

Pseudocódigo y diagramas de flujo II

Fundamentos de Computación

**Diego Caro
José Fernandez
Fernanda Kri**

Terminología

Variables: entidad que almacena valores. Por ejemplo, números, vectores, etc.

1. Leer C1, C2, C3, C4

2. $P = (C1 + C2 + C3 + C4)/4$

3. Si $P < 4$:

4. Imprimir "Rechaza"

5. Si no: //aprobar es lo mejor!

6. Imprimir "Aprueba"

Expresión Booleana: Chequea el valor de verdad de alguna expresión que involucre alguna comparación

Comentario: Texto que no se ejecuta y sirve para que explicar pasos de un algoritmo

Condicionales:

Cambian el curso de ejecución de un programa de acuerdo al valor de una expresión booleana.

Operaciones de comparación

- Permiten verificar si variables cumplen algunas reglas básicas.
- Devuelven un valor booleano (Verdadero, Falso)

$<$, $>$, \leq , \geq : mayor, y mayor igual
 $=$, \neq : igual, distinto

Sintáxis	Operador	Ejemplo	Resultado
$a < b$	menor que	$2 < 5$	
$a \leq b$	menor o igual	$2 \leq 2$	
$a \geq b$	mayor o igual	$2 \geq 32$	
$a > b$	mayor	$0 > -1$	
$a == b$	igual	$1 == -1$	
$a \neq b$	distinto	$1 \neq -1$	

