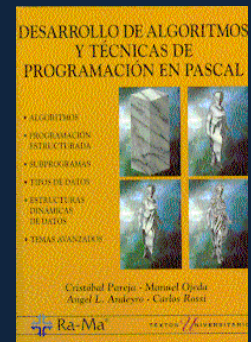


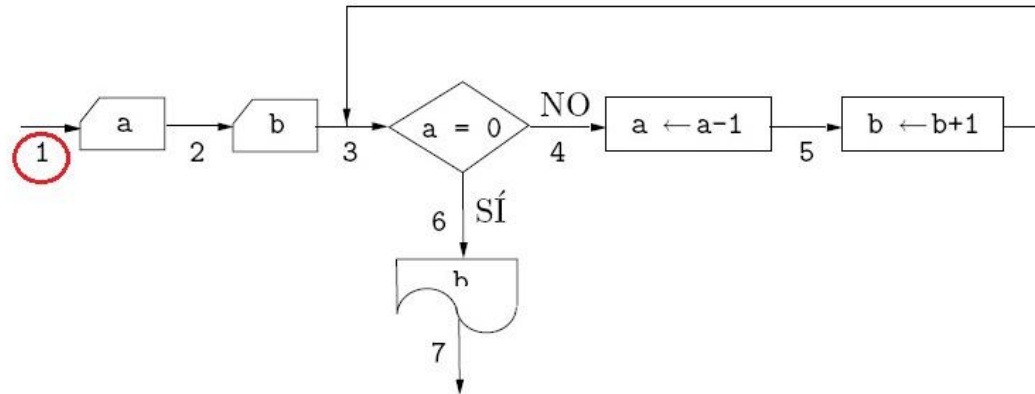
Problemas, algoritmos y programas (fragmento)

- **Concepto de algoritmo**
- **Una primera definición intuitiva:**
 - Idea, definición, partes (atención a la entrada y salida)
 - Características
 - Ejemplo: suma lenta (imperativo, funcional).
 - Aspectos nuevos: variables, estados y transiciones
- **Una definición formal (Modelo imperativo, von Neumann):**
 - Estados (E, I, F) y transiciones
- **Aspectos de interés:**
 - Especificaciones (predicados)
 - Corrección
 - Complejidad
 - Computabilidad
- **Bibliografía principal:**

Pareja et al., “Desarrollo de algoritmos [...]” cap. 1

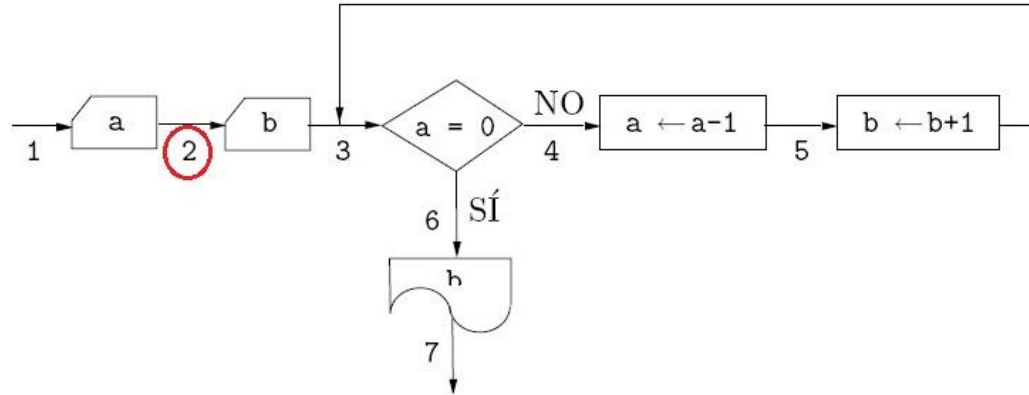


1. Suma lenta: diagrama de flujo (1/11)



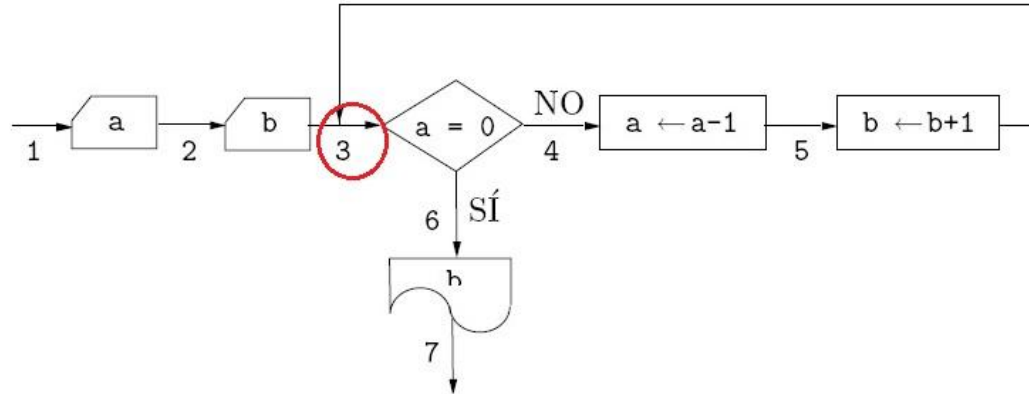
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (2/11)



