Universidad Nacional Autónoma de México
Facultad de Ciencias
Lenguajes de Programación

•••

Karla Ramírez Pulido

Juicios de Tipo

Juicios de tipo (notación ":" es de tipo)

n es de tipo **number**

$$n : \textbf{number}$$

una función es de tipo **function**

$$\{\, fun\ \{i\}\ \ b\,\} : \textbf{function}$$

Ahora agregamos un contexto: Γ (Gamma)

En el contexto Γ se prueba que "**e**" tiene tipo "**t**"

$$\Gamma \vdash e : T$$

Aplicando esta notación a una n de tipo number tenemos:

$$\Gamma \vdash n : \text{number}$$

Ahora agregamos un contexto: Γ (Gamma)

$$\Gamma \vdash i : \Gamma(i)$$

El tipo del identificador “i” es cualquier tipo que esté ligado en su contexto o ambiente, es decir, el contexto Γ

$$\Gamma \vdash i : \Gamma(i)$$

Ahora agregamos un contexto: Γ (Gamma)

Para la expresión suma:

$$\frac{\Gamma \vdash l : \text{number} \qquad \Gamma \vdash r : \text{number}}{\Gamma \vdash \{ + \ l \ r \} : \text{number}}$$

Vamos a extender la gramática:

Tenemos 2 tipos:

- number
- funciones

```
<type> ::= number
         | (<type> -> <type>)
```

Las funciones se conforman por un tipo que recibe y otro que regresa

(**<type-recibe>** ----> <type-regresa>)

Ejemplo de código con tipos explícitos:

```
{fun {x : number} : number
    {+ x x} }
```

Lo que recibe: number

Lo que regresa: number

Ejemplo de código con tipos explícitos:

```
{fun {x : number} : (number -> number)
  {fun {y : number} : number
    {+ x y}}}
```

Primer renglón del código:

Lo que recibe: number

Lo que regresa: (number -> number) esto es una función.

Ahora agregamos un contexto: Γ (Gamma)

Para la expresión función:

$$\{\, \textbf{fun}\ \{\, i \,\}\quad b \,\} : \textbf{function}$$

¿Cuántas sub-expresiones pueden tener un tipo en la función?

- Parámetro: i
- Cuerpo de la función: b

Ahora agregamos un contexto: Γ (Gamma)

Juicio de tipo para funciones:

$$\frac{\Gamma\,[\,i \leftarrow T_1\,] \vdash b : T_2}{\Gamma \vdash \{\, fun\ \{\ i : T_1\ \}\ : T_2 \quad b\,\} : (T_1 \rightarrow T_2)}$$

Ahora agregamos un contexto: Γ (Gamma)

Juicio de tipo para la expresión condicional IF:

$$\frac{\Gamma \vdash c : \texttt{boolean} \qquad \Gamma \vdash t : \tau \qquad \Gamma \vdash e : \tau}{\Gamma \vdash \{\texttt{if}\ c\ t\ e\} : \tau}$$

NOTA: Es importante distinguir que este juicio de tipo tanto en la rama del then-expr como el else-expr regresan el mismo tipo T.

Vamos a extender de nuevo los tipos de la gramática

Ahora tendremos booleanos:

```
<type> ::= number
         | boolean
         | (<type> -> <type>)
```

Juicio de tipo para el if donde t y e no necesariamente regresan algo del mismo tipo

$$\Gamma \vdash c : \text{boolean} \qquad \Gamma \vdash t : T1 \qquad \Gamma \vdash e : T2$$

$$\Gamma \vdash \{\text{if } c \ t \ e\} : T1 \mid T2$$

Ejemplo 1:

Expresión es suma

{+ 2

{+ 5 7} }

Los argumentos de la función son: 2 y {+ 5 7}

La expresión del lado derecho es otra expresión suma: {+ 5 7} entonces ahí tenemos que aplicar la regla de nuevo sobre la nueva expresión.

Ejemplo 1:

```
{+ 2
   {+ 5 7}}
```

Antecedente:

Consecuente: $\Gamma \vdash \{+\ 2\ \ \{+\ 5\ \ 7\}\} : \text{number}$

Ejemplo 1:

```
{+ 2
   {+ 5 7}}
```

$\Gamma \vdash$ 2 : number $\qquad$ $\Gamma \vdash$ {+ 5 7} : number

---

$\Gamma \vdash$ {+ 2 {+ 5 7}} : number

Ejemplo 1:

```
{+ 2
   {+ 5 7}}
```

$\Gamma \vdash 5$: number $\Gamma \vdash 7$:number

$\Gamma \vdash 2$: number $\Gamma \vdash \{+\ 5\ 7\}$: number

---

$\Gamma \vdash \{+\ 2\ \ \{+\ \ 5\ \ 7\}\}$: number

Ejemplo 2:

Haz el juicio de tipo para la siguiente expresión

```
{{fun {x : number} : number
    {+ x 3}}
 5}
```

Ejemplo 2:

```
{{fun {x : number} : number
    {+ x 3}}
 5}
```

$\Gamma \vdash$ { {fun {x : number} : number {+ x 3}} 5 } : ???

