



El futuro digital
es de todos

MinTIC

Estructuras de Datos:

Listas dentro de Listas

Tuplas

Diccionarios

»
Misión TIC 2022

xxx



Misión
TIC 2022



El futuro digital
es de todos





MinTIC



Temas – Sesión 1



Estructura de Datos

-  Listas dentro de listas
-  Tuplas
-  Conjuntos
-  Diccionarios



El futuro digital
es de todos

MinTIC



Estructura de Datos



Estructura de Datos – Listas dentro de listas



Listas dentro de listas

Listas dentro de Listas (Matrices)

MEMORIA

		Columnas				
Filas	1	2	3	4		
	5	6	7	8		
	9	10	11	12		

Matriz de Números: En esta matriz se almacenan números, que serán utilizados en el proceso. En este caso, la matriz maneja la dimensión horizontal ó fila y la dimensión vertical o columna.

Número de Filas: 3

Número de Columnas: 4



Listas dentro de listas

Listas dentro de Listas (Matrices) .

1	2	3
4	5	6

Matriz

Filas y Columnas empiezan en cero

Fila 0=> 1,2,3

Fila 1=>4,5,6

Para referenciar un elemento de la matriz:

1=matriz[0][0]

Matriz[i][j] para referenciar un elemento de la matriz. El que se encuentra en la fila i, columna j



Listas dentro de listas

Listas dentro de Listas (Matrices) . En Python

```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bi
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> matriz=[[1,2,3],[4,5,6]]
>>> matriz
[[1, 2, 3], [4, 5, 6]]
>>> matriz[0]
[1, 2, 3]
>>> matriz[1]
[4, 5, 6]
>>> matriz[0][0]
1
>>> matriz[1][1]
5
>>> |
```



Listas dentro de listas

Listas dentro de Listas (Matrices) .

Para el recorrido de una matriz (lista dentro de una lista), es decir pasar por cada uno de sus elementos, se utilizan DOS ciclos FOR anidados, uno dentro de otro, uno para las filas y otro para las columnas

Matriz Letras	a	t	i	Fila 0		
	f	o	j	Fila 1		
	u	r	e	Fila 2		
	Col 0	Col 1	Col 2			
Recorrer la matriz: Pasar por cada uno de los elementos=> 2 ciclos anidados						
Ciclo fila	0	1	2			
Ciclo Columna	0,1,2	0,1,2	0,1,2			



El futuro digital
es de todos

MinTIC

Estructuras de Control

Ejercicios

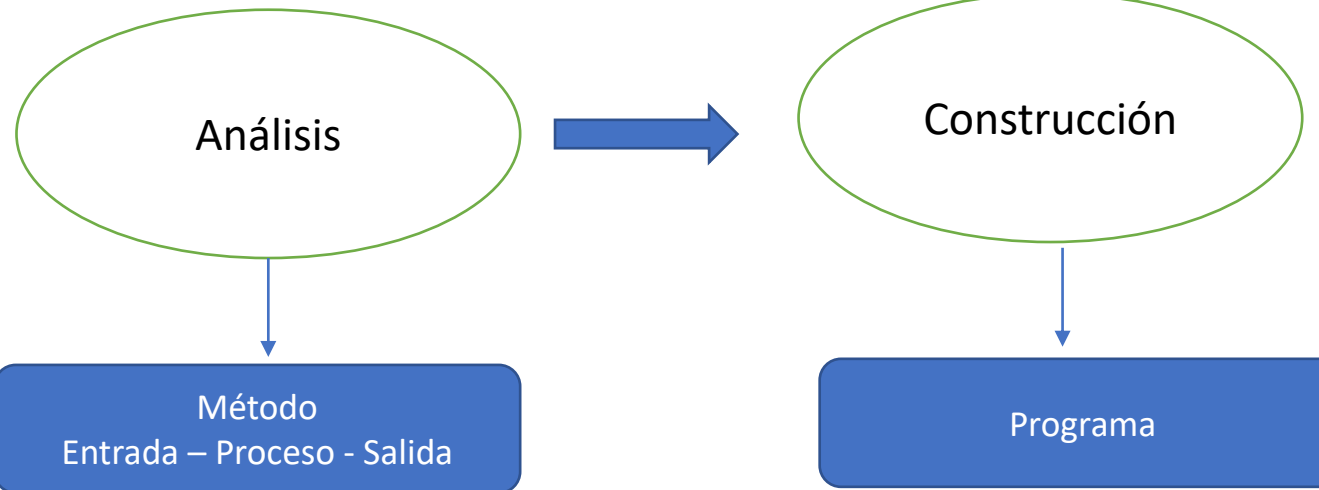


Dada una lista dentro de lista (**matriz**) de 2 filas X 2 **columnas** que almacena números enteros, mostrar la **cantidad de números pares e impares** que hay en la matriz.



