

# FUNDAMENTOS DE PROGRAMACIÓN

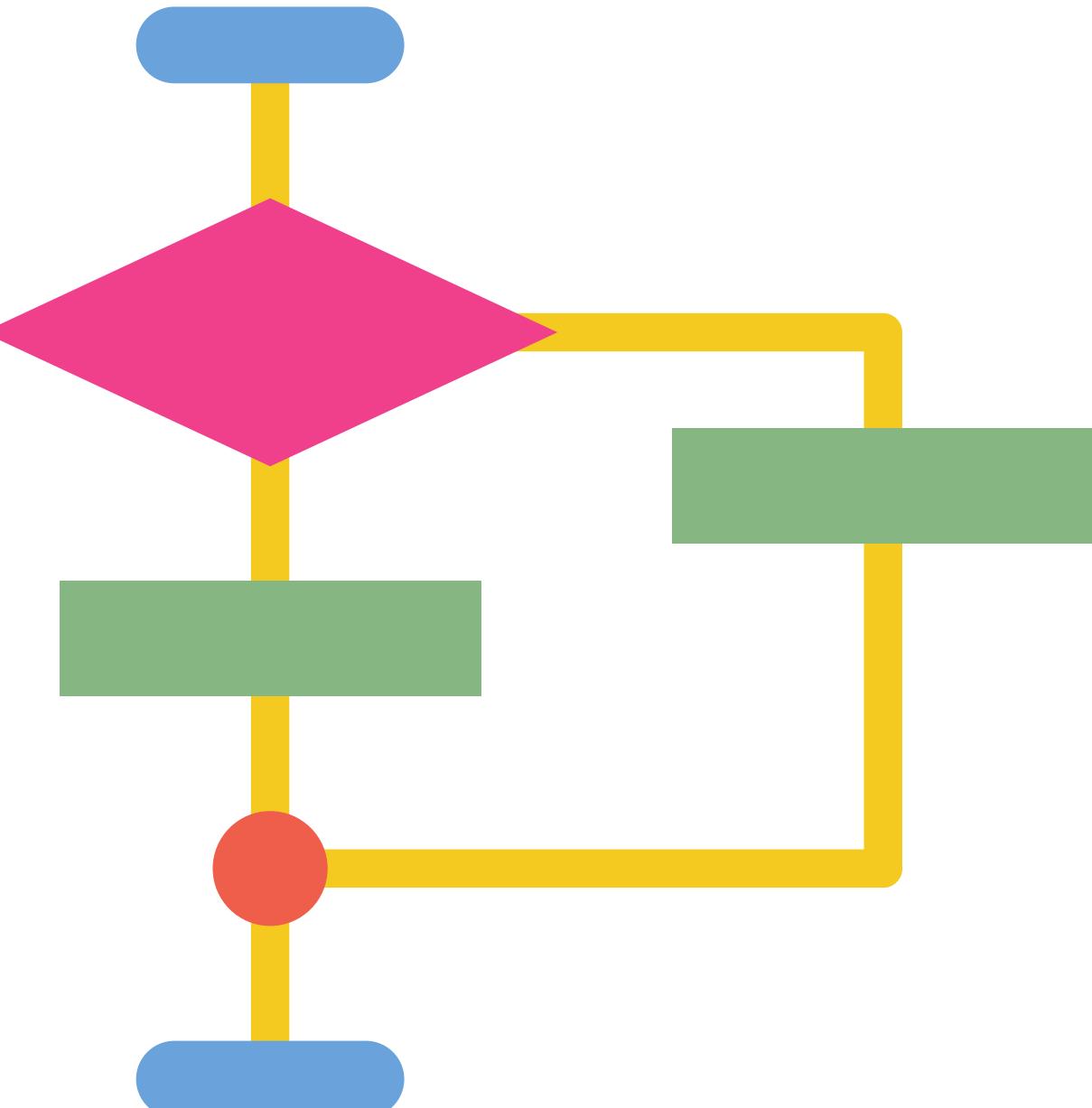


# FUNDAMENTOS DE PROGRAMACIÓN



## ¿QUÉ VAMOS A VER?

- Diagramación de algoritmos
- Representación
- Diagramas de flujo para algoritmos
- Arreglos unidimensionales
- Dimensionamiento de un arreglo
- Arreglos en base cero