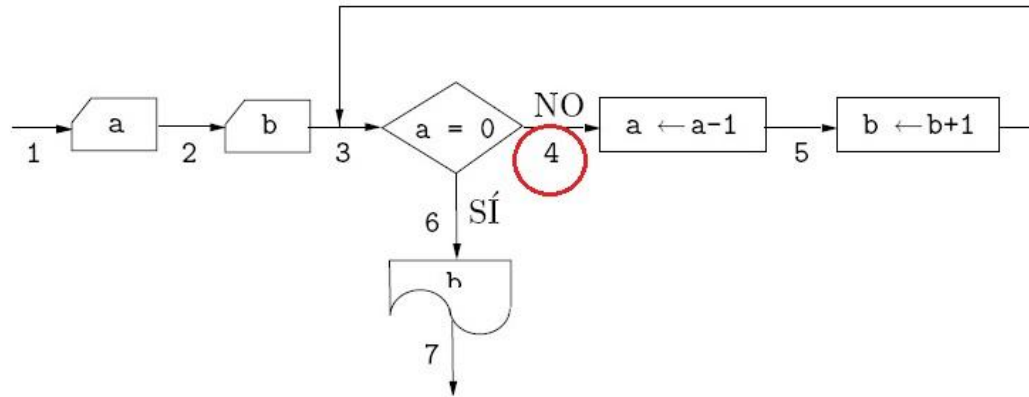
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (3/11)



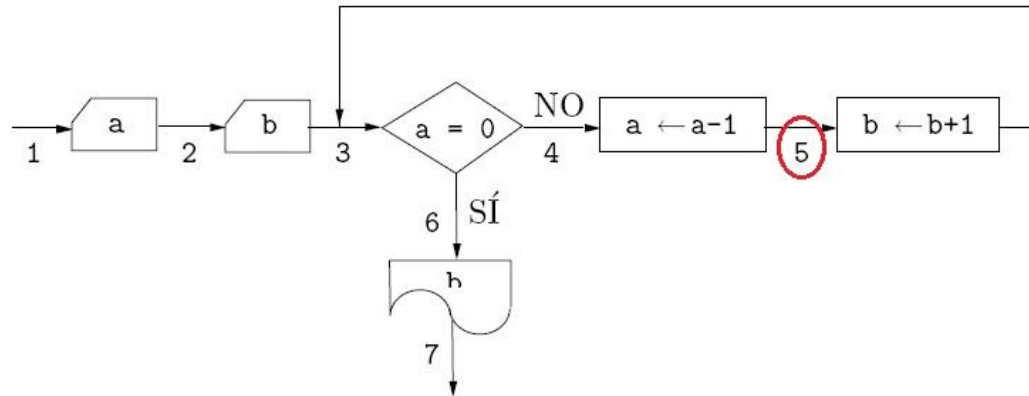
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (4/11)



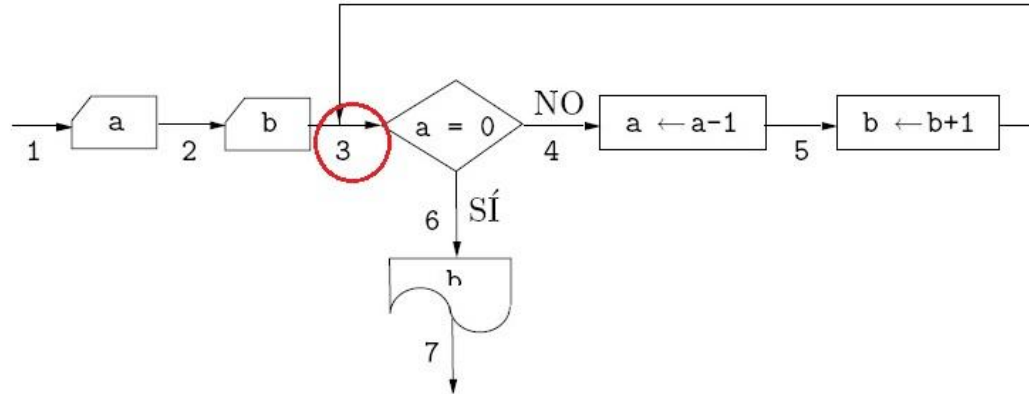
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[[]]	?	?
2	[3]	[[]]	2	?
3	[[]]	[[]]	2	3
4	[[]]	[[]]	2	3
5	[[]]	[[]]	1	3
3	[[]]	[[]]	1	4
4	[[]]	[[]]	1	4
5	[[]]	[[]]	0	4
3	[[]]	[[]]	0	5
6	[[]]	[[]]	0	5
7	[[]]	[5]	0	5

