Modelos de Computación y Complejidad

Grado en Ingeniería Informática. Tecnologías Informáticas

Tema 1: Modelos de Computación

David Orellana Martín
Mario de J. Pérez Jiménez

Dpto. Ciencias de la Computación e Inteligencia Artificial
E.T.S. Ingeniería Informática
Universidad de Sevilla

dorellana@us.es (http://www.cs.us.es/∼dorellana/)
marper@us.es (http://www.cs.us.es/∼marper/)

Curso 2021-2022

Objetivos

- ⋆ Presentar un **modelo de computación** orientado a programas.
  - El modelo de computación GOTO.
- ⋆ Definir el concepto de **computabilidad de funciones** en el modelo GOTO.
  - Ejemplos de funciones GOTO-computables.
- ⋆ Introducir **macros** en el modelo GOTO.
  - Salto incondicional.
  - Poner a cero una variable.
  - Asignación.
- ⋆ Ejemplos del uso de macros en el modelo GOTO.

Modelos de Computación

Definición matemática que captura la idea informal de **procedimiento mecánico**:

- ⋆ **Sintaxis**.
- ⋆ **Semántica**.

Modelos de computación orientado a:

- ⋆ **Programas** (Ej: modelo `GOTO`).
- ⋆ **Funciones** (Ej: funciones recursivas).
- ⋆ **Máquinas** (Ej: máquinas de Turing).

Modos de computación:

- ⋆ **Determinista**.
- ⋆ **No-determinista**.
- ⋆ **Secuencial**.
- ⋆ **Paralelo**.

El modelo de computación GOTO: Sintaxis

Modelo secuencial y determinista (conjunto de datos: $\mathbb{N}$).

El lenguaje GOTO

1.- Variables: $\begin{cases} \text{De entrada:} & X_1(=X), X_2, X_3, \ldots \\ \text{De salida:} & Y \\ \text{Auxiliares:} & Z_1(=Z), Z_2, Z_3, \ldots \end{cases}$

VAR: conjunto de todas las variables del lenguaje GOTO.

2.- Etiquetas: $A_1, B_1, C_1, D_1, E_1, A_2, B_2, C_2, D_2, E_2, \ldots$

- Notación: $A = A_1$, $B = B_1$, $C = C_1$, $D = D_1$, $E = E_1$.
- Etiqueta distinguida: E (**etiqueta de salida**).

3.- Instrucciones básicas: (Para cada variable V y etiquetas K, L)

- **Incremento**: $V \longleftarrow V + 1$
- **Decremento**: $V \longleftarrow V - 1$
- **Condicional**: $IF\ V \neq 0\ GOTO\ L$
- **Skip**: $V \longleftarrow V$
- **Instrucciones etiquetadas**: $[K]\ V \longleftarrow V + 1$; $[K]\ V \longleftarrow V - 1$; $[K]\ IF\ V \neq 0\ GOTO\ L$; $[K]\ V \longleftarrow V$, siendo K una **etiqueta distinta de E**.

Programas GOTO

Programa GOTO:

> **Sucesión finita de instrucciones básicas** (**etiquetadas o no**)
> **cuya última instrucción NO es** $Y \longleftarrow Y$.

Longitud de un programa $P = (I_1, \ldots, I_n)$: número n de instrucciones ($|P|$).

El **programa vacío**: consta de 0 instrucciones (se notará $P_\emptyset$).

Ejemplos:

$$P \equiv \boxed{\begin{array}{ll} [A] & X \longleftarrow X - 1 \\ & Y \longleftarrow Y + 1 \\ & IF\ X \neq 0\ GOTO\ A \end{array}} \qquad Q \equiv \boxed{}$$

El programa P tiene longitud 3 y el programa Q tiene longitud 0.

El modelo de computación GOTO: Semántica

Sea P un programa GOTO.

* $\mathtt{VAR}(P)$: conjunto de todas las variables que aparecen en P.
* Un **estado** de P es una aplicación s de VAR en $\mathbb{N}$:
  - Para cada variable V, $s(V)$ indica el valor de V en el estado s.
  - Al describir un estado de P, sólo notaremos los valores de las variables de P y el valor de la variable de salida Y (que pudiera no aparecer en P).
* Una **configuración** o *descripción instantánea* (d.i) de P, es un par ordenado $C = (j, s)$, en donde $1 \leq j \leq 1 + |P|$ y s es un estado de P.
  - $C = (j, s)$ es una **configuración inicial** de P si:
    - $j = 1$.
    - $s(V) \neq 0$ sólo para un número finito de variables de entrada.
    - $s(V) = 0$ para cada variable que no sea de entrada.
  - $C = (j, s)$ es una **configuración de parada** de P si $j = 1 + |P|$.

El modelo de computación GOTO: Semántica

La semántica de un modelo de computación define (formalmente) cómo se ejecutan los procedimientos mecánicos del mismo.

Si P es un programa GOTO entonces toda configuración de P que no sea de parada va a tener una única configuración siguiente.

- Para ello, se ha de definir cómo se ejecutan las instrucciones básicas a un estado y qué efecto produce.

Sea $P = (I_1, \ldots, I_n)$ un programa GOTO y $C = (j, s)$ una configuración de P que **no** es de parada (es decir: $1 \leq j \leq |P|$).

La configuración siguiente de $C = (j, s)$, que notaremos $C' = (j', s')$, se obtiene de C ejecutando la instrucción I_j al estado s, y se define como sigue:

El modelo de computación GOTO: Semántica

Si $I_j \equiv \boxed{V \longleftarrow V + 1}$ entonces $j' = j + 1$ y, además,

- $s'(V) = s(V) + 1$ y $s'(V') = s(V')$ para cada $V' \neq V$.

Si $I_j \equiv \boxed{V \longleftarrow V - 1}$ entonces $j' = j + 1$ y, además,

- Si $s(V) > 0$ entonces $s'(V) = s(V) - 1$ y $s'(V') = s(V')$ para cada $V' \neq V$.
- Si $s(V) = 0$ entonces $s' = s$.

Si I_j es una instrucción de tipo **SKIP** entonces $j' = j + 1$ y $s' = s$.

Si $I_j \equiv \boxed{IF\ V \neq 0\ GOTO\ L}$ entonces $s' = s$ y, además,

- Si $s(V) = 0$ entonces $j' = j + 1$ (si $j < n$, ir a la siguiente instrucción; si $j = n$, terminar).
- Si $s(V) \neq 0$ y existe k tal que I_k es la primera instrucción de P etiquetada con L, entonces $j' = k$ (ir a la primera instrucción etiquetada con L).
- Si $s(V) \neq 0$ y no existe tal k entonces $j' = 1 + |P|$ (finalizar).

Notaremos $C \vdash_P C'$, o bien $C \Rightarrow_P C'$.

El modelo de computación GOTO: Semántica

Configuraciones iniciales de un programa GOTO P:

- Para cada tupla de números naturales $(a_1, \ldots, a_k) \in \mathbb{N}^k$, $k \geq 1$, la **configuración inicial de P con dato de entrada** $(a_1, \ldots, a_k)$ es $C_0 = (1, s_0)$, siendo $s_0(X_1) = a_1, \ldots, s_0(X_k) = a_k$ y $s_0(V) = 0$, para las restantes variables V.

