Teoría

Pseudocódigo: Estructura de datos

Resolución de Problemas y Algoritmos
Fundamentos de la Programación

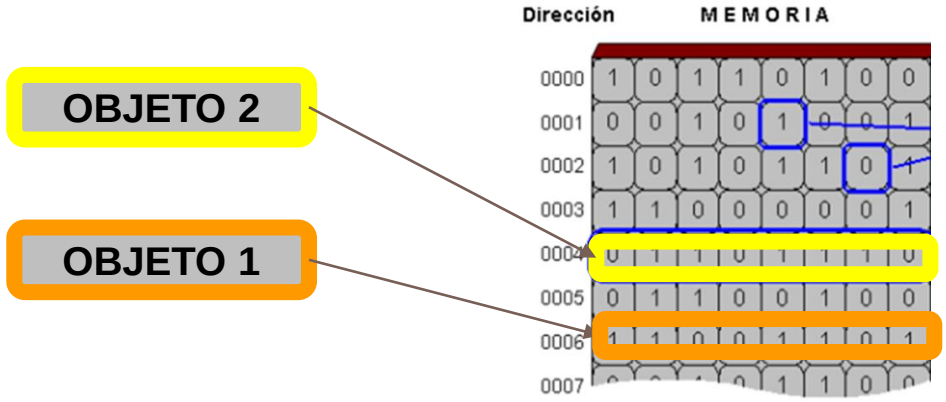
Ingeniería en Computación (TU y TFA)
Ingeniería en Minas (TU)
Profesorado en Ciencias de la Computación (TU y TFA)

Teoría

Pseudocódigo: Estructura de datos

- ✓ **Estructura de datos: Arreglo lineal.**
  - Declaración/definir.
  - Operaciones con arreglos: Asignación y recuperación de valores.
- ✓ **Estructuras de control: secuencial, condicional y de repetición (PARA).**
- ✓ **Pseudocódigo con PSeInt.**

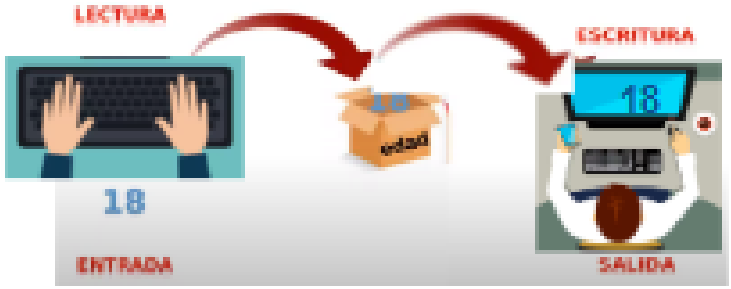
Representación de datos (objetos).



Los **objetos** en los algoritmos escritos en pseudocódigo son: las **variables** y las **constantes**.

Los principales **tipos de datos primitivos o simples** en los que pueden definirse los **objetos** son:

- Numéricos: Enteros y Reales.
- Caracteres
- Booleanos



TIPOS DE DATOS PRIMITIVOS

ENTERO: consiste de un conjunto finito de valores de los números enteros positivos y negativos. Ejemplos: 3, -3, 345, -56.

REAL: consiste de un conjunto finito de valores de los números reales. Números con **punto decimal**, positivos y negativos. Ejemplos: 2.3, -4.6.

LÓGICO: también llamado tipo **booleano**, es el conjunto de los valores de verdad: VERDADERO y FALSO.

CARACTER: es el conjunto finito y "ordenado" de caracteres que el procesador puede reconocer. Ejemplos: 'A', '3', '+'.

las letras mayúsculas del abecedario.
las letras minúsculas del abecedario.
los caracteres numéricos del 0...9.
el caracter de espacio blanco, caracteres especiales tales como: *, +, -, _, /, (,), $, ^, %, $, < ,>, ", .

Objetos.

Variables: tienen la capacidad de almacenar UN dato y cuyo contenido **puede cambiar** durante el ciclo de vida del algoritmo.

Constantes: tienen la capacidad de almacenar UN dato y cuyo contenido **NO puede cambiar** durante el ciclo de vida del algoritmo.

Una **variable** es un objeto del ambiente cuyo valor puede cambiar y que posee además los siguientes atributos:

- un **identificador de variable, nombre** que la identifica,
- un **tipo de dato** que describe los valores que puede tomar la variable y las operaciones que con dicha variable pueden realizarse.