# DIAGRAMACIÓN DE ALGORITMOS

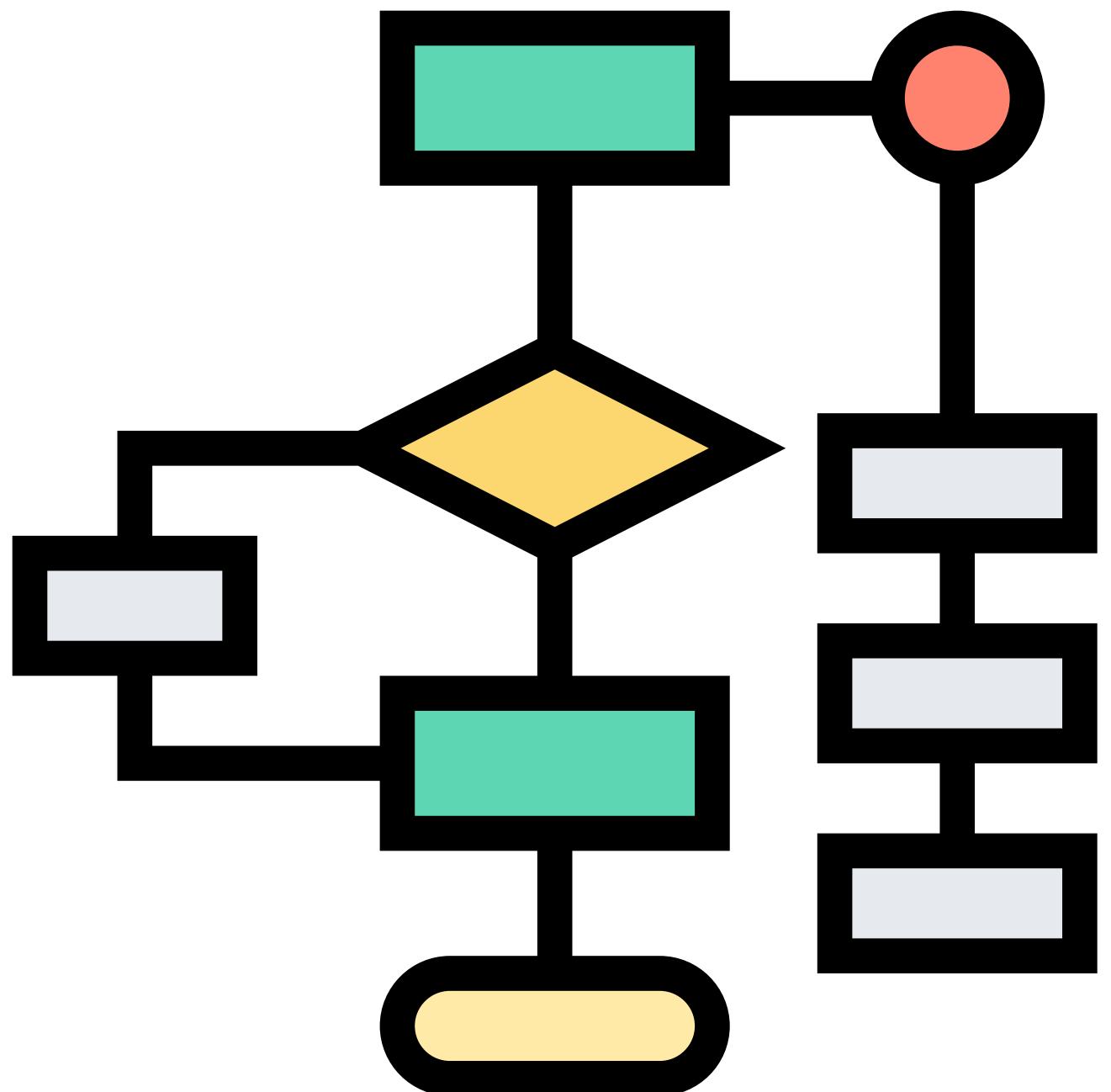
- Un diagrama de flujo es una de las técnicas de representación de algoritmos más conocida y utilizada.
- Puede describir un proceso, un sistema o un algoritmo informático. Se usan para documentar, estudiar, planificar, mejorar y comunicar procesos complejos en diagramas fáciles de comprender.
- Emplean rectángulos, óvalos, diamantes y otras numerosas figuras para definir el tipo de paso, junto con flechas conectoras que establecen el flujo y la secuencia.