Ejecución de un programa GOTO P:

- Una **computación** $\mathcal{C}$ de P es una sucesión (finita o infinita) de configuraciones $C_0, C_1, \ldots, C_i, \ldots$ tal que:
  - C_0 es una configuración inicial.
  - Para cada $i \in \mathbb{N}$, C_{i+1} es **la** configuración siguiente de C_i .
  - Si la sucesión es finita entonces la última configuración ha de ser de parada.

Una computación $\mathcal{C}$ de P queda unívocamente determinada por una configuración inicial.

El modelo de computación GOTO: Semántica

La computación de P con entrada la tupla de números naturales $(a_1, \ldots, a_k) \in \mathbb{N}^k$, $k \geq 1$, se notará $P(a_1, \ldots, a_k)$.

* Si la computación $P(a_1, \ldots, a_k)$ es la sucesión finita $(C_0, \ldots, C_n)$ entonces:
  - C_0 es la configuración inicial de P con dato de entrada $(a_1, \ldots, a_k)$.
  - La configuración C_n es de parada.
  - Diremos que $P(a_1, \ldots, a_k)$ es de parada y notaremos $P(a_1, \ldots, a_k)\downarrow$.
  - El resultado de la computación $P(a_1, \ldots, a_k)$ es el valor $b \in \mathbb{N}$ de la variable de salida Y en la configuración de parada C_n. Notaremos $P(a_1, \ldots, a_k) = b$.

* Si $P(a_1, \ldots, a_k)$ es una sucesión infinita entonces diremos que dicha computación no es de parada y notaremos $P(a_1, \ldots, a_k)\uparrow$.

El programa vacío

¿Quién es el programa vacío $P_\emptyset$?

- El programa vacío, $P_\emptyset$, carece de instrucciones y, por tanto, su longitud es 0.
- $P_\emptyset$ es una aplicación de $n = 0$ en el conjunto de las instrucciones básicas `GOTO`.
- Esa aplicación tendrá como dominio el conjunto vacío y, por tanto, $P_\emptyset = \emptyset$.
- **El programa vacío $P_\emptyset$ es el conjunto vacio**.

Configuraciones del programa vacío $P_\emptyset$:

- Si $C = (j, s)$ es una configuración del programa vacío entonces se ha de verificar que $1 \leq j \leq 1 + |P_\emptyset|$. Luego, $j = 1$.
- Toda configuración del programa vacío será de parada, ya que $1 = 1 + |P_\emptyset|$.
- Para cada tupla de números naturales $(a_1, \ldots, a_k) \in \mathbb{N}^k$, $k \geq 1$, se tiene que la computación $P_\emptyset(a_1, \ldots, a_k)$ es de parada y, además, $P_\emptyset(a_1, \ldots, a_k) = 0$.

Un ejemplo

Hallar la computación $P(3)$, siendo P el programa `GOTO` siguiente:

$$P \equiv \boxed{\begin{array}{ll} & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \\ & Z \longleftarrow Z + 1 \\ & \mathit{IF}\ Z \neq 0\ \mathit{GOTO}\ E \\ [A] & X \longleftarrow X - 1 \\ & Y \longleftarrow Y + 1 \\ & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \end{array}}$$

Una traza de la computación $P(3)$ será la siguiente:

$$\boxed{\begin{array}{l} C_0 = (1, \{X = 3,\ Y = 0,\ Z = 0\}) \\ C_1 = (4, \{X = 3,\ Y = 0,\ Z = 0\}) \\ C_2 = (5, \{X = 2,\ Y = 0,\ Z = 0\}) \\ C_3 = (6, \{X = 2,\ Y = 1,\ Z = 0\}) \\ C_4 = (4, \{X = 2,\ Y = 1,\ Z = 0\}) \\ C_5 = (5, \{X = 1,\ Y = 1,\ Z = 0\}) \\ C_6 = (6, \{X = 1,\ Y = 2,\ Z = 0\}) \\ C_7 = (4, \{X = 1,\ Y = 2,\ Z = 0\}) \\ C_8 = (5, \{X = 0,\ Y = 2,\ Z = 0\}) \\ C_9 = (6, \{X = 0,\ Y = 3,\ Z = 0\}) \\ C_{10} = (7, \{X = 0,\ Y = 3,\ Z = 0\}) \end{array}}$$

Por tanto, la computación $P(3)$ es de parada y, además, $P(3) = 3$.

Otro ejemplo

Hallar la computación $Q(3)$, siendo Q el programa `GOTO` siguiente:

$$Q \equiv \boxed{\begin{array}{ll} [A] & X \longleftarrow X + 1 \\ & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \end{array}}$$

Una traza de la computación $Q(3)$ será la siguiente:

$$\begin{array}{l} C_0 = (1, \{X = 3,\ Y = 0\}) \\ C_1 = (2, \{X = 4,\ Y = 0\}) \\ C_2 = (1, \{X = 4,\ Y = 0\}) \\ C_3 = (2, \{X = 5,\ Y = 0\}) \\ C_4 = (1, \{X = 5,\ Y = 0\}) \\ C_5 = (2, \{X = 6,\ Y = 0\}) \\ C_6 = (1, \{X = 6,\ Y = 0\}) \\ C_7 = (2, \{X = 7,\ Y = 0\}) \\ C_8 = (1, \{X = 7,\ Y = 0\}) \\ \vdots \quad \vdots \quad \vdots \quad \vdots \end{array}$$

Por tanto, la computación $Q(3)$ **no es de parada**.

Funciones GOTO–computables

En el modelo GOTO se trabaja con funciones parciales de $\mathbb{N}^k$ en $\mathbb{N}$.

Sean P un programa GOTO y $k \geq 1$.

* La **función de aridad k calculada por P** es la función parcial $[\![P]\!]^{(k)}$ de $\mathbb{N}^k$ en $\mathbb{N}$, definida así: $[\![P]\!]^{(k)}(a_1, \ldots, a_k) = P(a_1, \ldots, a_k)$.

Nota: Un programa GOTO calcula **infinitas funciones parciales** entre números naturales (una para cada aridad $k \geq 1$).

Diremos que una **función** f de aridad $k \geq 1$ es **GOTO–computable sii** existe un programa GOTO, P, tal que $f = [\![P]\!]^{(k)}$.

Ejemplos de funciones GOTO–computables

* La **función idénticamente nula** $\mathcal{O}^{(k)}$ de aridad $k \geq 1$, se define como sigue: $\mathcal{O}^{(k)}(x_1, \ldots, x_k) = 0$, para cada tupla $(x_1, \ldots, x_k) \in \mathbb{N}^k$.
  - El programa vacío, $P_\emptyset$, calcula la función idénticamente nula $\mathcal{O}^{(k)}$, para cada $k \geq 1$, ya que para cada tupla $(x_1, \ldots, x_k) \in \mathbb{N}^k$ se verifica que $P_\emptyset(x_1, \ldots, x_k) = 0$.
  - Cualquier programa GOTO que **pare siempre** y en el que **no** aparezca la variable de salida Y, calcula la función $\mathcal{O}^{(k)}$, para cada $k \geq 1$.

* La **función vacía** $f_\emptyset^{(k)}$ de aridad $k \geq 1$, se define como sigue $\mathrm{dom} f_\emptyset^{(k)} = \emptyset$; es decir, $f_\emptyset^{(k)}(x_1, \ldots, x_k) \uparrow$, $\forall (x_1, \ldots, x_k) \in \mathbb{N}^k$.