Toda **variable** debe definirse indicando el nombre y el tipo de valores que la misma puede tomar.

Definir <nombre de variable> [,<nombre de variable>]* **como** <tipo>

Ejemplos:
Definir NUMERO **como** Entero
Definir Peso, Suma **como** Real

```
//Identificador de variable

No puede tener espacios;
No puede empezar por un numero;
No puede ser una palabra
reservada;
Distingue mayusculas
```

Tipos de datos.

Tipos de datos

- Primitivos o simples
  - Numéricos
    - Enteros
    - Reales
  - Caracteres
  - Booleanos
- Estructurados
  - Internos
    - Estáticos → Arreglos
    - Dinámicos → Listas, Pilas, Colas, Árboles
  - Externos → Archivos

Estructuras de datos (objetos).

Una **estructura de datos** es una forma particular de organizar datos en una computadora para que pueda ser utilizado de manera eficiente.

Diferentes tipos de estructuras de datos son adecuados para diferentes tipos de aplicaciones, y algunos son altamente especializados para tareas específicas.

Las estructuras de datos son un medio para manejar grandes cantidades de datos de manera eficiente

Una **estructura de datos** es un conjunto de datos reunidos bajo un **único nombre** colectivo.

Arreglo lineal.

Es una estructura de datos, donde se pueden almacenar una colección de datos, todos del **mismo tipo primitivo**.

Homogénea: todos del mismo tipo de dato.

Finita: todo arreglo tiene un límite, se debe determinar la dimensión o cantidad máxima de elementos.

Ordenada: se puede determinar cuál es el primer elemento, el segundo y así siguiendo. Por lo tanto, es posible acceder a un dato en forma directa con solo indicar la posición.

Arreglos lineales.

-3	17	-10				
1	2	3	4	5	6	7

elementos

subíndices

Dimensión= 7

¿Cuál podría ser el nombre de esta variable arreglo?

1. La **dimensión o tamaño del arreglo** es la cantidad de celdas.
2. Los **elementos del arreglo** son los valores de cada celda.
3. Si se quiere recuperar o almacenar un elemento de una celda, se necesita la posición del arreglo a través de los **subíndices**.

IMPORTANTE!

Definir el **tipo de dato** que es el **mismo** para todos los elementos del arreglo**.**

Arreglos en PSeInt.

La definición de arreglos en PSeInt se realiza en **2 partes**:

V

-3	17	-10				
1	2	3	4	5	6	7

En primer lugar, se define el **nombre del arreglo** con el **tipo de dato** asociado a todos los elementos del arreglo. Por ejemplo:

Definir V Como Entero

Luego se establece la dimensión (cantidad de elementos) de la estructura.

Dimension V [7]

En PSeInt el subíndice (posición) del arreglo puede comenzar con diferentes valores del 0 en adelante.

El perfil de la UNSL que se utiliza en la materia establece que el subíndice comienza en 1.

Definición de arreglos en PSeInt.

Definir <identificador > **Como** <tipo >

Dimension <identificador > **[** <max > **]**

Ejemplo:

Definir A Como Real

Dimension A [5]

Actividad 1: Graficar el arreglo del ejemplo dado y contestar a las siguientes preguntas:

A.¿De qué tipo son los datos que se van a almacenar? Ejemplifique con un elemento en el gráfico realizado.

B.¿Cuál es la dimensión del arreglo?

C.¿Cuál es el nombre del arreglo?

Operaciones con arreglos

Al igual que sucede con cualquier variable simple o primitiva, es posible:

- **ALMACENAR UN VALOR EN UN ELEMENTO DE UN ARREGLO Y**
- **RECUPERAR SU VALOR.**

```
M ← 54
car ← 'k'
flag ← VERDADERO
Leer NUM
Leer Nota
```

```
Escribir NUM
Escribir Nota
Prom ← ( M * 8 ) / 4
```

Operaciones con arreglos: Asignación

Al igual que sucede con cualquier variable simple o primitiva, es posible **asignar UN valor a UN elemento de un arreglo y recuperar su valor**.

