Pseudocódigo y diagramas de flujo II

Fundamentos de Computación

Diego Caro

2020-2

https://github.com/diegocaro/

Terminología

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

```
1. read C1, C2, C3, C4
2. P = (C1 + C2 + C3 + C4)/4
3. if P < 4:
4.     print “Rechaza”
5. else: //aprobar es lo mejor!
6.     print “Aprueba"
```

Condicionales:

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

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

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	V
a ⩽ b	menor o igual	2 ⩽ 2	V
a ⩾ b	mayor o igual	2 ⩾ 32	F
a > b	mayor	0 > -1	V
a == b	igual	1 == -1	F
a ≠ b	distinto	1 ≠ -1	V

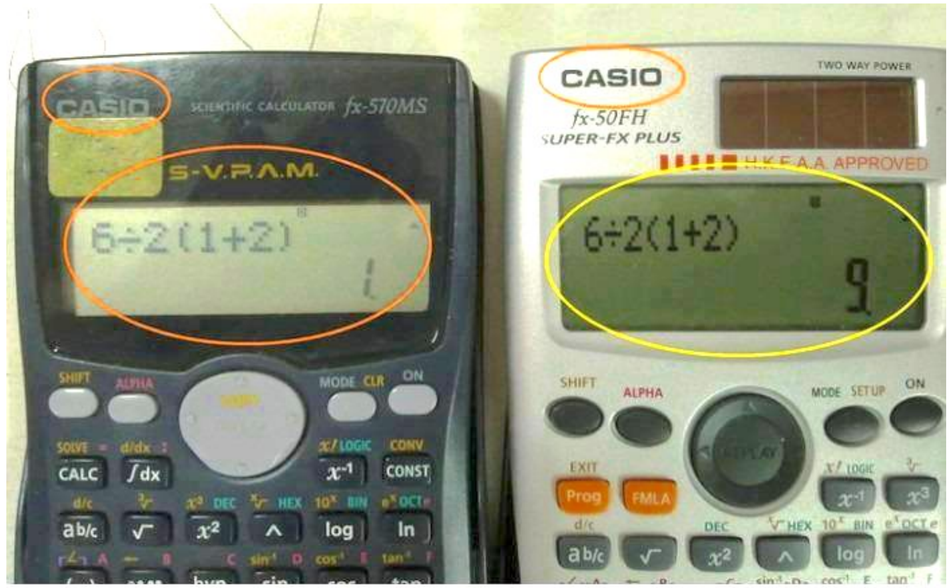
Precedencia de operadores aritméticos

- 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	2	2	2	3	3
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)	

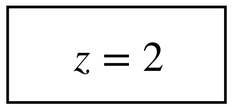


Esto se conoce como BUG (o error de programación)

¿Estas dos expresiones significan lo mismo?