1. Suma lenta: diagrama de flujo (5/11)



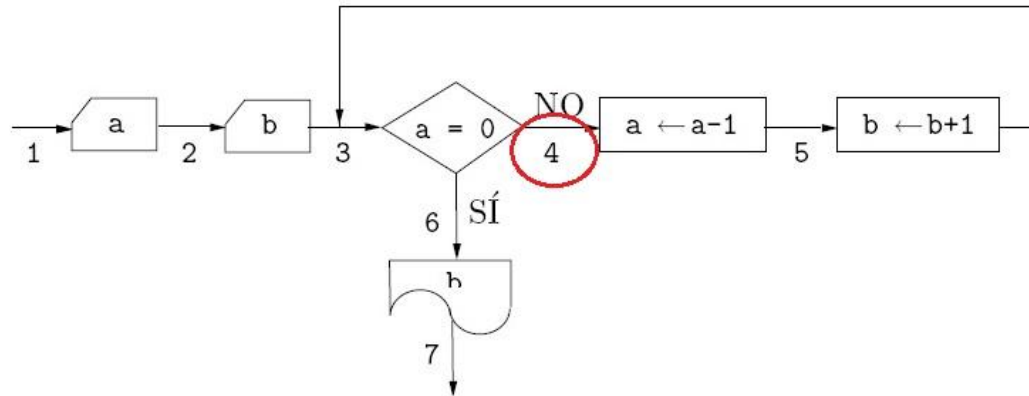
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (6/11)



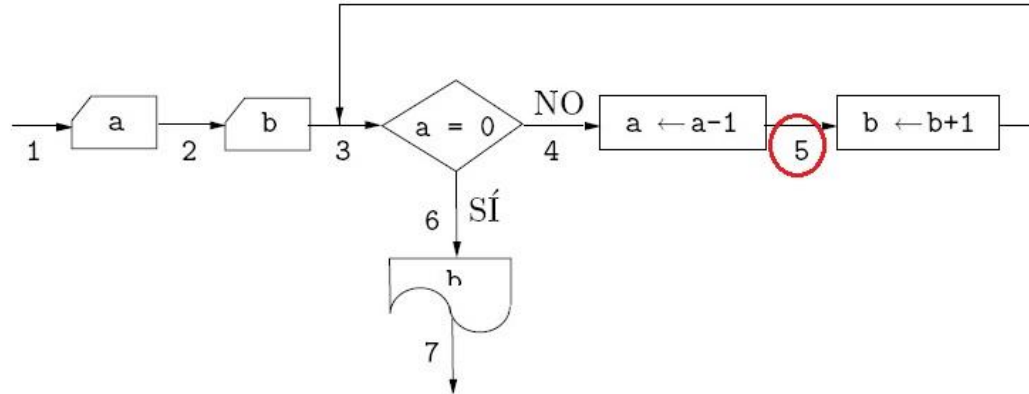
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (7/11)



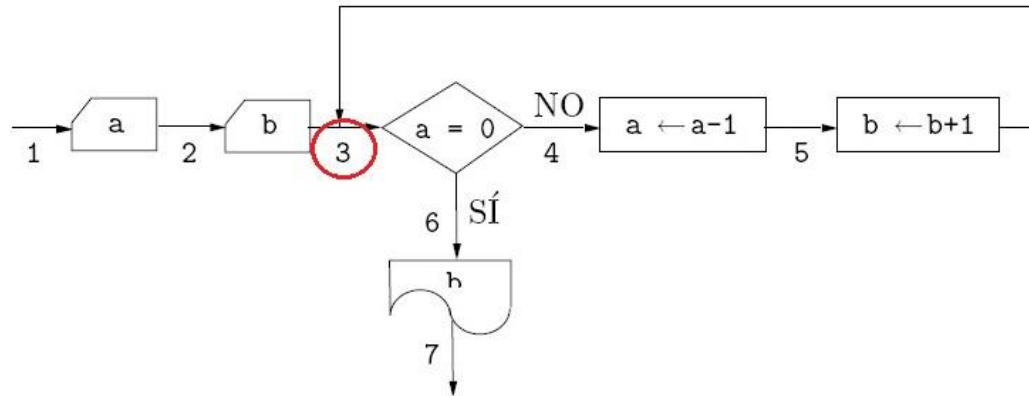
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (8/11)