Por ejemplo, sea **V** declarado como un arreglo de 7 elementos enteros, entonces la operación de asignación:

$$V[3] \leftarrow -15$$

Indica que al tercer elemento de V se le asigna el valor -15, gráficamente:

V

		-15				
1	2	3	4	5	6	7

Al igual que sucede con cualquier variable simple o primitiva, es posible **almacenar UN valor en UN elemento de un arreglo con Leer**.

Por ejemplo, sea **V** declarado como un arreglo de 7 elementos enteros, entonces la sintaxis es:

Leer V [3]

Indica que al tercer elemento de V se le almacena el valor ingresado por teclado con la sentencia Leer.

Por ejemplo, el usuario teclea el valor 112.

Gráficamente:

V

		112				
1	2	3	4	5	6	7

Operaciones con arreglos: Recuperación de valores

			-2											
1	2	3	4	5	6	7								K

V

Escribir **V** [4]

Para resolver:

1. Recuperar el valor del cuarto elemento del arreglo **V.**
2. Muestra en la pantalla el valor recuperado.

Operaciones con arreglos: Recuperación de valores

			-2											
1	2	3	4	5	6	7								K

V

K ← **V** [4]

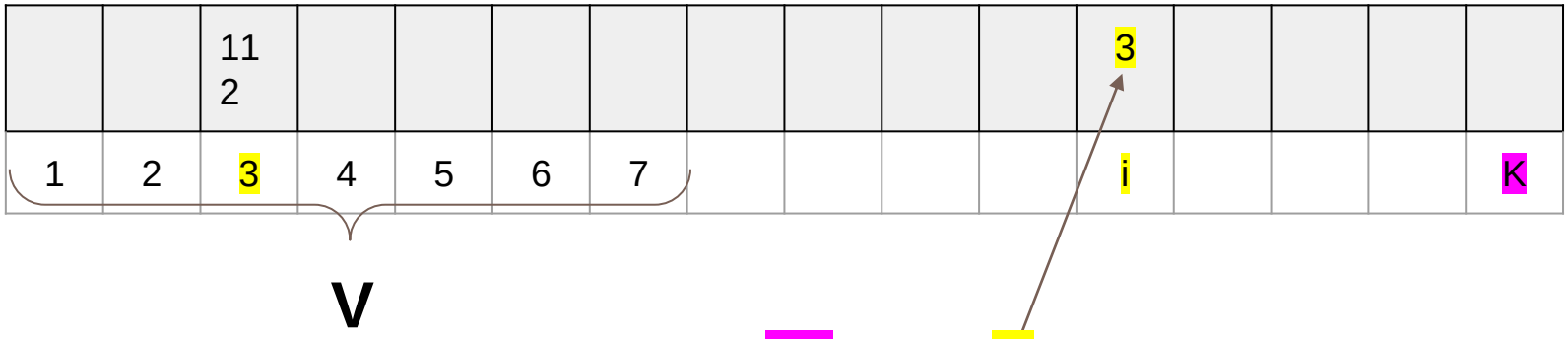
Para resolver:

1. Recuperar el valor del cuarto elemento del arreglo **V.**
2. Asignar a la variable K el valor recuperado.

			-2											**-2**
1	2	3	4	5	6	7								K

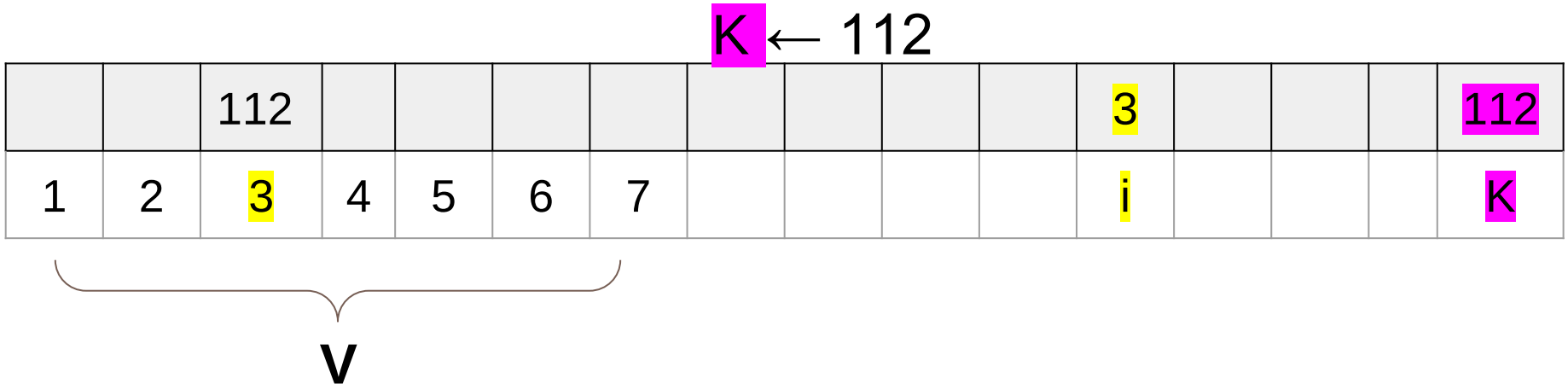
V

Operaciones con arreglos: Recuperación de valores



Para resolver:

1. Recuperar el valor del i-ésimo elemento del arreglo **V**, es decir el valor del tercer elemento.
2. Asignar a la variable K el valor recuperado.



Operaciones con arreglos: Recuperación de valores

			-2								4		3			
1	2	3	4	5	6	7					i		J			K

V

K ← 5 * J + **V** [i]

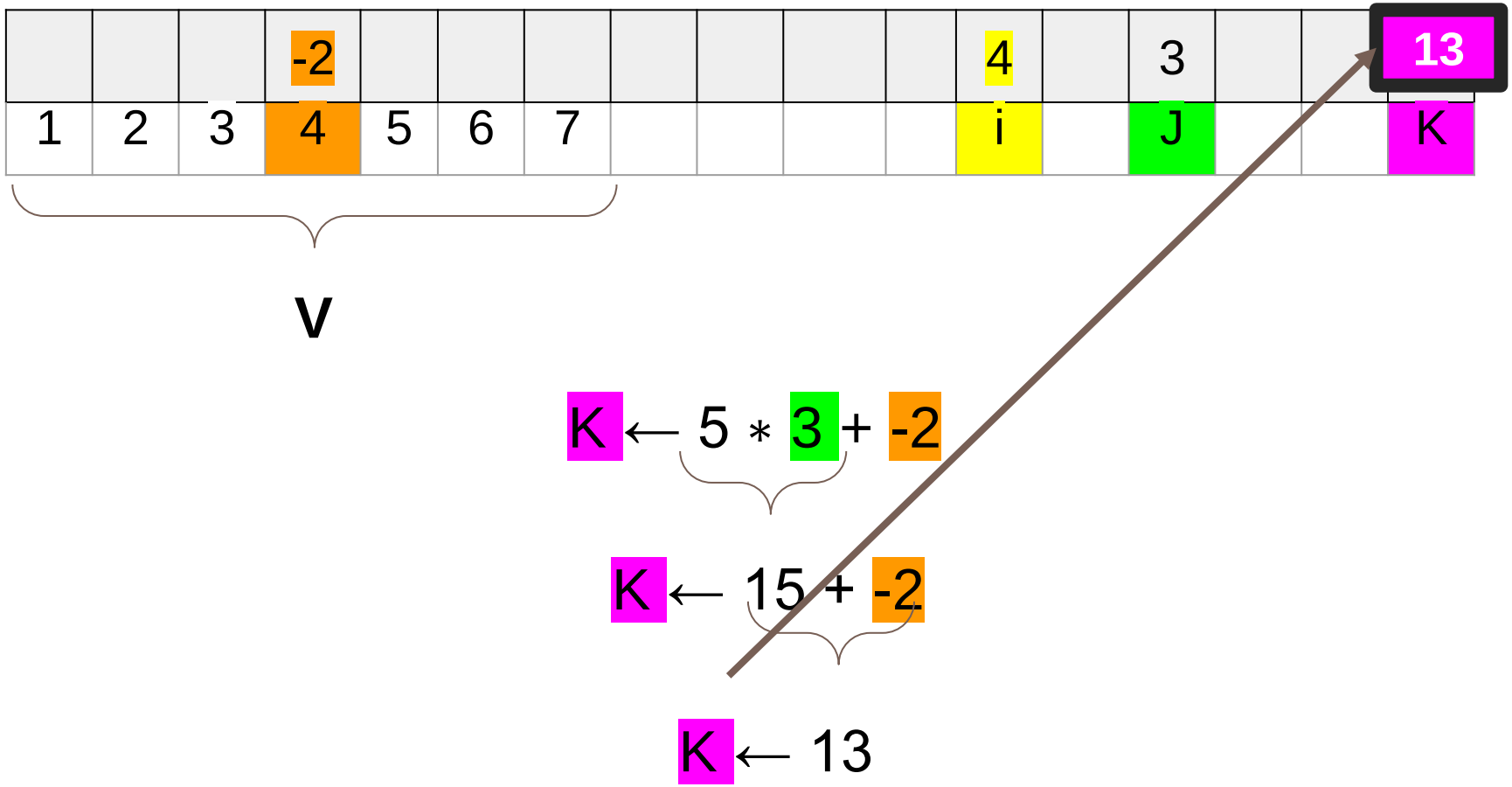
Para resolver:

1. Recuperar el valor de la variable J
2. Recuperar el valor de la variable i

K ← 5 * 3 + **V** [4]

3.Recuperar el valor de V[4]

K ← 5 * 3 + -2



IMPORTANTE!

Los valores del arreglo **V,** y de las variables **i** y **J** se conservan igual, no se modifican, ni se sobreescriben en estas sentencias.

Operaciones con arreglos: Recuperación de valores

33	25	9	12	39							4		3			
1	2	3	4	5	6						s		J			

Res

Actividad 2: Completar la tabla de ejecución para las sentencias.

Escribir Res[s]
Escribir Res[J+1]

Acciones	**Res**								
	1	**2**	**3**	**4**	**5**	**6**	**s**	**J**	**Pantalla**
	33	25	9	12	39		4	3	
Escribir Res[s]									
Escribir Res[J+1]									

‘M’	‘a’	‘r’	‘e’				
1	2	3	4	5	6	7	8

1.23	5.67	6.78	2.4							
1	2	3	4	5	6	7	8	9	10	11

verdadero	falso	falso	falso	verdadero
1	2	3	4	5

Actividad 3: Dados 3 arreglos.

1. Realizar la declaración de dichos arreglos en PSeInt.
2. Escribir la sentencia para mostrar en pantalla el cuarto elemento de cada uno.
3. ¿Cuál sería el valor que mostraría para cada uno al ejecutar el ejercicio anterior?

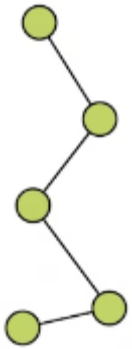
Estructuras de control.

En la programación estructurada, se disponen de estructuras de control de flujo necesarias para desarrollar un programa.

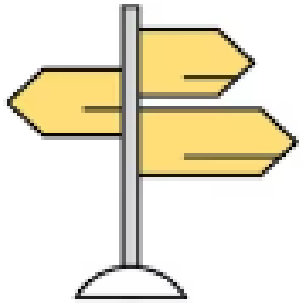
Esto hace referencia al orden en el que se ejecutarán las instrucciones de un programa, desde su comienzo hasta que finaliza.

Vamos a utilizar **3 estructuras de control**:

secuencial



condicional o selección



repetición o iteración

```
1  Proceso ParImpar
2      Definir nro Como Entero
3      Escribir "Ingrese un número cualquiera"
4      Leer nro
5  FinProceso
```

<u>Secuencial</u>: las tareas (acciones) se deben realizar en el orden en que se escriben, es decir, primero una, luego otra, desde la primera hasta la última (o de arriba hacia abajo).

```
5      si nro MOD 2 = 0 Entonces
6          Escribir "El numero es par"
7      sino
8          Escribir "El numero es impar"
9      FinSi
```

<u>Selección (Condicional)</u>: las tareas (acciones) se realizarán dependiendo de cierta situación, estado previo o **condición** que se debe cumplir.

```
5      Mientras nro <= 0 hacer
6          Escribir "Ingrese un número cualquiera"
7          Leer nro
8      FinMientras
```

```
11     Mientras cont <= 4 Hacer
12         Escribir "Ingrese un número cualquiera"
13         Leer nro
14         si nro MOD 2 = 0 Entonces
15             Escribir "El numero es par"
16         sino
17             Escribir "El numero es impar"
18         FinSi
19         con<-cont+1
20     FinMientras
```

<u>Iteración (Repetición)</u>: una tarea o conjunto de tareas (acciones) se deben realizar en forma repetitiva una **cantidad de veces específica** o dependiendo de una **condición** que se debe cumplir.