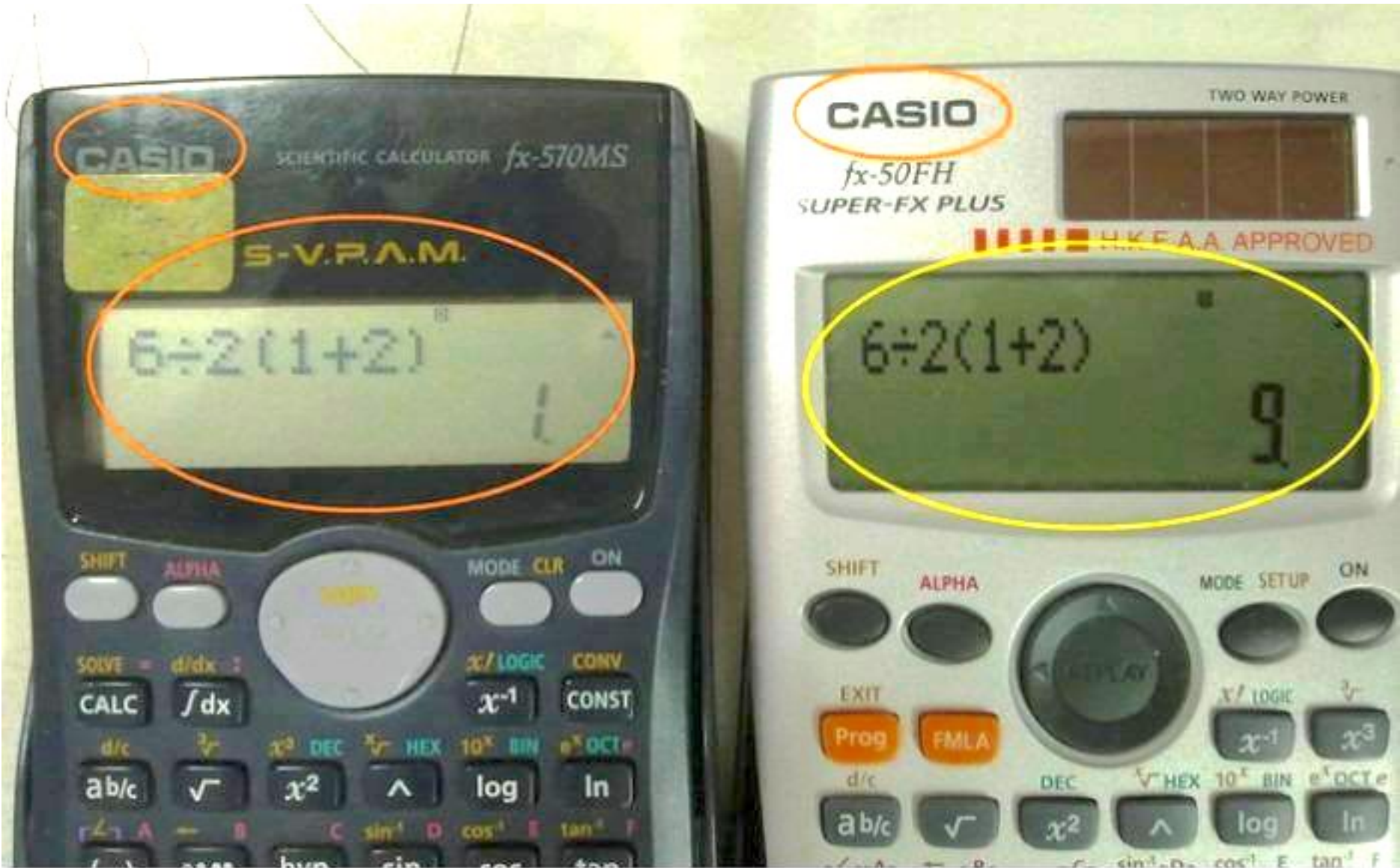
Precedencia de operadores aritméticos

- Orden en el que se evalúa una expresión.
 - Si... igual que en álgebra
 - Por prioridad
 - Si dos prioridades son iguales, se evalúa de izquierda a derecha

Prioridad	1	1	1	1	2	2
Operador	()	*	/	%	+	-

División Resto ó módulo de la división

Expresión	Resultado
$2+3*7$	
$6-2*4$	
$(6-2)*4$	
$4*5\%3$	
$(12*(-1))$	
$(10+1)\%9$	
$6/2(1+2)$	



Esto se conoce como BUG (o error de programación)

¿Estas dos expresiones significan lo mismo?

$z = 2$

Asignación: asignar el valor 2
a la variable z

$z = 2$

$z == 2$

Comparación: ¿el contenido
de la variable z es igual a 2?

$z == 2$

Traza: siguiendo variables

- Traza: tabla con el seguimiento de variables y operaciones de comparación para cada instrucción de pseudocódigo.
- Para qué: asegurar que tu pseudocódigo haga lo que debe.
- Ejemplo: intercambiar el contenido de dos variables.

Variables/Comparaciones			
	a	b	t
1. a = 2	2	-	-
2. b = 9	2	9	-
3. t = a	2	9	2
4. a = b	9	9	2
5. b = t	9	2	2

1. $n = 1$
2. **Mientras** $n \leq 10$:
3. $t = 2n$
4. Imprimir t
5. $n = n + 1$

	n	$n \leq 10$	t
$n = 1$	1	-	-
Mientras $n \leq 10$	1	V	-
$t = 2n$	1	V	2
$n = n+1$	2	V	2
Mientras $n \leq 10$	2	V	2
$t = 2n$	2	V	4
$n = n+1$	3	V	4
Mientras $n \leq 10$	3	V	4
$t = 2n$	3	V	6
$n = n+1$	4	V	6
Mientras $n \leq 10$	4	V	6
$t = 2n$	4	V	8
$n = n+1$	5	V	8
Mientras $n \leq 10$	5	V	8
$t = 2n$	5	V	10
$n = n+1$	6	V	10
...
$n = n + 1$	10	V	18
Mientras $n \leq 10$	10	V	18
$t = 2n$	10	V	10
$n = n + 1$	11	V	10
Mientras $n \leq 10$	11	F	10

Expresiones booleanas

- En los condicionales también podemos agrupar más expresiones booleanas.
- Podemos usar and, or y not.
- Ejemplo: máximo de tres números
 1. Leer x, y, z
 2. **Si** $x == y \wedge x == z$:
 3. Imprimir “x, y, z son iguales”
 4. **Si** $x \geq y \wedge y > z$: // si x e y son iguales, asumiré que x es el mayor.
 5. Imprimir “x mayor”
 6. **Si** $y \geq x \wedge y > z$:
 7. Imprimir “y mayor”
 8. **Si** $z \geq x \wedge z > y$:
 9. Imprimir “z mayor”

a	$\neg a$	a	b	$a \wedge b$	$a \vee b$
true	false	false	false	false	false
false	true	false	true	false	true
		true	false	false	true
		true	true	true	true

Ejercicio para ayudantía, hacer la traza para:

1. $x = 8, y=0, z=2$

2. $x = 8, y=8, z=8$

3. $x=8, y=1, z=9$

Ciclo Para

- Ciclo **Para** (o for en inglés) se utiliza para recorrer una secuencia de números enteros.
- Se indica el inicio y el fin (incluido), y se asume que el siguiente elemento es el sucesor.
 - **Notación: Para** $i=1$ hasta n : genera la secuencia 1, 2, 3, ..., n (incluido n)
- Es equivalente al ciclo **Mientras** (o while en inglés) cuando tenemos un contador.