Ejemplo 2:

```
{{fun {x : number} : number
    {+ x 3}}
 5}
```

$\Gamma \vdash$ { {fun {x : number} : number {+ x 3}} 5 } : number

Ejemplo 2:

$\Gamma \vdash$ {fun {x : number} : number {+ x 3}} : (number → number) $\Gamma \vdash$ 5 :number

---

$\Gamma \vdash$ { {fun {x : number} : number {+ x 3}} 5 } : number

Ejemplo 2:

$$\frac{\Gamma[x \leftarrow number] \vdash \{+\ x\ 3\} : number}{\Gamma \vdash \{fun\ \{x : number\} : number\ \{+\ x\ 3\}\} : (number \rightarrow number) \qquad \Gamma \vdash 5 : number}$$

$$\Gamma \vdash \{\ \{fun\ \{x : number\} : number\ \{+\ x\ 3\}\}\ 5\ \} : number$$

Ejemplo 2:

$$\frac{\Gamma[x \leftarrow number] \vdash x:number \qquad \Gamma[x \leftarrow number] \vdash 3:number}{\Gamma[x \leftarrow number] \vdash \{+\ x\ 3\} : number}$$

$$\frac{\Gamma \vdash \{fun\ \{x : number\} : number\ \ \{+\ x\ 3\}\}: (number \rightarrow number) \qquad \Gamma \vdash 5:number}{\Gamma \vdash \{\ \{fun\ \{x : number\} : number\ \ \{+\ x\ 3\}\}\ \ 5\ \} :\ number}$$

Ejercicio 3:

Haz el juicio de tipo para la siguiente expresión:

```
{ +  3

    {fun {x : number} : number  x} }
```

Ejercicio 3:

$\Gamma \vdash$ {+ 3 {fun {x : number} : number x}} : number

Ejercicio 3:

$$\Gamma \vdash 3 : \text{number} \qquad \Gamma \vdash \{\text{fun } \{x : \text{number}\} : \text{number } x\} : (\text{number} \to \text{number})$$

$$\Gamma \vdash \{+ \ 3 \ \{\text{fun } \{x : \text{number}\} : \text{number } x\}\} : \text{number}$$

Ejercicio 3:

$$\Gamma[x \leftarrow number] \vdash x : number$$

$$\frac{\Gamma \vdash 3 : number \qquad \Gamma \vdash \{fun\ \{x : number\} : number\ \ x\} : (number \rightarrow number)}{\Gamma \vdash \{+\ \ 3\ \ \{fun\ \{x : number\} : number\ \ x\}\} : number}$$

```
<TRCFAE> ::= ...
           | {rec {<id>  : <type> <TRCFAE>} <TRCFAE>}
```

```
{rec {fac {fun {n}
           {if (= n 0)
               1
               {* n {fac {- n 1}}}}}
 {fac 2} }
```

```
<TRCFAE> ::= ...
           | {rec {<id>  : <type> <TRCFAE>} <TRCFAE>}
```

```
{rec {fac : (number → number)
        {fun { n }
          {if (= n 0)
              1
              {* n {fac {- n 1}}}}}
  {fac 2} }
```

```
<TRCFAE> ::= ...
           | {rec {<id>   : <type> <TRCFAE>} <TRCFAE>}
```

```
{rec {fac : (number → number)
        {fun {n : number}  : number
          {if (= n 0)
              1
              {* n {fac {- n 1}}}}}
  {fac 2} }
```

```
<TRCFAE> ::= ...
           | {rec {<id>  : <type> <TRCFAE>} <TRCFAE>}
```

```
{rec {fac : (number → number)
        {fun {n : number} : number
          {if (= n 0)
              1
              {* n {fac {- n 1}}}}}
  {fac 2} }
```

$$\frac{???}{\Gamma \vdash \{\texttt{rec}\ \{i\ :\ \tau_i\ v\}\ b\} : \tau}$$

Juicio de tipo para el constructor rec

$$???$$

$$\frac{}{\Gamma \vdash \{\, rec \;\; \{\, i \;:\; \mathrm{T}_i \;\; v \,\} \;\; b \,\} \;:\; \mathsf{T}}$$

donde ***i*** (id) representa el nombre de la función, ***v*** (value) representa la definición de la función y ***b*** representa al cuerpo del rec (body-rec)

Juicio de tipo para el constructor rec

Aquí se muestra el cuerpo del rec, i.e. “b”

$$\frac{\Gamma[\, i \leftarrow T_i \,] \vdash b : T \qquad ???}{\Gamma \vdash \{\, rec \;\; \{\, i \;:\; T_i \quad v \,\} \;\; b \} \;:\; T}$$

Juicio de tipo para el constructor rec

Aquí se define el tipo de “v” es decir la definición de la función.

$$\frac{\Gamma[\, i \leftarrow T_i \,] \vdash b : T \qquad \Gamma[\, i \leftarrow T_i \,] \vdash v : T_i}{\Gamma \vdash \{\, rec \;\; \{\, i \;:\; T_i \quad v \,\} \quad b \}\; :\; T}$$

¿Dudas?

Gracias