Repetición (Iteración)

Iteración simple

Mientras <condicion> **Hacer**

<secuencia de acciones>

FinMientras

```
cont<-1
Mientras cont <= 4 Hacer
    Escribir "Ingrese un número cualquiera"
    Leer nro
    si nro MOD 2 = 0 Entonces
        Escribir "El numero es par"
    FinSi
    cont<-cont+1
FinMientras
```

Iteración condicional

Mientras <condicion> **Hacer**

<secuencia de acciones>

FinMientras

```
Mientras nro <= 0 hacer
    Escribir "Ingrese un número cualquiera"
    Leer nro
FinMientras
```

Iteración simple

Mientras <condicion> **Hacer**

<secuencia de acciones>

FinMientras

```
cont<-1
Mientras cont <= 4 Hacer
    Escribir "Ingrese un número cualquiera"
    Leer nro
    si nro MOD 2 = 0 Entonces
        Escribir "El numero es par"
    FinSi
    cont<-cont+1
FinMientras
```

Para trabajar con arreglos, puede resultar útil disponer de una estructura de control repetitiva que permita que una secuencia de acciones se ejecute un **número fijo** de veces, conocido de antemano.

Además, de la sentencia de iteración simple **Mientras**.

```
cont<-1
Mientras cont <= 4 Hacer
    Escribir "Ingrese un número cualquiera"
    Leer nro
    si nro MOD 2 = 0 Entonces
        Escribir "El numero es par"
    FinSi
    cont<-cont+1
FinMientras
```

En la estructura de control Iteración simple tenemos:

Iteración simple

V ← Vinicial

Mientras V <= Vfinal **hacer**

<secuencia de acciones>

V ← V + P

FinMientras

Teniendo en cuenta las partes de la estructura de control Iteración simple:

Iteración simple

```
V ← Vinicial
Mientras V <= Vfinal hacer
        <secuencia de
acciones>
        V ← V + P
FinMientras
```

El formato de esta nueva estructura de control es el siguiente:

```
Para V ← Vinicial Hasta Vfinal Con Paso P Hacer
    <secuencia de acciones>
FinPara
```

Entonces este código escrito con la sentencia **Mientras**:

```
cont<-1
Mientras cont <= 4 Hacer
    Escribir "Ingrese un número cualquiera"
    Leer nro
    si nro MOD 2 = 0 Entonces
        Escribir "El numero es par"
    FinSi
    cont<-cont+1
FinMientras
```

Se puede escribir con la sentencia **Para** como:

```
Para cont<-1 hasta 4 con paso 1 hacer
    Escribir "Ingrese un número cualquiera "
    Leer nro
    si nro MOD 2 = 0 Entonces
        Escribir "El numero es par"
    FinSi
FinPara
```

Partes de la nueva estructura de control es el siguiente:

Para V ← Vinicial **Hasta** Vfinal **Con Paso** P **Hacer**
    <secuencia de acciones>
FinPara

Donde:

- V es una variable de **tipo entero** llamada **variable de control** de la repetición.

- Vinicial, Vfinal y P pueden ser variables o constantes de tipo entero o expresiones aritméticas.

El formato de esta nueva estructura de control es el siguiente:

Para V ← Vinicial **Hasta** Vfinal **Con Paso** **P** **Hacer**
    <secuencia de acciones>
FinPara

Donde:

- Vinicial recibe el nombre de valor inicial;
- Vfinal recibe el nombre de valor final y

- **P** es el paso, o sea en cuanto se incrementa (o decrementa) la variable V para llegar desde Vinicial al Vfinal y debe ser distinto de cero (pero puede ser positivo o negativo).