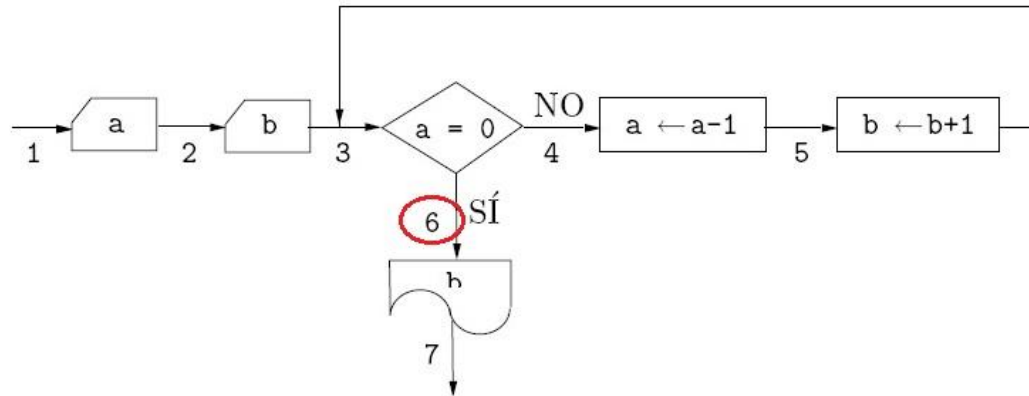
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (9/11)



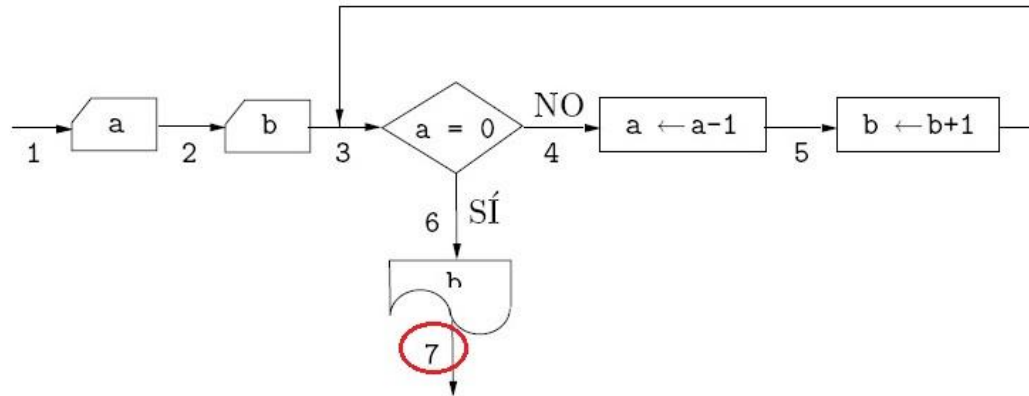
Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (10/11)



Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

1. Suma lenta: diagrama de flujo (11/11)



Posición	Datos pendientes	Resultados emitidos	Var a	Var b
1	[2, 3]	[]	?	?
2	[3]	[]	2	?
3	[]	[]	2	3
4	[]	[]	2	3
5	[]	[]	1	3
3	[]	[]	1	4
4	[]	[]	1	4
5	[]	[]	0	4
3	[]	[]	0	5
6	[]	[]	0	5
7	[]	[5]	0	5

Algoritmo:

(E, I, F, t)

- E

- I \subset E

- F \subset E

- t: E \rightarrow E : ...

$$E = \left\{ \begin{array}{l} \langle 1 ; [d_1, d_2] ; [] ; \text{---} \rangle \\ \langle 2 ; [d_2] ; [] ; a \equiv a_0 \rangle \\ \langle p ; [] ; [] ; a \equiv a_0, b \equiv b_0 \rangle \\ \langle 7 ; [] ; [r] ; \text{---} \rangle \end{array} \right\}$$

$$I = \{ \langle 1; [d_1, d_2]; []; \text{---} \rangle \}$$

$$F = \{ \langle 7; []; [r]; \text{---} \rangle \} \quad \forall e \in F, t(e) = e$$

$$\begin{aligned} t(\langle 1; [d_1, d_2]; []; \text{---} \rangle) &= \langle 2; [d_2]; []; a \equiv d_1 \rangle \\ t(\langle 2; [d_2]; []; a \equiv d_1 \rangle) &= \langle 3; []; []; a \equiv d_1, b \equiv d_2 \rangle \\ t(\langle 3; []; []; a \equiv a_0, b \equiv b_0 \rangle) &= \begin{cases} \langle 6; []; []; b \equiv b_0 \rangle & \text{si } a = 0 \\ \langle 4; []; []; a \equiv a_0, b \equiv b_0 \rangle & \text{si } a \neq 0 \end{cases} \\ t(\langle 4; []; []; a \equiv a_0, b \equiv b_0 \rangle) &= \langle 5; []; []; a \equiv a_0 - 1, b \equiv b_0 \rangle \\ t(\langle 5; []; []; a \equiv a_0, b \equiv b_0 \rangle) &= \langle 3; []; []; a \equiv a_0, b \equiv b_0 + 1 \rangle \\ t(\langle 6; []; []; b \equiv b_0 \rangle) &= \langle 7; []; [b_0]; \text{---} \rangle \\ t(\langle 7; []; [r]; \text{---} \rangle) &= \langle 7; []; [r]; \text{---} \rangle \end{aligned}$$

$$\dots \quad \forall e \in E, (\exists! e' = t(e) \in E) \quad \wedge \quad (\exists k \in \mathbb{N}: t^k(e) \in F)$$