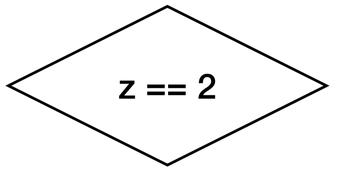
$z = 2$

Asignación: asignar el valor 2 a la variable z



$z == 2$

Comparación: ¿el contenido de la variable z es igual a 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. while n ≤ 10:
3.     t = 2n
4.     print t
5.     n = n + 1
```

	n	n ≤ 10	t
n = 1	1	V	-
Mientras n ≤ 10	1	V	-
t = 2n	1	V	2
n = n+1	2	V	2
Mientras n ≤ 10	2	V	2
t = 2n	2	V	4
n = n+1	3	V	4
Mientras n ≤ 10	3	V	4
t = 2n	3	V	6
n = n+1	4	V	6
Mientras n ≤ 10	4	V	6
t = 2n	4	V	8
n = n+1	5	V	8
Mientras n ≤ 10	5	V	8
t = 2n	5	V	10
n = n+1	6	V	10
...	...	...	...
n = n + 1	10	V	18
Mientras n ≤ 10	10	V	18
t = 2n	10	V	20
n = n + 1	11	F	20
Mientras n ≤ 10	11	F	20

Expresiones booleanas

- En los condicionales también podemos agrupar más expresiones booleanas.
- Podemos usar and, or y not.
- Ejemplo: máximo de tres números

a	$\neg a$
true	false
false	true

a	b	$a \wedge b$	$a \vee b$
false	false	false	false
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

1. **read** x, y, z
2. **if** $x == y \wedge x == z$:
3. **print** “x, y, z son iguales"
4. **if** $x \geq y \wedge y > z$: // si x e y son iguales, asumiré que x es el mayor.
5. **print** “x mayor”
6. **if** $y \geq x \wedge y > z$:
7. **print** “y mayor”
8. **if** $z \geq x \wedge z > y$:
9. **print** “z mayor”

Ciclo For

- Ciclo **for**(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**: **for** i=1 **to** n: genera la secuencia 1, 2, 3, …, n (incluido n)
- Es equivalente al ciclo **while** cuando tenemos un contador.
- Atención: no tiene traducción directa a diagrama de flujo (

1. **for** i=1 **to** 10:
2. t = 2i
3. **print** t

Son Equivalentes

1. i = 1
2. **while** $i \leq 10$:
3. t = 2i
4. **print** t
5. i = i + 1

1. **for** i=1 **to** 5:
2. **print** i*i

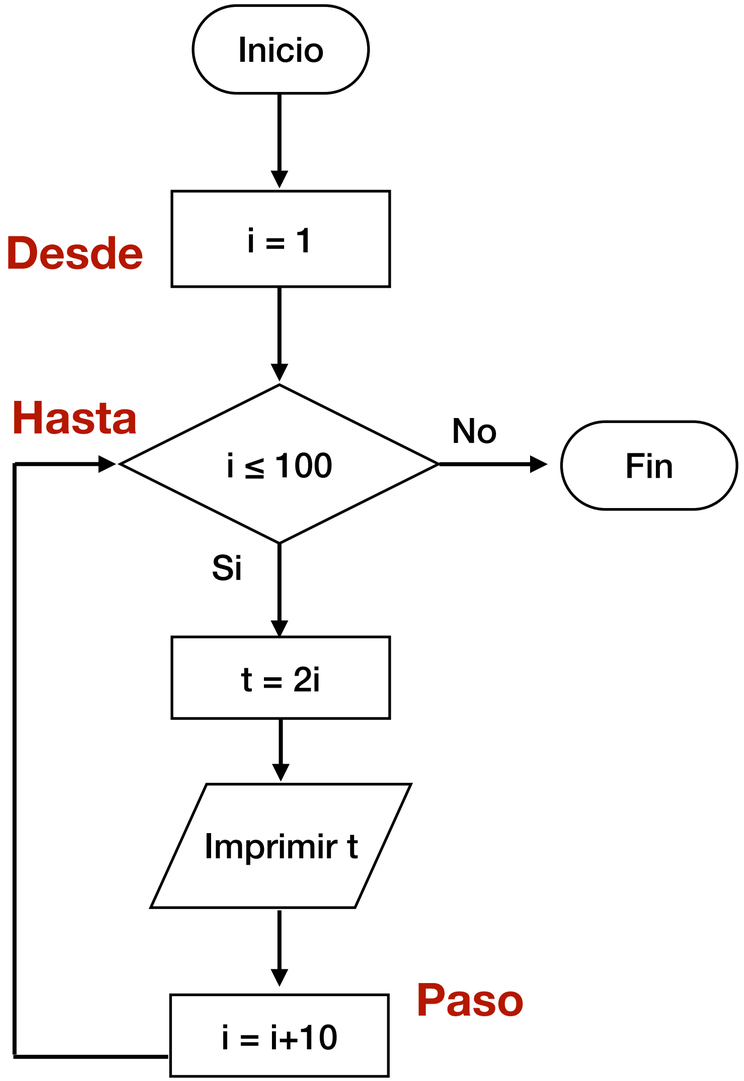
i	i*i
1	1
2	4
3	9
4	16
5	25

Diagrama de Flujo ciclo Para

Desde Hasta Paso

1. **for** i=1 **to** 100 **step** 10:
2. t = 2i
3. Imprimir t

¿Qué imprime este código?



1. **for** i=0 **to** 4:
2. k = 2i + 1
3. t = 2k
4. **print** t

i	k	t
0	1	2
1	3	6
2	5	10
3	7	14
4	9	18

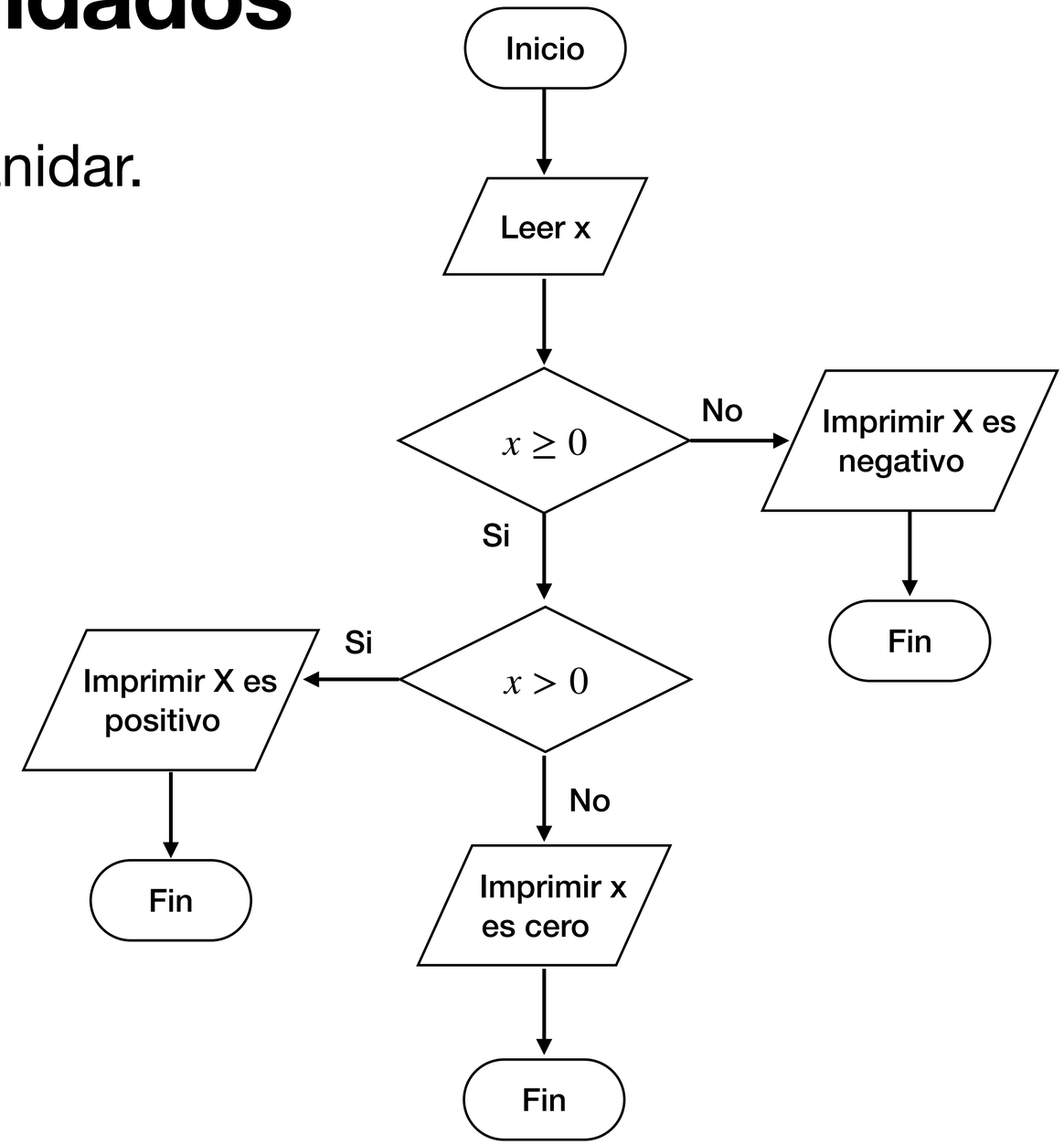
1. i = 1
2. **while** $i \leq 10$:
3. t = 2i
4. **print** t
5. i = i + 2

i	t
1	2
3	6
5	10
7	14
9	18
11	18

Condicionales y ciclos anidados

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

1. **read** x
2. **if** $x \geq 0$:
3. **if** $x > 0$:
4. **print** “x es positivo”
5. **else**:
6. **print** “x es cero"
7. **if:**
8. **print** “x es negativo"



¿Qué hace este pseudocódigo?