Para V ← Vinicial **Hasta** Vfinal **Con Paso** P **Hacer**
<secuencia de acciones>
FinPara

Altura											
	1.23	1.67	1.78	1.6	1.5						
	1	2	3	4	5	6	7	8	9	10	11

```
1  // Almacenar los datos en el arreglo Alturas
2
3  Proceso leerAlturas
4      Definir Altura como Real
5      Dimension Altura[11]
6      Definir i,N como Entero
7
8      N<-5
9
10     i<-1
11     Mientras i <= N Hacer
12         Escribir "Ingrese la altura ",i,":"
13         Leer Altura[i]
14     i<-i+1
15     FinMientras
16
17 FinProceso
```

```
// Almacenar los datos en el arreglo Altu

Proceso leerAlturas
    Definir Altura como Real
    Dimension Altura[11]
    Definir i,N como Entero

    N<-5

    Para i<-1 Hasta N Con Paso 1 Hacer
        Escribir "Ingrese el dato ",i,":"
        Leer Altura[i]
    FinPara


FinProceso
```

```
1   // Almacenar los datos en el arreglo Alturas
2
3   Proceso leerAlturas
4       Definir Altura como Real
5       Dimension Altura[11]
6       Definir i,N como Entero
7
8       N<-5
9
10      Para i<-1 Hasta N Con Paso 1 Hacer
11          Escribir "Ingrese el dato ",i,":"
12          Leer Altura[i]
13      FinPara
14
15
16  FinProceso
17
```

Actividad 4: Identificar en el algoritmo escrito en PSeInt.

1. ¿Cuál es el nombre del arreglo? ¿Cuál su dimensión?
2. En la sentencia PARA especifique:

variable de control	
valor de inicio	
valor final	
valor del paso	

¿Cómo almacenar los valores de elementos de un arreglo?

Altura

1.23	1.67	1.78	1.6	1.5						
1	2	3	4	5	6	7	8	9	10	11

```
// Almacenar los datos en el arreglo Alturas

Proceso leerAlturas
    Definir Altura como Real
    Dimension Altura[11]
    Definir i,N como Entero

    N<-5

    Para i<-1 Hasta N Con Paso 1 Hacer
        Escribir "Ingrese la altura ",i,":"
        Leer Altura[i]
    FinPara

FinProceso
```

Almacenando valores de **N** elementos del arreglo, o sea de 5 elementos del arreglo.

Almacena valor de un elemento del arreglo

¿Cómo muestro en pantalla los elementos de un arreglo?

Altura

1.23	1.67	1.78	1.6	1.5						
1	2	3	4	5	6	7	8	9	10	11

```
// Almacenar los datos en el arreglo Alturas

Proceso leerAlturas
    Definir Altura como Real
    Dimension Altura[11]
    Definir i,N como Entero

    N<-5

    Para i<-1 Hasta N Con Paso 1 Hacer
        Escribir "Ingrese la altura ",i,":"
        Leer Altura[i]
    FinPara

    Para i<-1 Hasta N Con Paso 1 Hacer
        Escribir Altura[i]
    FinPara

FinProceso
```

Almacena la altura en **N** elementos

¿Cómo muestro en pantalla los elementos de un arreglo?

Altura

1.23	1.67	1.78	1.6	1.5						
1	2	3	4	5	6	7	8	9	10	11

```
// Almacenar los datos en el arreglo Alturas

Proceso leerAlturas
    Definir Altura como Real
    Dimension Altura[11]
    Definir i,N como Entero

    N<-5

    Para i<-1 Hasta N Con Paso 1 Hacer
        Escribir "Ingrese la altura ",i,":"
        Leer Altura[i]
    FinPara

    Para i<-1 Hasta N Con Paso 1 Hacer
        Escribir Altura[i]
    FinPara

FinProceso
```

Muestra valores de **N** elementos del arreglo, o sea de 5 elementos del arreglo.

Muestra el valor de un elemento del arreglo

1

250	67	478	240					
1	2	3	4	5	6	7	8	9

2

verdadero	falso	falso	falso	verdadero
1	2	3	4	5

3

‘M’	‘a’	‘r’	‘e’’				
1	2	3	4	5	6	7	8

Actividad 5: en equipos de a dos.

Realizar el **algoritmo en PSeInt** para almacenar y mostrar los valores del arreglo que se le asignó.

Teoría

Pseudocódigo: Estructura de datos

- ✓ **Estructura de datos: Arreglo lineal.**
  - Declaración.
  - Operaciones con arreglos: Asignación y recuperación de valores.
- ✓ **Estructuras de control: secuencial, condicional y de repetición (PARA).**
- ✓ **Pseudocódigo con PSeInt.**

Teoría

PSeInt y DF

¡Ya podemos comenzar con el Práctico!