  Obsérvese que **la función vacía** $f_\emptyset^{(k)}$ **es el conjunto vacío**.
  - Consideremos el programa Q siguiente:

$$Q \equiv \boxed{\begin{array}{ll} [A] & X \longleftarrow X + 1 \\ & IF\ X \neq 0\ GOTO\ A \end{array}}$$

    El programa Q, calcula la función vacía $f_\emptyset^{(k)}$ de aridad $k \geq 1$, ya que para cada tupla $(x_1, \ldots, x_k) \in \mathbb{N}^k$ se verifica que $Q(x_1, \ldots, x_k) \uparrow$.

Ejemplos de funciones GOTO–computables

* La **función identidad** en $\mathbb{N}$, se define como sigue: $Id_{\mathbb{N}}(x) = x$, $\forall x \in \mathbb{N}$.

$$P \equiv \boxed{\begin{array}{ll} & IF\ X \neq 0\ GOTO\ A \\ & Z \longleftarrow Z + 1 \\ & IF\ Z \neq 0\ GOTO\ E \\ [A] & X \longleftarrow X - 1 \\ & Y \longleftarrow Y + 1 \\ & IF\ X \neq 0\ GOTO\ A \end{array}}$$

Vamos a demostrar que la función de aridad 1 que calcula el programa P es, precisamente, la función identidad en $\mathbb{N}$; es decir, $[\![P]\!]^{(1)} = Id_{\mathbb{N}}$.

Concretamente, vamos a probar que para cada $n \in \mathbb{N}$ se verifica $P(n) = n$.

Lo haremos por inducción débil sobre n.

Verificación formal de un programa GOTO

Proposición: Para cada número natural $n \in \mathbb{N}$ se verifica $(P(n) = n)$.

Demostración: Por **inducción débil** sobre n

1. Caso base: $n = 0$.

Hemos de probar que $P(0) = 0$. Para ello, basta describir una traza de la computación $P(0)$:

$$P \equiv \boxed{\begin{array}{ll} & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \\ & Z \longleftarrow Z + 1 \\ & \mathit{IF}\ Z \neq 0\ \mathit{GOTO}\ E \\ [A] & X \longleftarrow X - 1 \\ & Y \longleftarrow Y + 1 \\ & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \end{array}}$$

$$\boxed{\begin{array}{l} C_0 = (1, \{X = 0,\ Y = 0,\ Z = 0\}) \\ C_1 = (2, \{X = 0,\ Y = 0,\ Z = 0\}) \\ C_2 = (3, \{X = 0,\ Y = 0,\ Z = 1\}) \\ C_3 = (7, \{X = 0,\ Y = 0,\ Z = 1\}) \end{array}}$$

Por tanto, se tiene que $P(0) = 0$.

Verificación formal de un programa GOTO

2. Paso inductivo: Sea n y supongamos que $P(n) = n$. Veamos que $P(n+1) = n+1$.

Para ello, describamos una traza de la computación $P(n+1)$:

$$P \equiv \boxed{\begin{array}{ll} & IF\ X \neq 0\ GOTO\ A \\ & Z \longleftarrow Z+1 \\ & IF\ Z \neq 0\ GOTO\ E \\ [A] & X \longleftarrow X-1 \\ & Y \longleftarrow Y+1 \\ & IF\ X \neq 0\ GOTO\ A \end{array}} \qquad \boxed{\begin{array}{l} C_0 = (1, \{X = n+1,\ Y = 0,\ Z = 0\}) \\ C_1 = (4, \{X = n+1,\ Y = 0,\ Z = 0\}) \\ C_2 = (5, \{X = n,\ Y = 0,\ Z = 0\}) \\ C_3 = (6, \{X = n,\ Y = 1,\ Z = 0\}) \\ \quad \vdots \qquad\qquad \vdots \qquad\qquad \vdots \end{array}}$$

Obsérvese que $I_1 = I_6$ y que, por tanto, la configuración C_3 es "similar" a $C_0' = (1, \{X = n,\ Y = 0,\ Z = 0\})$, que es la configuración inicial de $P(n)$.

- ⋆ La única diferencia radica en lo siguiente: en el estado de C_3 el valor de Y es 1 mientras que en el estado de C_0' el valor de Y es 0.
- ⋆ En la ejecución de P el valor de la variable Y nunca disminuye.
- ⋆ Luego la computación a partir de C_3 (cuyo resultado es $P(n+1)$) será similar a la computación a partir de C_0' (cuyo resultado es $P(n)$), incrementado en 1 el resultado
- ⋆ Por tanto, $P(n+1) = P(n) + 1 \overset{h.i.}{=} n+1$.

GOTO-computabilidad de la función sucesor en $\mathbb{N}$

La **función sucesor en $\mathbb{N}$** es la aplicación S de $\mathbb{N}$ en $\mathbb{N}$ definida como sigue: $S(n) = n \cup \{n\}$, para cada $n \in \mathbb{N}$. Notaremos $S(n) = n + 1$.

Consideremos el siguiente programa GOTO:

$$P_S \equiv \boxed{\begin{array}{ll} & IF\ X \neq 0\ GOTO\ A \\ & Y \longleftarrow Y + 1 \\ & Z \longleftarrow Z + 1 \\ & IF\ Z \neq 0\ GOTO\ E \\ [A] & X \longleftarrow X - 1 \\ & Y \longleftarrow Y + 1 \\ & IF\ X \neq 0\ GOTO\ A \\ & Y \longleftarrow Y + 1 \end{array}}$$

Vamos a **verificar formalmente** que la función de aridad 1 que calcula el programa P_S es la función sucesor en $\mathbb{N}$; es decir, vamos a probar que:

$$\forall a \in \mathbb{N}\ (\llbracket P_S \rrbracket^{(1)}(a) = S(a))$$

Verificación formal del programa el programa P_S

Proposición: Para cada número natural $a \in \mathbb{N}$ se verifica $P_S(a) = S(a)$.

Demostración: Por **inducción débil**.

1. Caso base: $a = 0$.

Hemos de probar que $P_S(0) = S(0) = 1$. Para ello, hallemos una traza de la computación $P_S(0)$:

$$P_S \equiv \left|\begin{array}{cll} 1 & & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \\ 2 & & Y \longleftarrow Y + 1 \\ 3 & & Z \longleftarrow Z + 1 \\ 4 & & \mathit{IF}\ Z \neq 0\ \mathit{GOTO}\ E \\ 5 & [A] & X \longleftarrow X - 1 \\ 6 & & Y \longleftarrow Y + 1 \\ 7 & & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \\ 8 & & Y \longleftarrow Y + 1 \end{array}\right|$$

$$\begin{array}{l} C_0 = (1, \{X_1 = 0,\ Y = 0,\ Z = 0\}) \\ C_1 = (2, \{X_1 = 0,\ Y = 0,\ Z = 0\}) \\ C_2 = (3, \{X_1 = 0,\ Y = 1,\ Z = 0\}) \\ C_3 = (4, \{X_1 = 0,\ Y = 1,\ Z = 1\}) \\ C_4 = (9, \{X_1 = 0,\ Y = 1,\ Z = 1\}) \end{array}$$

Por tanto, se tiene que $P_S(0) = 1 = S(0)$.