```
1. for i=1 to 12:
2.     for j = 1 to 12:
3.         print i, j, i*j
```

i	j	i*j
1	-	-
1	1	1
1	2	2
1	3	3
1	..	..
1	12	12
2	1	2

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: X es una lista de números $X_1, X_2, \ldots, X_N$
// Salida: promedio de los números en la lista X

Entrada

Nombre

1. **def** Promedio(X):
2. p = 0
3. **for** i =1 **to** N:
4. p = p + X_i
5. **return** p/N

Salida

1. **read** $X_1, X_2, \ldots, X_N, X_i \in \mathbb{R}$
2. X = list($X_1, X_2, \ldots, X_N$)
3. z = Promedio(X)
4. **print** “El promedio es”, z

Nombre de función que queremos usar

// Entrada: X es una lista de números $X_1, X_2, \ldots, X_N$
// Salida: promedio de los números en la lista X

1. **def** Promedio(X):
2. p = 0
3. **for** i =1 **to** N:
4. p = p + X_i
5. **return** p/N

X	i	Xi	p	N
9,8,7,6	-	-	0	4
9,8,7,6	1	9	9	4
9,8,7,6	2	8	17	4
9,8,7,6	3	7	24	4
9,8,7,6	4	6	30	4

p/N = 30/4 = 7.5

1. **read** $X_1, X_2, \ldots, X_N, X_i \in \mathbb{R}$
2. X = list($X_1, X_2, \ldots, X_N$)
3. z = **Promedio**(X)
4. **print** “El promedio es”, z

X	z
9,8,7,6	7,5

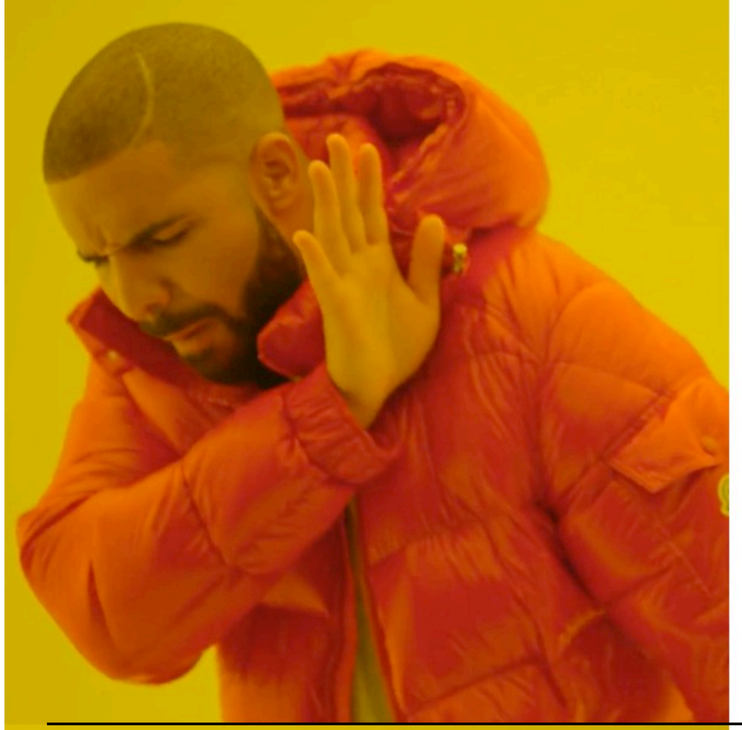
Funciones disponibles

- Asumiremos que todas las funciones matemáticas están disponibles:
  - Potencias, Raíces, Exponenciales, etc…
  - Trigonométricas: cos, sin, tan, + funciones inversas
  - (a menos que se le pida implementar alguna con operaciones más básicas)

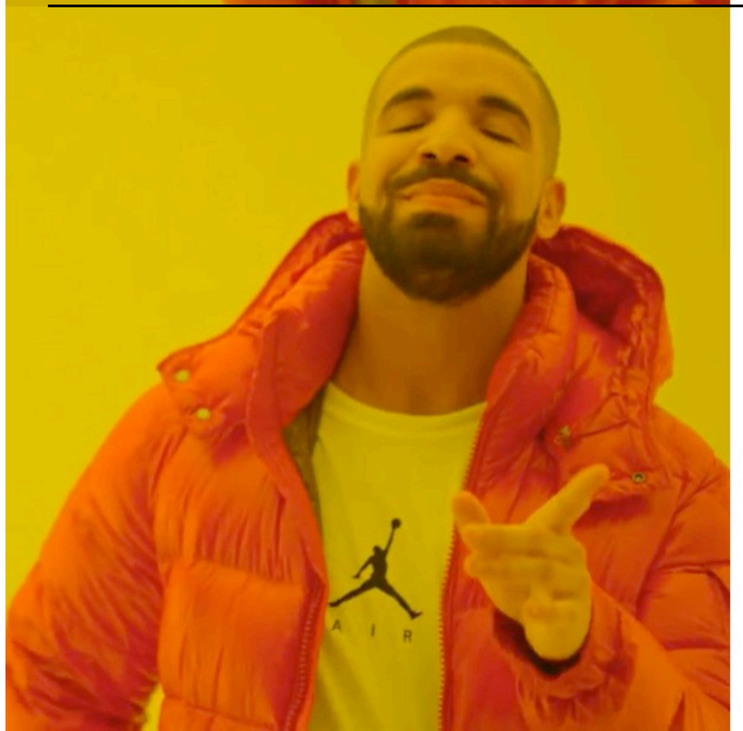
¿Cuán detallado debo ser?

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

2. Debes usar expresiones matemáticas con un solo nivel de anidación



$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \frac{1}{n}\sum_{j=0}x_j)^2}$$



$$\bar{x} = \frac{1}{n}\sum_{j=0}x_j$$

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