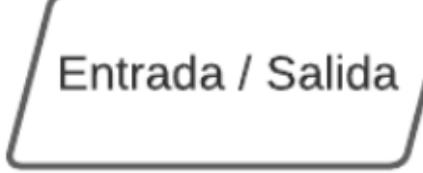
# REPRESENTACIÓN

- Cada símbolo del diagrama de flujo representa una acción en específico.
- Es importante recordar que cada diagrama de flujo se ejecuta desde arriba hacia abajo, siguiendo cada uno de los pasos e instrucciones.
- Otro elemento importante son las líneas de flujo que van a representar la secuencia lógica del algoritmo.



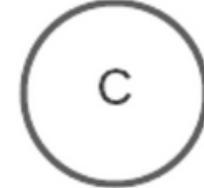
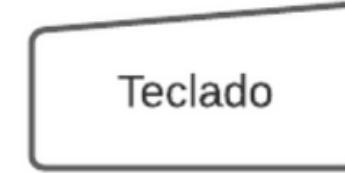


# REPRESENTACIÓN

Símbolo	Función	Representación
Terminal	Representa el inicio y fin de un algoritmo.	
Entrada / Salida	Cualquier dato de entrada o salida.	
Proceso	Cualquier tipo de operación que pueda originar un cambio de valores en las variables, formato o posición de la información almacenada en memoria, operaciones aritméticas, etc.	

# REPRESENTACIÓN



Símbolo	Función	Representación
Decisión	Indica operaciones lógicas o de comparación entre datos.	 Decisión
Conector	Sirve para enlazar dos partes cualquiera de un algoritmo.	 C
Teclado	Se utiliza en ocasiones para las entradas de datos por teclado.	 Teclado
Pantalla	Se utiliza en ocasiones para las salidas en pantalla.	 Pantalla



# DIAGRAMAS DE FLUJO PARA ALGORITMOS

---

- Son útiles para escribir un programa o algoritmo y explicárselo a otros.
- Se pueden usar para explicar detalladamente la lógica detrás de un programa antes de empezar a codificar el proceso automatizado.
- Específicamente, los diagramas de flujo pueden:
  - Demostrar cómo el código está organizado.
  - Visualizar la ejecución de un código dentro de un programa.
  - Mostrar la estructura de un sitio web o aplicación.
  - Comprender cómo los usuarios navegan por una aplicación.