1. **Para** $i=1$ a 10:

2. $t = 2i$

3. Imprimir t

Son Equivalentes

1. $i = 1$

2. **Mientras** $i \leq 10$:

3. $t = 2i$

4. Imprimir t

5. $i = i + 1$

1. **Para** $i=1$ hasta 5:

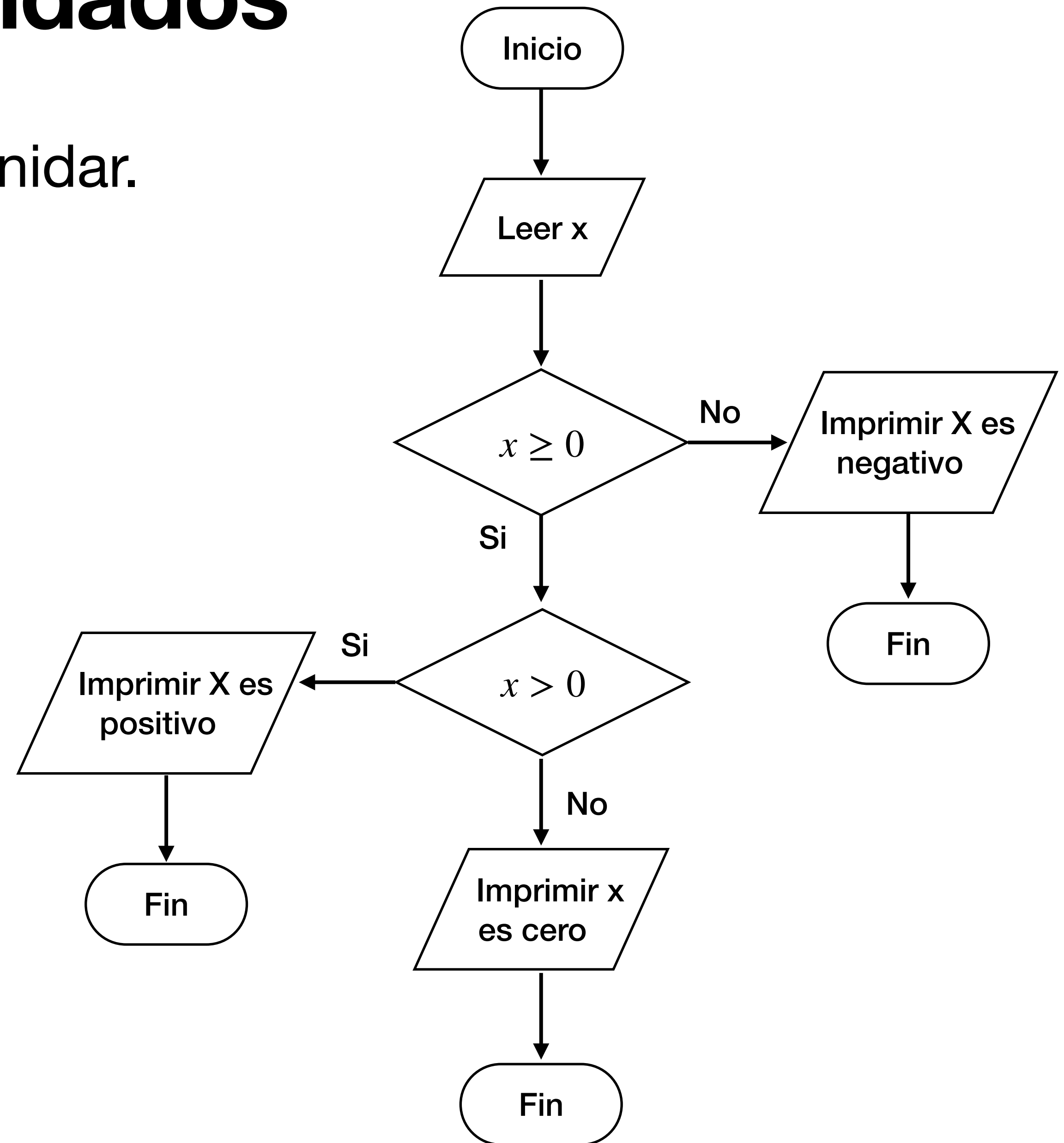
2. Imprimir $i*i$

i	$i*i$
1	1
2	4
3	9
4	16
5	25

Condicionales y ciclos anidados

- Los condicionales y ciclos se pueden anidar.
- ¡Una condición/ciclo dentro de otra!

1. Leer x
2. **Si** $x \geq 0$:
3. **Si** $x > 0$:
4. Imprimir “x es positivo”
5. **Si no:**
6. Imprimir “x es cero”
7. **Si no:**
8. Imprimir “x es negativo”



¿Qué hace este pseudocódigo?