Verificación formal del programa el programa P_S

2. Paso inductivo: Sea $a \in \mathbb{N}$ y supongamos que $P_S(a) = S(a)$. Hemos de probar que $P_S(a+1) = S(a+1)$.

Para ello, describamos una traza de la computación $P_S(a+1)$:

$$P_S \equiv$$

1		$IF\ X \neq 0\ GOTO\ A$
2		$Y \longleftarrow Y + 1$
3		$Z \longleftarrow Z + 1$
4		$IF\ Z \neq 0\ GOTO\ E$
5	$[A]$	$X \longleftarrow X - 1$
6		$Y \longleftarrow Y + 1$
7		$IF\ X \neq 0\ GOTO\ A$
8		$Y \longleftarrow Y + 1$

$$C_0 = (1, \{X_1 = a+1,\ Y = 0,\ Z = 0\})$$
$$C_1 = (5, \{X_1 = a+1,\ Y = 0,\ Z = 0\})$$
$$C_2 = (6, \{X_1 = a,\ Y = 0,\ Z = 0\})$$
$$C_3 = (7, \{X_1 = a,\ Y = 1,\ Z = 0\})$$
$$\vdots \quad \vdots \quad \vdots$$

Ahora bien:

- Las instrucciones I_1 e I_7 son idénticas.
- La computación a partir de C_3 equivale a la computación $P_S(a)$ pero incrementando en 1 el resultado final (ya que en C_3 el valor de Y es 1 en lugar de 0).

Por tanto: $P_S(a+1) = P_S(a) + 1 \overset{h.i}{=} S(a) + 1 = S(a+1)$.

Predicados GOTO-computables

Recuérdese que un **predicado** sobre $\mathbb{N}$ de aridad $k \geq 1$ es una **función total booleana** sobre $\mathbb{N}^k$; es decir, una aplicación de $\mathbb{N}^k$ en $\{0, 1\}$.

Si $\theta(\vec{x})$ es un predicado sobre $\mathbb{N}$ de aridad $k \geq 1$ y $\theta(\vec{x}) = 1$ (resp. $\theta(\vec{x}) = 0$) entonces diremos que el predicado $\theta(\vec{x})$ es **verdadero** (resp. **falso**) sobre $\vec{x}$.

Un predicado es GOTO-computable **sii** lo es la función que lo define.

Un ejemplo: Predicado binario de igualdad en $\mathbb{N}$. $(\theta(a, b) = 1$ **sii** $a = b)$. Vamos a demostrar que este predicado es GOTO-computable.

Consideremos el siguiente programa GOTO:

$$P_{=} \equiv$$

$[C]$	$IF\ X_1 \neq 0\ GOTO\ A$
	$IF\ X_2 \neq 0\ GOTO\ D$
	$Y \longleftarrow Y + 1$
$[D]$	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ E$
$[A]$	$IF\ X_2 \neq 0\ GOTO\ B$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ E$
$[B]$	$X_1 \longleftarrow X_1 - 1$
	$X_2 \longleftarrow X_2 - 1$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ C$

Vamos a **verificar formalmente** que la función de aridad 2 que calcula el programa $P_{=}$ es el predicado binario de igualdad en $\mathbb{N}$; es decir, vamos a probar que:

$$\forall a, b \in \mathbb{N}\ (\llbracket P_{=} \rrbracket^{(2)}(a, b) = 1 \Leftrightarrow a = b)$$

Verificación formal del programa el programa $P_=$

Proposición: Para cada $a, b \in \mathbb{N}$ se verifica $P_=(a, b) = 1 \Leftrightarrow a = b$.

Demostración: Por **inducción débil** sobre la variable a.

1. Caso base: $a = 0$.

Hemos de probar que $P_=(0, b) = 1 \Leftrightarrow 0 = b$, para cada $b \in \mathbb{N}$. Para ello, sea $b \in \mathbb{N}$ y hallemos una traza de la computación $P_=(0, b)$:

$P_= \equiv$

1	$[C]$	$IF\ X_1 \neq 0\ GOTO\ A$
2		$IF\ X_2 \neq 0\ GOTO\ D$
3		$Y \longleftarrow Y + 1$
4	$[D]$	$Z \longleftarrow Z + 1$
5		$IF\ Z \neq 0\ GOTO\ E$
6	$[A]$	$IF\ X_2 \neq 0\ GOTO\ B$
7		$Z \longleftarrow Z + 1$
8		$IF\ Z \neq 0\ GOTO\ E$
9	$[B]$	$X_1 \longleftarrow X_1 - 1$
10		$X_2 \longleftarrow X_2 - 1$
11		$Z \longleftarrow Z + 1$
12		$IF\ Z \neq 0\ GOTO\ C$

Subcaso $b = 0$

$C_0 = (1, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_1 = (2, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_2 = (3, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_3 = (4, \{X_1 = 0,\ X_2 = 0,\ Y = 1,\ Z = 0\})$
$C_4 = (5, \{X_1 = 0,\ X_2 = 0,\ Y = 1,\ Z = 1\})$
$C_5 = (13, \{X_1 = 0,\ X_2 = b,\ Y = 1,\ Z = 1\})$

Subcaso $b \neq 0$

$C_0 = (1, \{X_1 = 0,\ X_2 = b,\ Y = 0,\ Z = 0\})$
$C_1 = (2, \{X_1 = 0,\ X_2 = b,\ Y = 0,\ Z = 0\})$
$C_2 = (4, \{X_1 = 0,\ X_2 = b,\ Y = 1,\ Z = 0\})$
$C_3 = (5, \{X_1 = 0,\ X_2 = b,\ Y = 0,\ Z = 1\})$
$C_4 = (13, \{X_1 = 0,\ X_2 = b,\ Y = 0,\ Z = 1\})$

Por tanto, se tiene que $P_=(0, b) = 1 \Leftrightarrow 0 = b$.

Verificación formal del programa el programa $P_=$

2. Paso inductivo: Sea $a \in \mathbb{N}$ y supongamos que $P_=(a,b) = 1 \Leftrightarrow a = b$, para cada $b \in \mathbb{N}$. Hemos de probar que $P_=(a+1,c) = 1 \Leftrightarrow a+1 = c$, para cada $c \in \mathbb{N}$.

Para ello, sea $c \in \mathbb{N}$ y describamos una traza de la computación $P_=(a+1,c)$.

$P_= \equiv$

1	$[C]$	$IF\ X_1 \neq 0\ GOTO\ A$
2		$IF\ X_2 \neq 0\ GOTO\ D$
3		$Y \longleftarrow Y + 1$
4	$[D]$	$Z \longleftarrow Z + 1$
5		$IF\ Z \neq 0\ GOTO\ E$
6	$[A]$	$IF\ X_2 \neq 0\ GOTO\ B$
7		$Z \longleftarrow Z + 1$
8		$IF\ Z \neq 0\ GOTO\ E$
9	$[B]$	$X_1 \longleftarrow X_1 - 1$
10		$X_2 \longleftarrow X_2 - 1$
11		$Z \longleftarrow Z + 1$
12		$IF\ Z \neq 0\ GOTO\ C$

Subcaso $c = 0$

$C_0 = (1, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_1 = (6, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_2 = (7, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_3 = (8, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 1\})$
$C_4 = (13, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 1\})$

Subcaso $c \neq 0$

$C_0 = (1, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_1 = (6, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_2 = (9, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_3 = (10, \{X_1 = a,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_4 = (11, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 0\})$
$C_5 = (12, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 1\})$
$C_6 = (1, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 1\})$

- En el **subcaso $c = 0$**, la computación devuelve 0.
- En el **subcaso $c \neq 0$**, la computación $P_=(a+1,c)$ **coincide** con la computación $P_=(a,c-1)$. Luego, $P_=(a+1,c) = 1 \Leftrightarrow P_=(a,c-1) = 1$. Pero por h.i. $P_=(a,c-1) = 1 \Leftrightarrow a = c-1$. En consecuencia, $P_=(a+1,c) = 1 \Leftrightarrow a = c-1 \Leftrightarrow a+1 = c$.