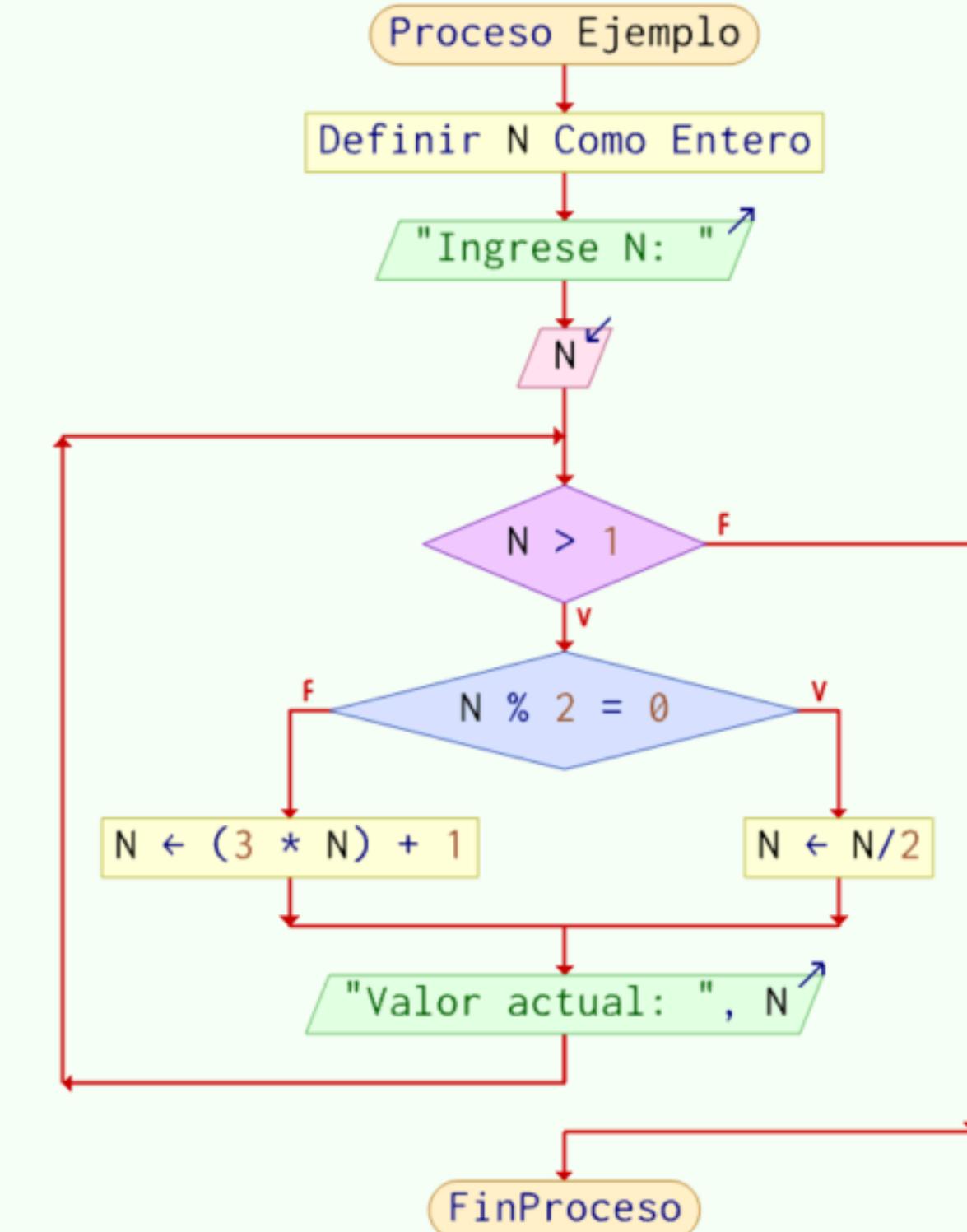




# DIAGRAMAS DE FLUJO PARA ALGORITMOS

## EJEMPLO

- De acuerdo a la conjetura de Ulam, dado número natural N puede llegar a convertirse en un valor igual a 1 a través de la siguiente secuencia:
  - Si N es par, se divide por dos
  - Si N es impar, se multiplica por tres y se suma 1.
- Desarrolle un algoritmo expresado en diagrama de flujo que resuelva este problema.

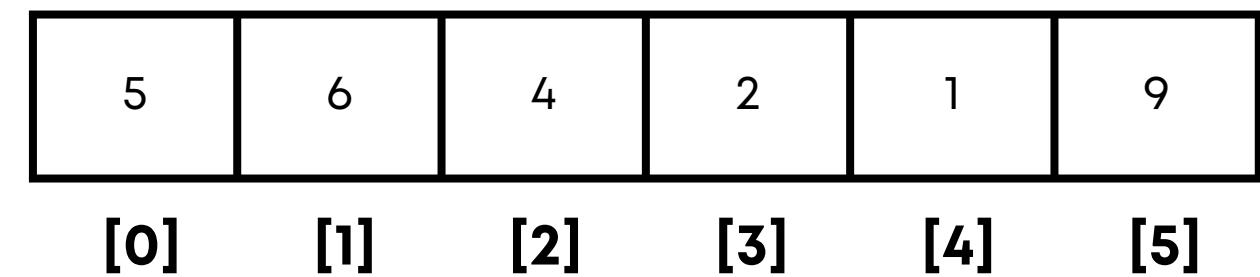




# ARREGLOS UNIDIMENSIONALES

- Los arreglos son usados para procesar una colección de datos del mismo tipo.
- Nos permiten almacenar un determinado número de datos bajo un mismo identificador, para luego referirse a ellos utilizando uno o más subíndices.
- Para poder utilizar un arreglo es obligatorio conocer su dimensión, lo cual va a determinar cuántos elementos podemos almacenar y como accederemos a ellos.
- Cada ítem del arreglo se denomina elemento, y se enumeran consecutivamente 0, 1, 2, 3... Estos números se denominan valores índice o subíndice del arreglo.

```
1 Proceso Ejemplo
2
3     Dimension arreglo[6];
4
5 FinProceso
```





# DIMENSIONAMIENTO DE UN ARREGLO

- La instrucción **Dimension** permite declarar un arreglo, indicando sus dimensiones.
  - **Dimension:** Es la palabra reservada del lenguaje que se usa para declarar un arreglo.
  - **nombre\_arreglo:** Corresponde al identificador por medio del cual podemos referenciar (nombrar) nuestro arreglo.
  - **tamaño:** Debe ser un número entero mayor que 0. Va a indicar el tamaño del arreglo o bien, el número de elementos que podremos almacenar dentro.