2.a Especificación de un algoritmo

- “Descripción precisa del cometido del algoritmo”
- Formalmente, se expresa mediante la relación entre el estado previo y posterior:

$a, b \in \mathbb{Z}$

//Pre.: $a = M, b = B$

SUMA

//Post.: $a = M, b = B, c = A+B$

Variables

Prop. estado previo

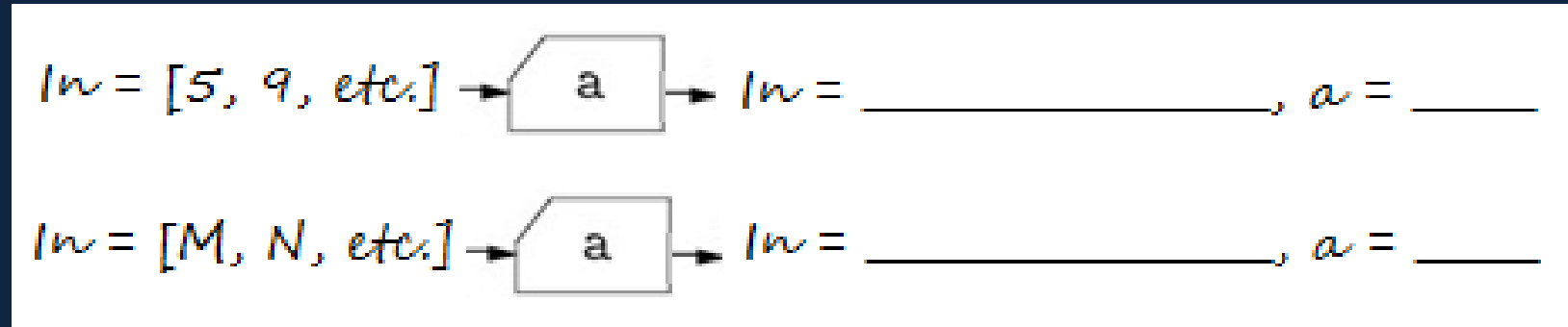
Algoritmo

Prop. estado posterior

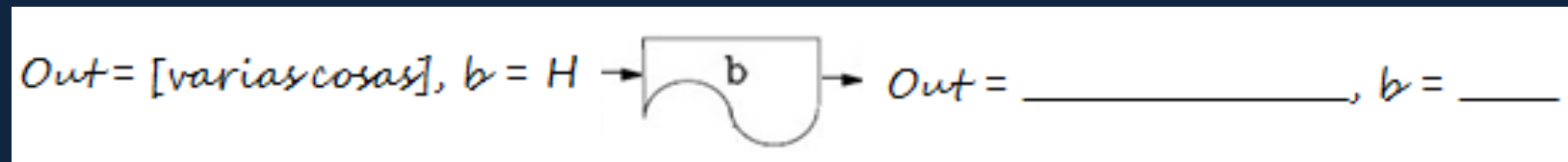
- Especificación: útil también para instrucciones o fragmentos de programa.

2.a Especificación: pre y post-condición

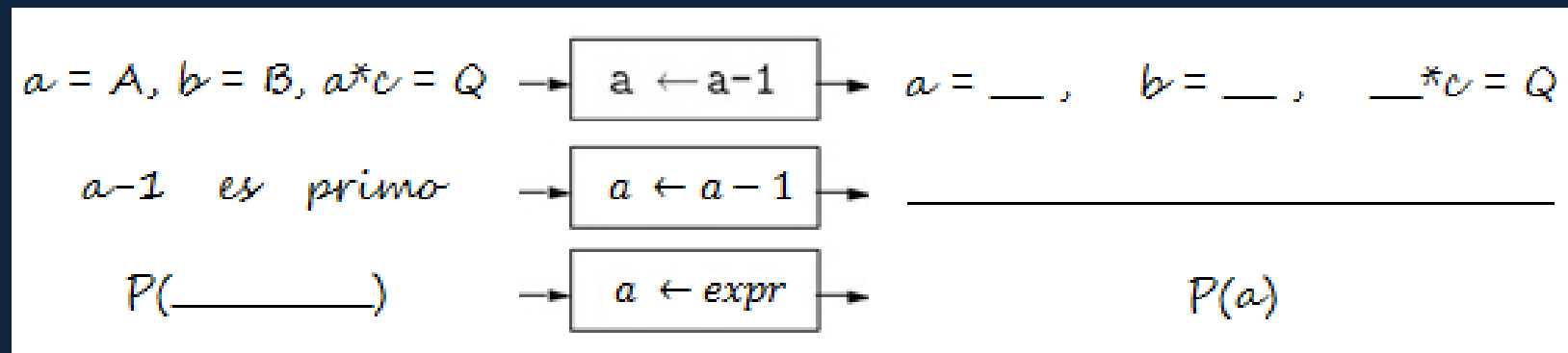
- Lectura:



- Escritura:



- Asignación:



2.a Especificación: pre y post-condición

- Ejercicios asignación:

(a)
// $x^2 + 6$ es primo
 $x \leftarrow x^2 + 6$
// _____

(b)
// $x^2 + 6$ es primo
 $x \leftarrow x^2 + 5$
// _____

(c)
// _____
 $x \leftarrow x^2 + 1$
// x es primo

(d)
// _____
 $x \leftarrow x + y$
// $x = A + B$

(e)
// _____
 $x \leftarrow 3x - 1$
// $x = 5$

(f)
// _____
 $x \leftarrow 3x - 1$
// $x^2 + 1$ es primo

2.a Especificación: pre y post-condición

- Ejercicios asignación:

(g)

// $x^2 + x + 8 = 0$

$x \leftarrow 3x - 1$

// _____

(pista)

Reescribir la precondición:

$$x \leftrightarrow (3x - 1 + 1)/3$$

(h)

// _____

$x \leftarrow x^2 - 4$

// $x = 5$

(h.1) Elige precondiciones

correctas entre las

$$x = 3$$

siguientes:

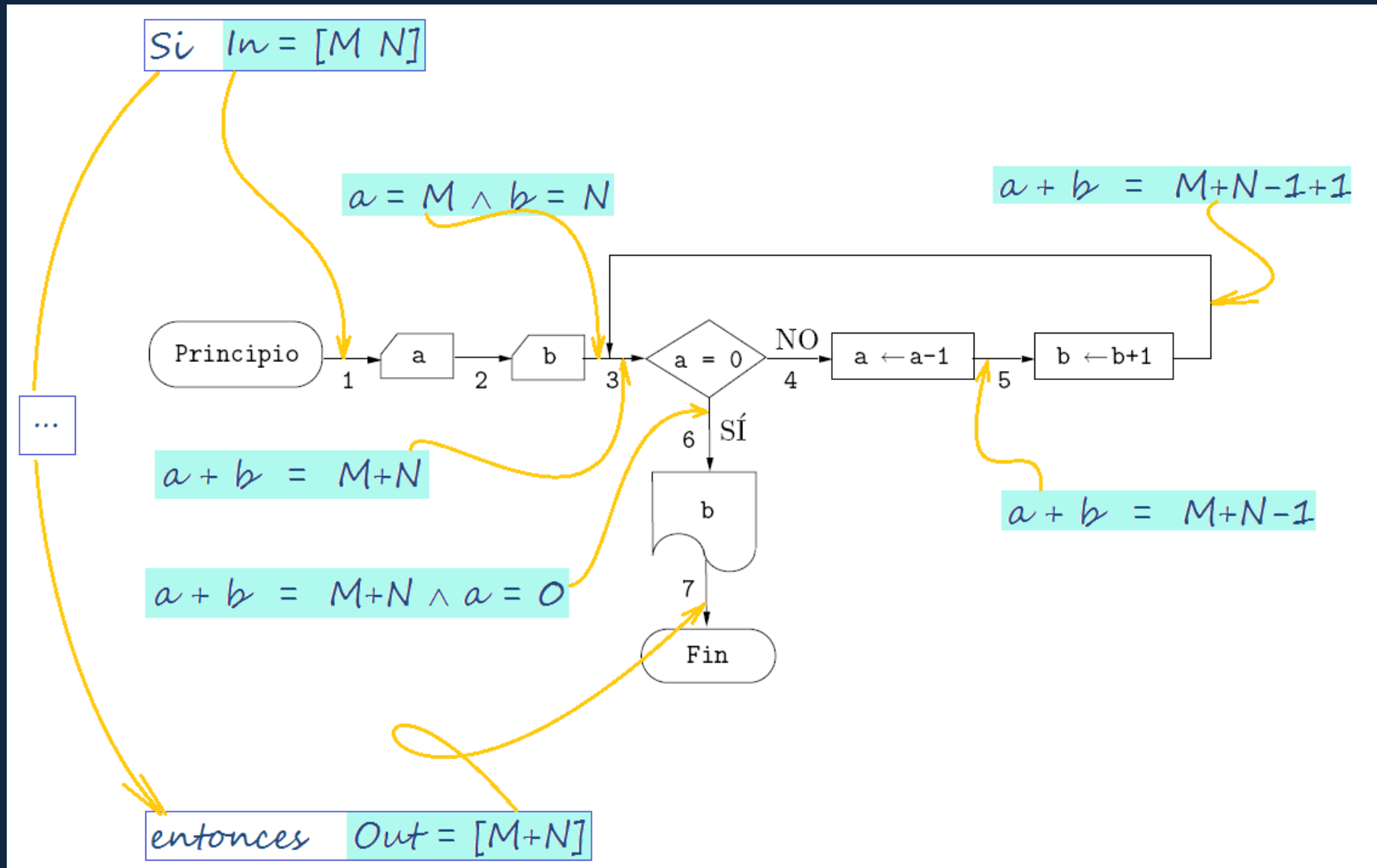
$$x = -3$$

(h.2) Elige la precondición

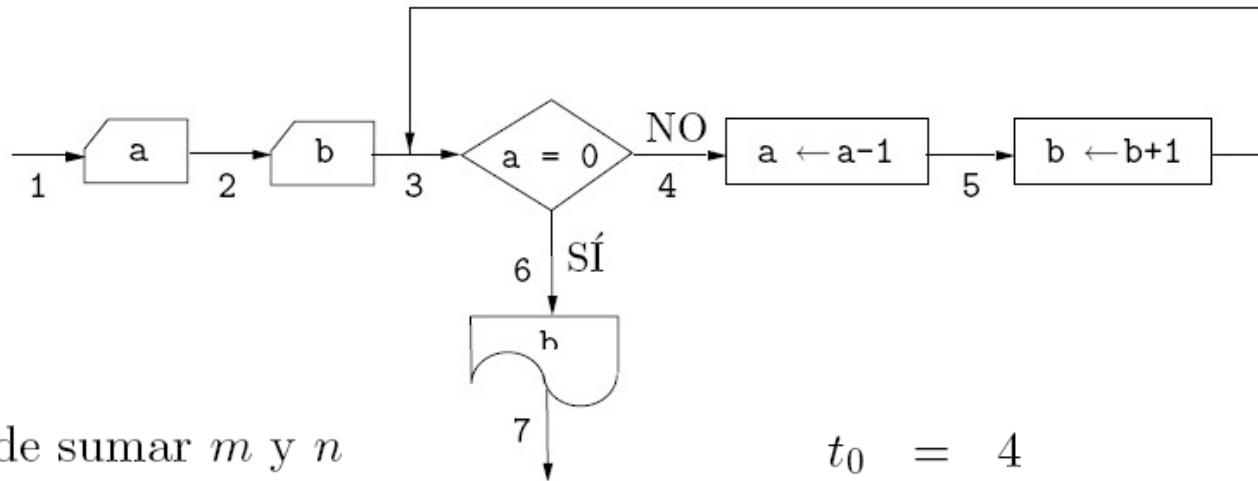
$$x = 3 \vee x = -3$$

más débil:

2.b Corrección



2.c Complejidad computacional = coste



t_m coste de sumar m y n

$$t_0 = 4 \text{ pasos}$$

$$t_m = t_{m-1} + 3, \text{ si } n \geq 1 \\ \text{independientemente de } n$$

$$t_0 = 4$$

$$t_1 = t_0 + 3 = 4 + 3 \cdot 1$$

$$t_2 = t_1 + 3 = 4 + 3 \cdot 2$$

$$\vdots \quad \vdots \quad \vdots$$

$$t_m = t_{m-1} + 3 = 4 + 3 \cdot m$$

$$3m + 4 \stackrel{m \rightarrow \infty}{\approx} 3m$$

$$3m \approx m$$

2.d Computabilidad: hay problemas no computables.

$$\text{STOP}(a, d) = \begin{cases} \text{Sí} & \text{si } a(d) \downarrow \\ \text{No} & \text{si } a(d) \uparrow \end{cases}$$