Predicado binario "menor o igual" en $\mathbb{N}$

El predicado binario "**menor o igual**" en $\mathbb{N}$ es el siguiente: $\theta((a,b)) = 1 \Leftrightarrow a \leq b$.

Vamos a demostrar que este predicado es GOTO-computable.

Consideremos el siguiente programa GOTO:

$$P_{\leq} \equiv$$

$[C]$	$IF\ X_1 \neq 0\ GOTO\ A$
	$Y \longleftarrow Y + 1$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ E$
$[A]$	$IF\ X_2 \neq 0\ GOTO\ B$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ E$
$[B]$	$X_1 \longleftarrow X_1 - 1$
	$X_2 \longleftarrow X_2 - 1$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ C$

Vamos a **verificar formalmente** que la función de aridad 2 que calcula el programa $P_{\leq}$ es el predicado "menor o igual" en $\mathbb{N}$; es decir, vamos a probar que:

$$\forall a, b \in \mathbb{N}\ (\llbracket P_{\leq} \rrbracket^{(2)}(a,b) = 1 \Leftrightarrow a \leq b)$$

Verificación formal del programa el programa $P_{\leq}$

Proposición: Para cada $a, b \in \mathbb{N}$ se verifica $P_{\leq}(a, b) = 1 \Leftrightarrow a \leq b$.

Demostración: Por **inducción débil** sobre la variable a.

1. Caso base: $a = 0$.

Hemos de probar que $P_{\leq}(0, b) = 1 \Leftrightarrow 0 \leq b$, para cada $b \in \mathbb{N}$; es decir, que $P_{\leq}(0, b) = 1$, para cada $b \in \mathbb{N}$. Para ello, sea $b \in \mathbb{N}$ y hallemos una traza de la computación $P_{\leq}(0, b)$:

$P_{\leq} \equiv$

1	[C]	*IF* $X_1 \neq 0$ *GOTO A*
2		$Y \longleftarrow Y + 1$
3		$Z \longleftarrow Z + 1$
4		*IF* $Z \neq 0$ *GOTO E*
5	[A]	*IF* $X_2 \neq 0$ *GOTO B*
6		$Z \longleftarrow Z + 1$
7		*IF* $Z \neq 0$ *GOTO E*
8	[B]	$X_1 \longleftarrow X_1 - 1$
9		$X_2 \longleftarrow X_2 - 1$
10		$Z \longleftarrow Z + 1$
11		*IF* $Z \neq 0$ *GOTO C*

$C_0 = (1, \{X_1 = 0,\ X_2 = b,\ Y = 0,\ Z = 0\})$
$C_1 = (2, \{X_1 = 0,\ X_2 = b,\ Y = 0,\ Z = 0\})$
$C_2 = (3, \{X_1 = 0,\ X_2 = b,\ Y = 1,\ Z = 0\})$
$C_3 = (4, \{X_1 = 0,\ X_2 = b,\ Y = 1,\ Z = 1\})$
$C_4 = (12, \{X_1 = 0,\ X_2 = b,\ Y = 1,\ Z = 1\})$

Por tanto, se tiene que $P_{\leq}(0, b) = 1$, para cada $b \in \mathbb{N}$.

Verificación formal del programa el programa $P_{\leq}$

2. Paso inductivo: Sea $a \in \mathbb{N}$ y supongamos que $P_{\leq}(a, b) = 1 \Leftrightarrow a \leq b$, para cada $b \in \mathbb{N}$. Hemos de probar que $P_{\leq}(a+1, c) = 1 \Leftrightarrow a+1 \leq c$, para cada $c \in \mathbb{N}$.

Para ello, sea $c \in \mathbb{N}$ y describamos una traza de la computación $P_{\leq}(a+1, c)$.

$P_{\leq} \equiv$

1	[C]	*IF* $X_1 \neq 0$ *GOTO A*
2		$Y \longleftarrow Y + 1$
3		$Z \longleftarrow Z + 1$
4		*IF* $Z \neq 0$ *GOTO E*
5	[A]	*IF* $X_2 \neq 0$ *GOTO B*
6		$Z \longleftarrow Z + 1$
7		*IF* $Z \neq 0$ *GOTO E*
8	[B]	$X_1 \longleftarrow X_1 - 1$
9		$X_2 \longleftarrow X_2 - 1$
10		$Z \longleftarrow Z + 1$
11		*IF* $Z \neq 0$ *GOTO C*

Subcaso $c = 0$

$C_0 = (1, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_1 = (5, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_2 = (6, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_3 = (7, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 1\})$
$C_4 = (13, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 1\})$

Subcaso $c \neq 0$

$C_0 = (1, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_1 = (5, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_2 = (8, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_3 = (9, \{X_1 = a,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_4 = (10, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 0\})$
$C_5 = (11, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 1\})$
$C_6 = (1, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 1\})$

- En el **subcaso** $c = 0$, la computación devuelve 0.
- En el **subcaso** $c \neq 0$, la computación $P_{\leq}(a+1, c)$ **coincide** con la computación $P_{\leq}(a, c-1)$. Luego, $P_{\leq}(a+1, c) = 1 \Leftrightarrow P_{\leq}(a, c-1) = 1$. Pero por h.i. $P_{\leq}(a, c-1) = 1 \Leftrightarrow a \leq c-1$. En consecuencia, $P_{\leq}(a+1, c) = 1 \Leftrightarrow a \leq c-1 \Leftrightarrow a+1 \leq c$.

Predicado binario "estríctamente menor" en $\mathbb{N}$

El predicado binario "**estríctamente menor**" en $\mathbb{N}$ es el siguiente:
$\theta((a, b) = 1 \Leftrightarrow a < b$.

Vamos a demostrar que este predicado es GOTO-computable.

Consideremos el siguiente programa GOTO:

$$P_{<} \equiv$$

$[C]$	$IF\ X_1 \neq 0\ GOTO\ A$
	$IF\ X_2 \neq 0\ GOTO\ D$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ E$
$[A]$	$IF\ X_2 \neq 0\ GOTO\ B$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ E$
$[B]$	$X_1 \longleftarrow X_1 - 1$
	$X_2 \longleftarrow X_2 - 1$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ C$
$[D]$	$Y \longleftarrow Y + 1$

Vamos a **verificar formalmente** que la función de aridad 2 que calcula el programa $P_{\leq}$ es el predicado binario "estríctamente menor" en $\mathbb{N}$; es decir, vamos a probar que:

$$\forall a, b \in \mathbb{N}\ (\llbracket P_{<} \rrbracket^{(2)}(a, b) = 1 \Leftrightarrow a < b)$$

Verificación formal del programa el programa $P_<$

Proposición: Para cada $a, b \in \mathbb{N}$ se verifica $P_<(a, b) = 1 \Leftrightarrow a < b$.

Demostración: Por **inducción débil** sobre la variable a.