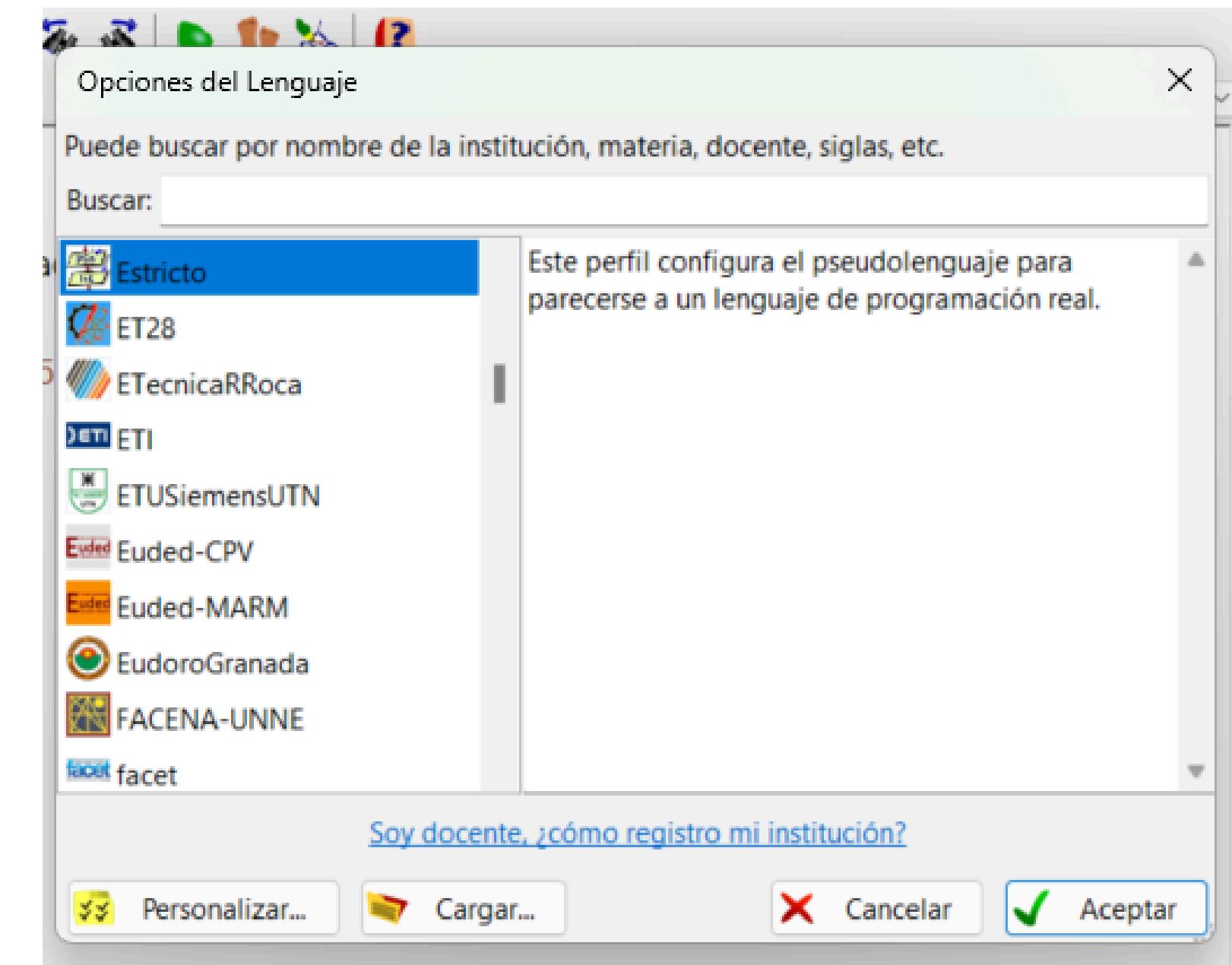
Dimension nombre\_arreglo[tamaño]

```
1  Proceso Ejemplo
2
3      Definir arregloDatos, edad Como Entero;
4
5      Dimension arregloDatos[25];
6      Dimension edad[50];
7
8  FinProceso
```



# ARREGLOS EN BASE CERO

- En la mayoría de los lenguajes de programación el índice inicial del arreglo siempre es cero.
- En PSeInt en ocasiones esto no es así por defecto, pero se puede configurar.
- Acceder al menú **Configurar -> Opciones del lenguaje (perfiles)**.
- Al seleccionar la opción “Estricto”, la sintaxis solicitada por PSeInt será más parecida a un lenguaje de programación regular que a un pseudocódigo. Esto ayudará a la transición al lenguaje de programación que tendremos en corto plazo.