Ej.: El problema de parada es “no computable”

$$\text{STOP}'(p) = \text{STOP}(p, p) = \begin{cases} \text{Sí} & \text{si } p(p) \downarrow \\ \text{No} & \text{si } p(p) \uparrow \end{cases}$$

$$\text{RARO}(p) = \begin{cases} \uparrow & \text{si } \text{STOP}'(p) = \text{Sí} \\ \text{No} & \text{si } \text{STOP}'(p) = \text{No} \end{cases} = \begin{cases} \uparrow & \text{si } p(p) \downarrow \\ \text{No} & \text{si } p(p) \uparrow \end{cases}$$

$$\begin{aligned} \text{RARO}(\text{RARO}) &= \begin{cases} \uparrow & \text{si } \text{STOP}'(\text{RARO}) = \text{Sí} \\ \text{No} & \text{si } \text{STOP}'(\text{RARO}) = \text{No} \end{cases} \\ &= \begin{cases} \uparrow & \text{si } \text{RARO}(\text{RARO}) \downarrow \\ \text{No} & \text{si } \text{RARO}(\text{RARO}) \uparrow \end{cases} \end{aligned}$$

2.d Sucesión $3n+1$

```
#include <iostream>
using namespace std;

int main () {
    cout << "Dame el inicio de la secuencia (entero estrictamente positivo): ";
    int n;          // el dato inicial y subsiguientes en la secuencia "3n+1"
    cin >> n;
    int numPasos = 0; // contador de pasos
    while (n != 1) {
        cout << n << ", ";
        if (n%2 != 0) {
            n = 3*n + 1;
        } else {
            n = n / 2;
        }
        numPasos++;
    }
    cout << "1." << endl;
    cout << "Num. de pasos en total: " << numPasos << endl;
    system("pause");
    return 0;
}
```