1. Caso base: $a = 0$.

Hemos de probar que $P_<(0, b) = 1, \Leftrightarrow 0 < b$ para cada $b \in \mathbb{N}$. Para ello, sea $b \in \mathbb{N}$ y hallemos una traza de la computación $P_<(0, b)$:

$P_< \equiv$

1	$[C]$	$IF\ X_1 \neq 0\ GOTO\ A$
2		$IF\ X_2 \neq 0\ GOTO\ D$
3		$Z \longleftarrow Z + 1$
4		$IF\ Z \neq 0\ GOTO\ E$
5	$[A]$	$IF\ X_2 \neq 0\ GOTO\ B$
6		$Z \longleftarrow Z + 1$
7		$IF\ Z \neq 0\ GOTO\ E$
8	$[B]$	$X_1 \longleftarrow X_1 - 1$
9		$X_2 \longleftarrow X_2 - 1$
10		$Z \longleftarrow Z + 1$
11		$IF\ Z \neq 0\ GOTO\ C$
12	$[D]$	$Y \longleftarrow Y + 1$

Subcaso $b = 0$

$C_0 = (1, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_1 = (2, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_2 = (3, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 1\})$
$C_3 = (4, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 1\})$
$C_4 = (13, \{X_1 = 0,\ X_2 = 0,\ Y = 0,\ Z = 1\})$

Subcaso $b \neq 0$

$C_0 = (1, \{X_1 = 0\ X_2 = b,\ Y = 0,\ Z = 0\})$
$C_1 = (2, \{X_1 = 0,\ X_2 = b,\ Y = 0,\ Z = 0\})$
$C_2 = (12, X_1 = 0\ X_2 = b,\ Y = 0,\ Z = 0\})$
$C_3 = (13, \{X_1 = 0,\ X_2 = b,\ Y = 1,\ Z = 0\})$

Por tanto, se tiene que $P_<(0, b) = 1 \Leftrightarrow 0 < b$, para cada $b \in \mathbb{N}$.

Verificación formal del programa el programa $P_<$

2. Paso inductivo: Sea $a \in \mathbb{N}$ y supongamos que $P_<(a,b) = 1 \Leftrightarrow a < b$, para cada $b \in \mathbb{N}$. Hemos de probar que $P_<(a+1,c) = 1 \Leftrightarrow a+1 < c$, para cada $c \in \mathbb{N}$.

Para ello, sea $c \in \mathbb{N}$ y describamos una traza de la computación $P_<(a+1,c)$.

$P_< \equiv$

1	[C]	IF $X_1 \neq 0$ GOTO A
2		IF $X_2 \neq 0$ GOTO D
3		$Z \longleftarrow Z+1$
4		IF $Z \neq 0$ GOTO E
5	[A]	IF $X_2 \neq 0$ GOTO B
6		$Z \longleftarrow Z+1$
7		IF $Z \neq 0$ GOTO E
8	[B]	$X_1 \longleftarrow X_1 - 1$
9		$X_2 \longleftarrow X_2 - 1$
10		$Z \longleftarrow Z+1$
11		IF $Z \neq 0$ GOTO C
12	[D]	$Y \longleftarrow Y+1$

Subcaso $c = 0$

$C_0 = (1, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_1 = (5, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_2 = (6, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 0\})$
$C_3 = (7, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 1\})$
$C_4 = (13, \{X_1 = a+1,\ X_2 = 0,\ Y = 0,\ Z = 1\})$

Subcaso $c \neq 0$

$C_0 = (1, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_1 = (5, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_2 = (8, \{X_1 = a+1,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_3 = (9, \{X_1 = a,\ X_2 = c,\ Y = 0,\ Z = 0\})$
$C_4 = (10, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 0\})$
$C_5 = (11, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 1\})$
$C_6 = (1, \{X_1 = a,\ X_2 = c-1,\ Y = 0,\ Z = 1\})$

- En el **subcaso $c = 0$**, la computación devuelve 0.
- En el **subcaso $c \neq 0$**, la computación $P_<(a+1,c)$ **coincide** con la computación $P_<(a,c-1)$. Luego, $P_<(a+1,c) = 1 \Leftrightarrow P_<(a,c-1) = 1$. Pero por h.i. $P_<(a,c-1) = 1 \Leftrightarrow a < c-1$. En consecuencia, $P_<(a+1,c) = 1 \Leftrightarrow a < c-1 \Leftrightarrow a+1 < c$.

Macros GOTO: Salto incondicional

Una **macro** GOTO es un programa GOTO que realiza una tarea determinada.

- Supongamos que en un programa P se encuentran las instrucciones:

$$Q \equiv \boxed{\begin{array}{rl} I_j: & Z \longleftarrow Z+1 \\ I_{j+1}: & IF\ Z \neq 0\ GOTO\ L \end{array}}$$

  * Su ejecución permite realizar un **salto incondicional** a la primera instrucción etiquetada por L (o **terminar** la ejecución de P).

  * Se trata de una macro GOTO que notaremos así: $GOTO\ L$.

$$\underbrace{GOTO\ L}_{macro} \Longrightarrow \underbrace{\left.\begin{array}{l} Z \longleftarrow Z+1 \\ IF\ Z \neq 0\ GOTO\ L \end{array}\right\}}_{\underline{una}\ expansión}$$

Macros GOTO: Poner a cero una variable

- Supongamos que en un programa P se encuentran las instrucciones:

$$Q \equiv \boxed{\begin{array}{lll} I_j: & [L] & V \longleftarrow V - 1 \\ I_{j+1}: & & \mathit{IF}\ V \neq 0\ \mathit{GOTO}\ L \end{array}}$$

  * Su ejecución pone a cero el valor de la variable V.
  * Notaremos esta macro así: $V \longleftarrow 0$.

$$\underbrace{V \longleftarrow 0}_{\mathit{macro}} \quad \Longrightarrow \quad \underbrace{\left\{\begin{array}{lll} & [L] & V \longleftarrow V - 1 \\ & & \mathit{IF}\ V \neq 0\ \mathit{GOTO}\ L \end{array}\right.}_{\underline{\mathit{una}}\ \mathit{expansión}}$$

Macros GOTO: asignación: $V \longleftarrow V'$

Tarea a realizar: **asignar a V el valor de V', conservando V' su valor**.

Una expansión de la macro $V \longleftarrow V'$ es:

$$
\begin{array}{ll}
[A_2] & V \longleftarrow V - 1 \\
 & IF\ V \neq 0\ GOTO\ A_2 \qquad \Big\} \ V \longleftarrow 0 \\
[A] & IF\ V' \neq 0\ GOTO\ B \\
 & Z_2 \longleftarrow Z_2 + 1 \\
 & IF\ Z_2 \neq 0\ GOTO\ C \qquad \Big\} \ \texttt{GOTO}\ C \\
[B] & V' \longleftarrow V' - 1 \\
 & V \longleftarrow V + 1 \\
 & Z \longleftarrow Z + 1 \\
 & Z_3 \longleftarrow Z_3 + 1 \\
 & IF\ Z_3 \neq 0\ GOTO\ A \qquad \Big\} \ \texttt{GOTO}\ A \\
[C] & IF\ Z \neq 0\ GOTO\ D \\
 & Z_4 \longleftarrow Z_4 + 1 \\
 & IF\ Z_4 \neq 0\ GOTO\ L \qquad \Big\} \ \texttt{GOTO}\ L \\
[D] & Z \longleftarrow Z - 1 \\
 & V' \longleftarrow V' + 1 \\
 & Z_5 \longleftarrow Z_5 + 1 \\
 & IF\ Z_5 \neq 0\ GOTO\ C \qquad \Big\} \ \texttt{GOTO}\ C
\end{array}
$$

Dicha expansión consta de **17 instrucciones**.