# INICIALIZACIÓN DE UN ARRAY

- Se deben asignar valores a los elementos del arreglo antes de ser utilizados, tal como se asignan valor a una variable.
- En el ejemplo, la primera sentencia fija edades[0] al valor 15, edades[1] al valor 25, etc.
- Este método no es práctico cuando el array contiene muchos elementos.

```
1  Proceso Ejemplo
2
3      Definir edades Como Entero;
4
5      Dimension edades[5];
6
7      edades[0] ← 15;
8      edades[1] ← 25;
9      edades[2] ← 30;
10     edades[3] ← 34;
11     edades[4] ← 40;
12
13 FinProceso
```



# ARREGLOS UNIDIMENSIONALES

## EJEMPLO

- Realice un algoritmo para inicializar un arreglo de 10 elementos a 0.

```
1  Proceso Ejemplo
2
3      Definir datos, i Como Entero;
4
5      Dimension datos[10];
6      Escribir Sin Saltar "|";
7
8      Para i<0 Hasta 9 Con Paso 1 Hacer
9          | datos[i] ← 0;
10         | Escribir Sin Saltar " ", datos[i], " |";
11     FinPara
12     Escribir " ";
13
14 FinProceso
15 |
```



# ALMACENAR O EXTRAER EN UN ARREGLO

- **Almacenamiento:** Permite guardar un valor en la posición que se indique por medio del índice.

- **Caso 1:** Tenemos un arreglo llamado misDatos con tamaño de tres elementos, en el queremos almacenar el valor 10 en la posición 1.

- **Extracción:** Permite acceder al valor que se encuentra en la posición indicada por medio del índice. Lo podemos almacenar dentro de una variable.

- **Caso 2:** Recuperar el valor desde una posición del arreglo.

```
3     Definir misDatos, var Como Entero;
4
5     Dimension misDatos[3];
6
7     misDatos[1] ← 10;
8
9     var ← misDatos[1];
10
11    Escribir var;
```



# ARREGLOS UNIDIMENSIONALES

## EJEMPLO

- Tenemos un algoritmo que declara e inicializa un arreglo de N datos, de modo que los elementos de índice par valgan 0 y los de índice impar valgan 1.

```
1  Proceso Ejemplo
2
3      Definir arreglo, contador, nDatos Como Entero;
4
5      Dimension arreglo[100];
6      contador ← 0;
7
8      Escribir "¿Cuantos datos desea en el arreglo?";
9      Leer nDatos;
10
11     Mientras contador < nDatos Hacer
12         Si (contador % 2) = 0 Entonces
13             arreglo[contador] ← 0;
14         SiNo
15             arreglo[contador] ← 1;
16         FinSi
17
18         Escribir "dato[", contador, "] = ", arreglo[contador];
19         contador ← contador + 1;
20     FinMientras
21
22     FinProceso
```



# EJERCICIO PRACTICO

- Realiza un programa que permita cargar 15 números en un arreglo.
- Una vez cargados, se necesita que el programa cuente e informe por pantalla cuántas veces se cargó el número 3.