2.d Sucesión 3n+1

```
/*-----  
Dame el inicio de la secuencia (entero estrictamente positivo): 3  
3, 10, 5, 16, 8, 4, 2, 1.  
Num. de pasos en total: 7  
-----  
Dame el inicio de la secuencia (entero estrictamente positivo): 6  
6, 3, 10, 5, 16, 8, 4, 2, 1.  
Num. de pasos en total: 8  
-----  
Dame el inicio de la secuencia (entero estrictamente positivo): 14  
14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.  
Num. de pasos en total: 17  
-----  
Dame el inicio de la secuencia (entero estrictamente positivo): 27  
27, 82, 41, 124, 62, 31, 94, 47, 142, 71, 214, 107, 322, 161, 484, 242, 121, 364  
, 182, 91, 274, 137, 412, 206, 103, 310, 155, 466, 233, 700, 350, 175, 526, 263,  
790, 395, 1186, 593, 1780, 890, 445, 1336, 668, 334, 167, 502, 251, 754, 377, 1  
132, 566, 283, 850, 425, 1276, 638, 319, 958, 479, 1438, 719, 2158, 1079, 3238,  
1619, 4858, 2429, 7288, 3644, 1822, 911, 2734, 1367, 4102, 2051, 6154, 3077, 923  
2, 4616, 2308, 1154, 577, 1732, 866, 433, 1300, 650, 325, 976, 488, 244, 122, 61  
, 184, 92, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1.  
Num. de pasos en total: 111  
-----  
Dame el inicio de la secuencia (entero estrictamente positivo): 1024  
1024, 512, 256, 128, 64, 32, 16, 8, 4, 2, 1.  
Num. de pasos en total: 10  
-----*/
```

3. Ejemplos de lenguajes algorítmicos y lenguajes de programación

Suma, de unidad en unidad:

```
Sean  $a, b \in \mathbb{Z}$ ;
Escribir "Datos: "
Leer a b
Escribir a b
Mientras  $a \neq 0$ , hacer
     $a - a-1$ ;
     $b - b+1$ ;
Escribir a b
```

```
suma :: Int -> Int -> Int
suma 0      b = b
suma (a+1) b = suma a (b+1)
```

Haskell
Prolog

```
suma (0, b, b).
suma (a+1, b, c+1) :- suma (a, b, c).
```

3. Ejs. de lenguajes de programación

```
program Suma (input, output);  
  
  {Suma, de unidad en unidad}  
  
  var  
    a, b: integer;  
  
begin  
  write('Datos: ');  
  readln(a, b);  
  writeln(a, ' ', b);  
  while a <> 0 do begin  
    a := a-1;  
    b := b+1  
  end;  
  writeln(a, ' ', b);  
  readln  
end.
```

Pascal

```
// Suma, de unidad en unidad  
  
#include <iostream>  
#include <stdlib.h>  
  
using namespace std;  
  
int main(int argc, char *argv[]) {  
  int a, b;  
  cout << "Datos: ";  
  cin >> a >> b;  
  cout << a << " " << b << endl;  
  while (a != 0) {  
    a--;  
    b++;  
  }  
  cout << a << " " << b << endl;  
  system("PAUSE");  
  return 0;  
}
```

C++

3. Ejs. de lenguajes de programación

```
/*-----  
Cometido: Suma dos enteros ...  
-----*/  
import java.io.*;  
import java.util.*;  
  
public class Sumar {  
    public static void main (String args []) throws Exception {  
        Scanner cin = new Scanner(System.in);  
        int a = cin.nextInt();  
        int b = cin.nextInt();  
        while (a != 0) {  
            a = a-1;  
            b = b+1;  
        }  
        System.out.println(b);  
    }  
}
```

Java

```
// Suma, de unidad en unidad  
  
#include <iostream>  
#include <stdlib.h>  
  
using namespace std;  
  
int main(int argc, char *argv[]) {  
    int a, b;  
    cout << "Datos: ";  
    cin >> a >> b;  
    cout << a << " " << b << endl;  
    while (a != 0) {  
        a--;  
        b++;  
    }  
    cout << a << " " << b << endl;  
    system("PAUSE");  
    return 0;  
}
```

C++

3. Ejs. de lenguajes de programación

```
#Suma, de unidad en unidad
#Lectura de datos:
[a, b] = [int(x) for x in raw_input("sumandos: ").split()]
#C'alculo, de unidad en unidad:
while a>0:
    a = a-1
    b = b+1
#Resultado:
print "La suma es %d" % b
```

Python

```
// Suma, de unidad en unidad
#include <iostream>
#include <stdlib.h>
using namespace std;
int main(int argc, char *argv[]) {
    int a, b;
    cout << "Datos: ";
    cin >> a >> b;
    cout << a << " " << b << endl;
    while (a != 0) {
        a--;
        b++;
    }
    cout << a << " " << b << endl;
    system("PAUSE");
    return 0;
}
```

C++