Macros GOTO: condicionales

Sea $\theta(V_1, \ldots, V_k)$ un **predicado** que es **computable** por un programa GOTO P.

$$\underset{\textit{macro}}{\textit{IF } \theta(V_1, \ldots, V_k) \neq 0 \textit{ GOTO } L} \quad \rightsquigarrow \quad \underset{\text{expansión}}{\begin{array}{l} Z \longleftarrow P(V_1, \ldots, V_k) \\ \textit{IF } Z \neq 0 \textit{ GOTO } L \end{array}}$$

Ejemplo: "*IF* $V = 0$ GOTO L" **es una macro** GOTO

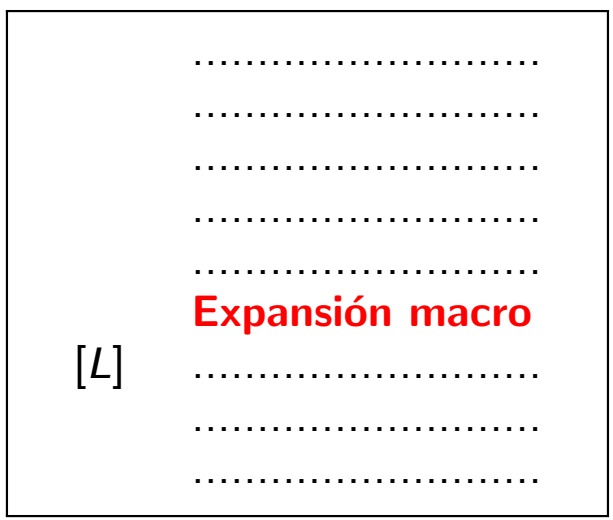
$$V = 0 \text{ es el predicado } \theta(x) = \begin{cases} 1 & \texttt{si} \quad x = 0 \\ 0 & \texttt{si} \quad x \neq 0 \end{cases}$$

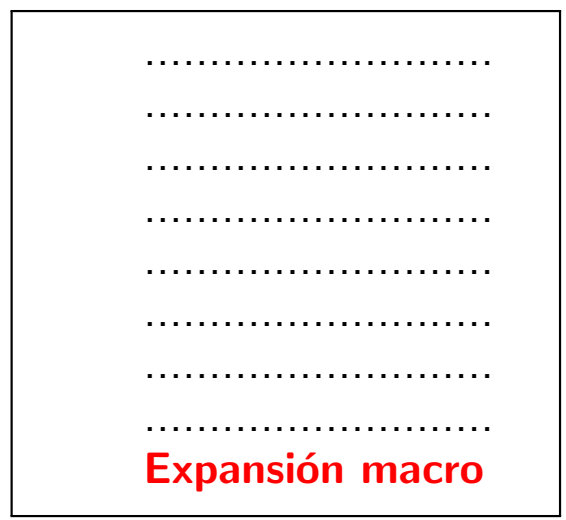
donde $\theta(x)$ es GOTO–computable: $\begin{cases} \textit{IF } X \neq 0 \textit{ GOTO } E \\ Y \longleftarrow Y + 1 \end{cases}$

Inserción de una macro en un programa GOTO

Para **insertar una macro en un programa** GOTO:

* Se considera **una** expansión de la macro.
* Las variables de esa expansión (distintas de las que aparecen en la macro) y las etiquetas, se renombran para que sean diferentes de las usadas en el programa.
* Si existe alguna instrucción tras el lugar donde se inserte la macro:
  - Las etiquetas de salida de la expansión (si las hubiere) se renombran por la etiqueta de la primera instrucción que sigue a la expansión (si ésta no estuviera etiquetada, se le asignaría una nueva etiqueta L).





Ejemplo del uso de las macros: la función suma en $\mathbb{N}$

¿**Quién es** la **función suma en** $\mathbb{N}$?

En el marco de la Teoría de Conjuntos se demuestra que **existe** una **única** función total, f, de $\mathbb{N} \times \mathbb{N}$ en $\mathbb{N}$ que satisface las dos condiciones siguientes:

* $f(a, 0) = a$, para cada número natural $a \in \mathbb{N}$.
* $f(a, b + 1) = S(f(a, b)) = f(a, b) + 1$, para cada $a, b \in \mathbb{N}$.

Esa función total f se denomina **función suma en** $\mathbb{N}$. Notaremos $f = Sum$ y $Sum(a, b) = a + b$.

Obsérvese que la función suma en $\mathbb{N}$ está definida, formalmente, a partir de la función sucesor S en $\mathbb{N}$.

Usando macros `GOTO`, vamos a demostrar que la función suma en $\mathbb{N}$ es una función `GOTO`-computable.

GOTO-computabilidad de la función suma en $\mathbb{N}$

Consideremos el siguiente programa GOTO:

$$P_+ \equiv \boxed{\begin{array}{ll} & Y \longleftarrow X_1 \\ & Z \longleftarrow X_2 \\ [B] & IF\ Z \neq 0\ GOTO\ A \\ & GOTO\ E \\ [A] & Z \longleftarrow Z - 1 \\ & Y \longleftarrow Y + 1 \\ & GOTO\ B \end{array}}$$

Vamos a **verificar formalmente** que la función de aridad 2 que calcula el programa P_+ es la función suma en $\mathbb{N}$; es decir, vamos a probar que:

$$\forall a, b \in \mathbb{N}\ (\llbracket P_+ \rrbracket^{(2)}(a, b) = a + b)$$

Verificación formal del programa el programa P_+

Proposición: Para cada par de números naturales $a, b \in \mathbb{N}$ se verifica $P_+(a, b) = a + b$.

Demostración: Por **inducción débil** sobre la variable b

1. Caso base: $b = 0$.

Hemos de probar que $P_+(a, 0) = a$, para cada $a \in \mathbb{N}$. Para ello, sea $a \in \mathbb{N}$ y hallemos una traza de la computación $P_+(a, 0)$:

$$P_+ \equiv \left|\begin{array}{lll} (1\text{-}17) & & Y \longleftarrow X_1 \\ (18\text{-}34) & & Z \longleftarrow X_2 \\ (35) & [B] & IF\ Z \neq 0\ GOTO\ A \\ (36\text{-}37) & & GOTO\ E \\ (38) & [A] & Z \longleftarrow Z - 1 \\ (39) & & Y \longleftarrow Y + 1 \\ (40\text{-}41) & & GOTO\ B \end{array}\right|$$

$$\begin{array}{ll} C_0 = & (1, \{X_1 = a,\ X_2 = 0,\ Y = 0,\ Z = 0\}) \\ C_{q_1} = & (18, \{X_1 = a,\ X_2 = 0,\ Y = a,\ Z = 0\}) \\ C_{q_2} = & (35, \{X_1 = a,\ X_2 = 0,\ Y = a,\ Z = 0\}) \\ C_{q_2+1} = & (36, \{X_1 = a,\ X_2 = 0,\ Y = a,\ Z = 0\}) \\ C_{q_3} = & (42, \{X_1 = a,\ X_2 = 0,\ Y = a,\ Z = 0\}) \end{array}$$

Por tanto, se tiene que $P_+(a, 0) = a$.

Verificación formal del programa el programa P_+

2. Paso inductivo: Sea $b \in \mathbb{N}$ y supongamos que $P_+(a, b) = a + b$, para cada $a \in \mathbb{N}$.
Hemos de probar que $P_+(a, b+1) = a + (b+1)$, para cada $a \in \mathbb{N}$.

