, \{\texttt{Y} \rightarrow y', \texttt{LowerBound} \rightarrow lb, \texttt{A} \rightarrow a, \texttt{C} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y' = 10, y = 5, a = 3, c\}\ )$
- Después de la invocación:
  $([(\textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end}, \{\texttt{Y} \rightarrow y, \texttt{X} \rightarrow a, \texttt{Z} \rightarrow c\})],$
  $\{lb = (\textbf{proc } \{\$\ \texttt{X Z}\}\ \textbf{if } \texttt{X>=Y } \textbf{then } \texttt{Z=X } \textbf{else } \texttt{Z=Y } \textbf{end end}, \{\texttt{Y} \rightarrow y\}), y' = 10, y = 5, a = 3, c\}\ )$