1. **Para** $i=1$ hasta 12:
2. **Para** $j = 1$ hasta 12:
3. Imprimir $i, j, i*j$

Ejercicio para ayudantía, traducir este pseudocódigo a diagrama de flujo

Funciones

- Funciones: bloque de pseudocódigo que se puede llamar cuantas veces sea necesario. Permite encapsular algoritmos. El objetivo es reducir pseudocódigo cuando una acción se debe ejecutar varias veces.
- Tienen un nombre, una entrada y una salida.
- ¡Debes describir **claramente** qué es la entrada y qué es la salida!

Entrada

Nombre Promedio(X):
// Entrada: X es una lista de números X_1, X_2, \dots, X_N
// Salida: promedio de los números en la lista X
1. Sea $p = 0$ una variable en los reales
2. **Para** $i = 1$ **hasta** N :
3. $p = p + X_i$
4. **retornar** p/N
Salida

1. Sea X una lista $X_1, X_2, \dots, X_N, X_i \in \mathbb{R}$
2. Leer N números y dejarlos en la lista X
3. $z = \underline{\text{Promedio}(X)}$ Nombre de función
que queremos usar
4. Imprimir “El promedio es”, z

¿Cuán detallado debo ser?

Lo suficiente para que se comprenda de manera no ambigua lo que se quiere computar.

¿Hay alguna diferencia con Python?

Promedio(X):

// Entrada: X es una lista de números X_1, X_2, \dots, X_N

// Salida: promedio de los números en la lista X

1. Sea $p = 0$ una variable en los reales
2. **Para** $i = 1$ **hasta** N :
3. $p = p + X_i$
4. **retornar** p/N

1. Sea X una lista $X_1, X_2, \dots, X_N, X_i \in \mathbb{R}$
2. Leer N números y dejarlos en la lista X
3. $z = \text{Promedio}(X)$
4. Imprimir “El promedio es”, z

```
1 def promedio(X):
2     N = len(X)
3     p = 0
4     for v in X:
5         p += v
6     return p/N
7
8 X = []
9 N = int(input('Ingrese tamaño lista: '))
10 for i in range(N):
11     y = int(input('Ingrese elemento: '))
12     X.append(y)
13 z = promedio(X)
14 print('El promedio es', z)
```

¿Hay alguna diferencia con Python?

Promedio(X):

// Entrada: X es una lista de números X_1, X_2, \dots, X_N

// Salida: promedio de los números en la lista X

1. $p = \sum_{i=1}^N X_i$
2. **retornar** p/N

1. Sea X una lista $X_1, X_2, \dots, X_N, X_i \in \mathbb{R}$
2. Leer N números y dejarlos en la lista X
3. $z = \text{Promedio}(X)$
4. Imprimir “El promedio es”, z

```
1 def promedio(X):
2     N = len(X)
3     p = sum(X)
4     return p/N
5
6 X = []
7 N = int(input('Ingrese tamaño lista: '))
8 for i in range(N):
9     y = int(input('Ingrese elemento: '))
10    X.append(y)
11 z = promedio(X)
12 print('El promedio es', z)
```


¿Hay alguna diferencia con C?

Promedio(X):

// Entrada: X es una lista de números X_1, X_2, \dots, X_N

// Salida: promedio de los números en la lista X

1. $p = \sum_{i=1}^N X_i$
2. **retornar** p/N

1. Sea X una lista $X_1, X_2, \dots, X_N, X_i \in \mathbb{R}$
2. Leer N números y dejarlos en la lista X
3. $z = \mathbf{Promedio}(X)$
4. Imprimir “El promedio es”, z

```
1 #include <stdio.h>
2
3 float promedio(float *A, int N) {
4     int i;
5     float p = 0;
6     for (i = 0; i < N; i++) {
7         p += A[i];
8     }
9     return p/N;
10 }
11
12 int main() {
13     float X[100];
14     int N = 100;
15     int i;
16     float z;
17
18     for (i = 0; i < N; i++) {
19         scanf("%f", &X[i]);
20     }
21
22     z = promedio(X, N);
23     printf("El promedio es %f\n", z);
24     return 0;
25 }
```

Actividad: calidad del transporte público

Entre menor es la variación de la duración de tu tiempo de viaje, **menos incertidumbre** tendrás sobre su duración.

Podrías planificar leer o estudiar Fundamentos de Computación :B

Día	Tiempo de viaje en minutos
1	67
2	45
3	84
s	19.553

Día	Tiempo de viaje en minutos
1	70
2	70
3	70
s	0

Actividad: Escribe el pseudocódigo y el diagrama de flujo para calcular la desviación estandar

Desviación Estándar

$$s = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}$$

↑
Promedio
(reusa la función promedio de la slide 12)

Bibliografía

- Sedgewick, R., & Wayne, K. (2016). Computer science: An interdisciplinary approach. Addison-Wesley Professional. Chapter 1.3. Available at <https://introcs.cs.princeton.edu/java/13flow/>