Para ello, describamos una traza de la computación $P_+(a, b+1)$:

$P_+ \equiv$

(1-17)		$Y \longleftarrow X_1$
(18-34)		$Z \longleftarrow X_2$
(35)	$[B]$	$IF\ Z \neq 0\ GOTO\ A$
(36-37)		$GOTO\ E$
(38)	$[A]$	$Z \longleftarrow Z - 1$
(39)		$Y \longleftarrow Y + 1$
(40-41)		$GOTO\ B$

$$
\begin{aligned}
C_0 &= (1, \{X_1 = a,\ X_2 = b+1,\ Y = 0,\ Z = 0\}) \\
C_{q_1} &= (18, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = 0\}) \\
C_{q_2} &= (35, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = b+1\}) \\
C_{q_2+1} &= (38, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = b+1\}) \\
C_{q_2+2} &= (39, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = b\}) \\
C_{q_2+3} &= (40, \{X_1 = a,\ X_2 = b+1,\ Y = a+1,\ Z = b\}) \\
C_{q_2+5} &= (35, \{X_1 = a,\ X_2 = b+1,\ Y = a+1,\ Z = b\}) \\
&\ \ \vdots \qquad\qquad \vdots \qquad\qquad \vdots
\end{aligned}
$$

A lo largo de cualquier computación:

- El valor de Y sólo puede aumentar (desde el valor inicial de X_1).
- El valor de Z sólo puede disminuir (desde el valor inicial de X_2).
- Una vez que se llegue a I_{35} ya no se ejecutará ninguna instrucción anterior $I_1, \ldots, I_{34}$.

Verificación formal del programa el programa P_+

$$P_+ \equiv \left|\begin{array}{lll} (1\text{-}17) & & Y \longleftarrow X_1 \\ (18\text{-}34) & & Z \longleftarrow X_2 \\ (35) & [B] & \mathit{IF}\ Z \neq 0\ \mathit{GOTO}\ A \\ (36\text{-}37) & & \mathit{GOTO}\ E \\ (38) & [A] & Z \longleftarrow Z - 1 \\ (39) & & Y \longleftarrow Y + 1 \\ (40\text{-}41) & & \mathit{GOTO}\ B \end{array}\right.$$

$$\begin{array}{rcl} C_0 & = & (1, \{X_1 = a,\ X_2 = b+1,\ Y = 0,\ Z = 0\}) \\ C_{q_1} & = & (18, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = 0\}) \\ C_{q_2} & = & (35, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = b+1\}) \\ C_{q_2+1} & = & (38, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = b+1\}) \\ C_{q_2+2} & = & (39, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = b\}) \\ C_{q_2+3} & = & (40, \{X_1 = a,\ X_2 = b+1,\ Y = a+1,\ Z = b\}) \\ C_{q_2+5} & = & (35, \{X_1 = a,\ X_2 = b+1,\ Y = a+1,\ Z = b\}) \\ \vdots & \vdots & \vdots \end{array}$$

La computación obtenida a partir de la configuración $C = (35, \{X_1 = \alpha, X_2 = \beta, Y = \alpha + \gamma, Z = \beta - \delta\})$ será de parada y el resultado es $P_+(\alpha, \beta - \delta) + \gamma$.

El resultado obtenido de la computación a partir de $C_{q_2} = (35, \{X_1 = a,\ X_2 = b+1,\ Y = a,\ Z = b+1\})$ será $\boxed{P_+(a, b+1)}$ y el obtenido a partir de $C_{q_2+5} = (35, \{X_1 = a,\ X_2 = b+1,\ Y = a+1,\ Z = b\})$ será $\boxed{P_+(a, b) + 1}$.

Puesto que C_{q_2+5} se obtiene a partir de C_{q_2}, resultará que $P_+(a, b+1) = P_+(a, b) + 1$.

En consecuencia: $P_+(a, b+1) = P_+(a, b) + 1 \overset{h.i.}{=} (a + b) + 1 \overset{asociat.}{=} a + (b + 1)$.

Ejemplo del uso de las macros: la función producto en $\mathbb{N}$

¿**Quién es** la **función producto en** $\mathbb{N}$?

En la Teoría de Conjuntos se demuestra que **existe** una **única** función total, g, de $\mathbb{N} \times \mathbb{N}$ en $\mathbb{N}$ que satisface las dos condiciones siguientes:

* $g(a, 0) = 0$, para cada número natural $a \in \mathbb{N}$.
* $g(a, b + 1) = g(a, b) + a$, para cada par de números naturales $a, b \in \mathbb{N}$.

Esa función total g se denomina **función producto en** $\mathbb{N}$. Notaremos $g = Prod$ y $Prod(a, b) = a \cdot b$.

Obsérvese que la función producto en $\mathbb{N}$ está definida, formalmente, a partir de la función suma en $\mathbb{N}$.

Usando macros `GOTO`, vamos a demostrar que la función producto en $\mathbb{N}$ es una función `GOTO`-computable.

GOTO-computabilidad de la función producto en $\mathbb{N}$

Consideremos el siguiente programa GOTO:

$$P_\bullet \begin{cases} & Z_2 \longleftarrow X_2 \\ [B] & \mathit{IF}\ Z_2 \neq 0\ \mathit{GOTO}\ A \\ & \texttt{GOTO}\ E \\ [A] & Z_2 \longleftarrow Z_2 - 1 \\ & Z_1 \longleftarrow X_1 + Y \\ & Y \longleftarrow Z_1 \\ & \texttt{GOTO}\ B \end{cases}$$

Nota: $Z_1 \longleftarrow X_1 + Y$ es, a su vez, una macro GOTO que ejecuta el programa *Suma* con entrada (X_1, Y), asignando el resultado a la variable Z_1.

Ejercicio: Verificar formalmente que la función de aridad 2 que calcula el programa $P_\bullet$ es la función producto. Es decir, probar que

$$\forall a, b \in \mathbb{N}\ (\llbracket P_\bullet \rrbracket^{(2)}(a, b) = a \cdot b)$$

Aplíquese inducción débil sobre la segunda variable.

GOTO-computabilidad de las funciones proyección en $\mathbb{N}$

Sean m y k números naturales tales que $m \geq 1$ y $1 \leq k \leq m$.

La **función proyección k-ésima de aridad m**, que notaremos $\Pi_k^{(m)}$, es la aplicación de $\mathbb{N}^m$ en $\mathbb{N}$ definida como sigue:

$$\Pi_k^{(m)}(a_1, \ldots, a_m) = a_k, \text{ para cada } (a_1, \ldots, a_m) \in \mathbb{N}^m$$

Consideremos el siguiente programa GOTO con macro:

$$P_k \equiv \boxed{Y \longleftarrow X_k}$$

Este programa se puede describir sin macro como sigue:

$$P_k \left\{ \begin{array}{ll} & \textit{IF } X_k \neq 0 \textit{ GOTO } A \\ & Z \longleftarrow Z + 1 \\ & \textit{IF } Z \neq 0 \textit{ GOTO } E \\ [A] & X_k \longleftarrow X_k - 1 \\ & Y \longleftarrow Y + 1 \\ & \textit{IF } X_k \neq 0 \textit{ GOTO } A \end{array} \right.$$

Obviamente, para cada número natural $m \geq k$ se verifica que la función de aridad m calculada por el programa P_k es, precisamente, la función $\Pi_k^{(m)}$.

Es decir, $P_k(a_1, \ldots a_m) = x_k$, para cada $(a_1, \ldots, a_m) \in \mathbb{N}^m$

Cuestión: Para cada $r \in \mathbb{N}$ tal que $1 \leq r < k$, ¿cuál es la función de aridad r calculada por el programa P_k?