¿Hay alguna diferencia con Python?

// Entrada: X es una lista de números $X_1, X_2, \ldots, X_N$
// Salida: promedio de los números en la lista X

1. **def** Promedio(X):
2. p = 0
3. **for** i =1 **to** N:
4. p = p + X_i
5. **return** p/N
6.
7. **read** $X_1, X_2, \ldots, X_N, X_i \in \mathbb{R}$
8. X = list($X_1, X_2, \ldots, X_N$)
9. z = **Promedio**(X)
10. **print** “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?

// Entrada: X es una lista de números $X_1, X_2, \ldots, X_N$
// Salida: promedio de los números en la lista X

1. def promedio(X)
2. $p = \sum_{i=1}^{N} X_i$
3. **return** p/N

5. **read** $X_1, X_2, \ldots, X_N, X_i \in \mathbb{R}$
6. X = list($X_1, X_2, \ldots, X_N$)
7. z = **Promedio**(X)
8. **print** “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?

// Entrada: X es una lista de números $X_1, X_2, \ldots, X_N$
// Salida: promedio de los números en la lista X

1. def promedio(X)
2. $p = \sum_{i=1}^{N} X_i$
3. **return** p/N

5. **read** $X_1, X_2, \ldots, X_N, X_i \in \mathbb{R}$
6. X = list($X_1, X_2, \ldots, X_N$)
7. z = **Promedio**(X)
8. **print** “El promedio es”, z

```c
#include <stdio.h>

float promedio(float *A, int N) {
  int i;
  float p = 0;
  for (i = 0; i < N; i++) {
      p += A[i];
  }
  return p/N;
}

int main() {
  float X[100];
  int N = 100;
  int i;
  float z;

  for (i = 0; i < N; i++) {
      scanf("%f", &X[i]);
  }

  z = promedio(X, N);
  printf("El promedio es %f\n", z);
  return 0;
}
```

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: En equipos de 4-5 personas escribe el pseudocódigo y el diagrama de flujo para calcular la desviación estándar.

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)

x= [67, 45, 84]

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/