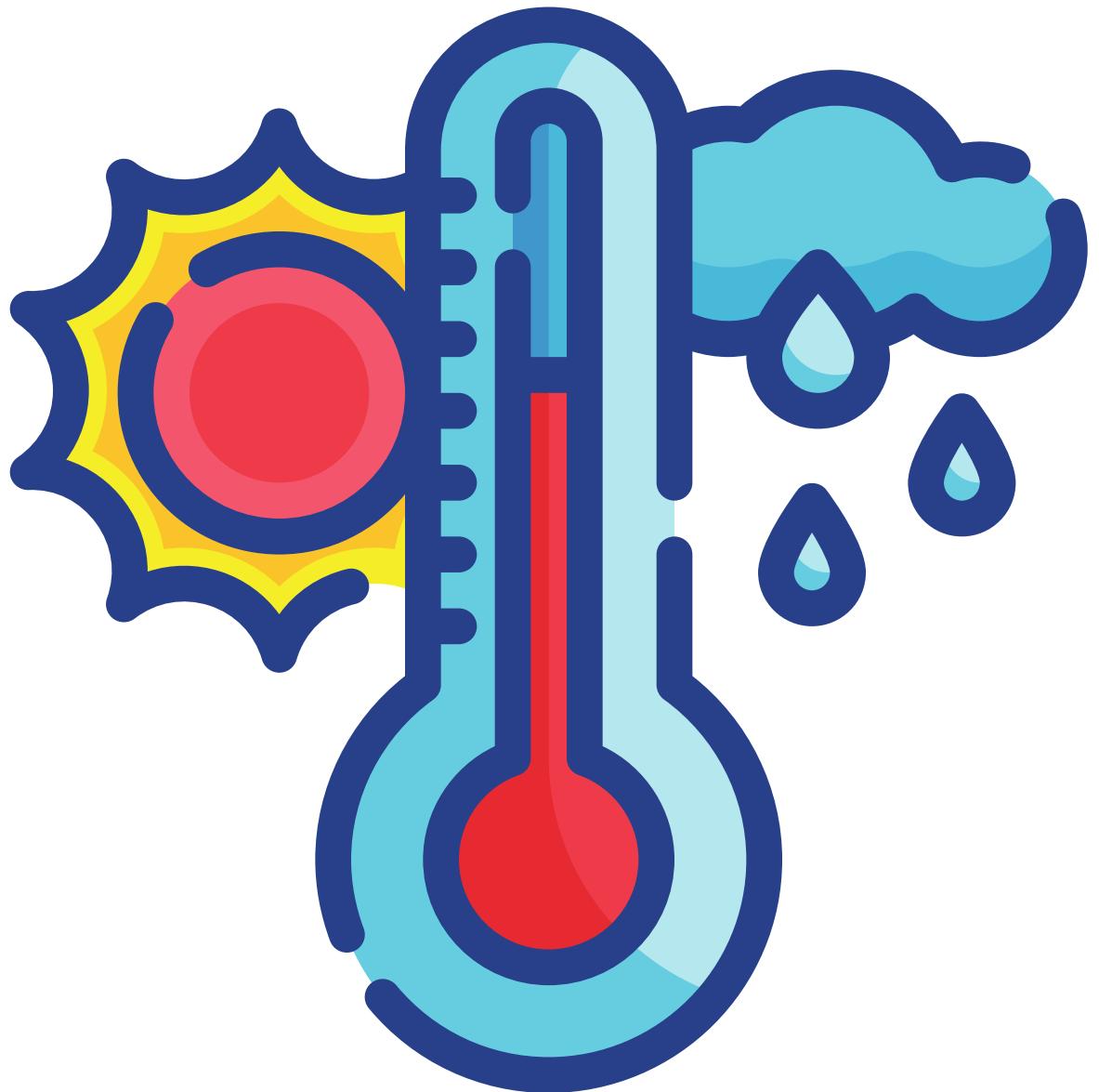


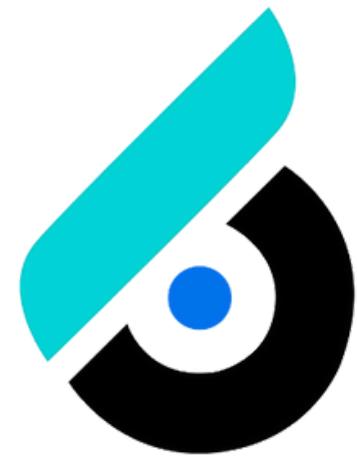


## EJERCICIO PRACTICO

---

- En tres arreglos diferentes guarda los nombres, la temperatura mínima y la máxima de cinco ciudades.
- Se necesita un programa que permita la carga de las ciudades, su temperatura mínima y máxima del día, además, deberá poder informar por pantalla cual es la ciudad con la temperatura más baja y cual tiene la más alta.





# AWAKELAB

#programmingbootcamp

[nodovirtual.awakelab.cl](http://nodovirtual.awakelab.cl)

 jELOU futurejob by  Adalid Chile