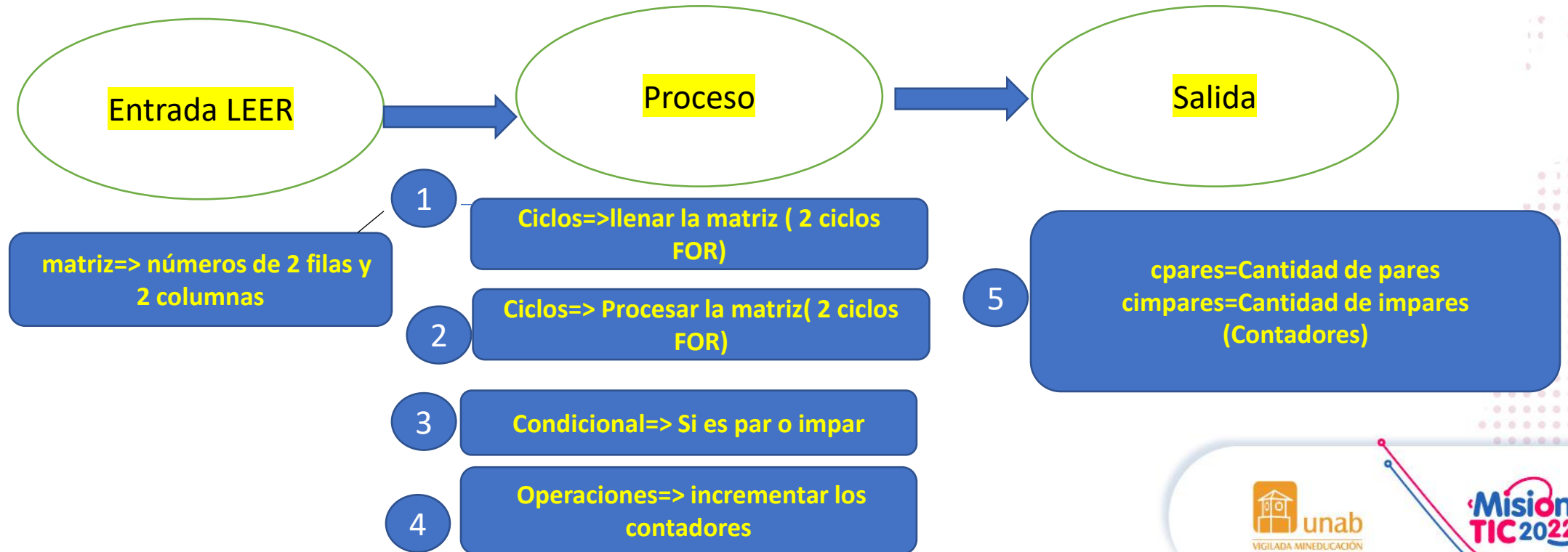
Listas dentro de listas - Ejercicio

Metodología -> Pensamiento lógico estructurado





Análisis → Ejercicio





Estructura de Datos

Construcción → Programa

Ejercicios



```
# Programa de listas dentro de listas
# Autor: Sergio Medina
# Fecha: 15/05/2022

#llenar la lista dentro de lista (matriz)
numeros=[]
for i in range(2):
    numeros.append([])
    for j in range(2):
        numeros[i].append(int(input("Número: ")))
#Imprimir lista dentro de lista (matriz)
print(numeros)
#Procesar lista dentro de lista
cpares=0
cimpares=0
for i in range(2):
    for j in range(2):
        if numeros[i][j]%2==0:
            cpares+=1
        else:
            cimpares+=1
print("Cantidad de pares: ",cpares)
print("Cantidad de impares: ",cimpares)
```



Estructura de Datos

```
# Programa de listas dentro de listas
# Autor: Sergio Medina
# Fecha: 15/05/2022

#Funciones
def valida_entero(etiqueta):
    while True:
        try:
            dato=int(input(etiqueta))
            break
        except ValueError:
            print(etiqueta, " debe ser entero")
    return dato

#llenar la lista dentro de lista (matriz)
numeros=[]
for i in range(2):
    numeros.append([])
    for j in range(2):
        num=valida_entero("Número: ")
        numeros[i].append(num)

#Imprimir lista dentro de lista (matriz)
print(numeros)

#Procesar lista dentro de lista
cpares=0
cimpares=0
for i in range(2):
    for j in range(2):
        if numeros[i][j]%2==0:
            cpares+=1
        else:
            cimpares+=1

print("Cantidad de pares: ",cpares)
print("Cantidad de impares: ",cimpares)
```

Ejercicios



Construcción → Programa – versión 2
(Validación)



El futuro digital
es de todos

MinTIC



Estructura de Datos



Estructura de Datos - Tuplas



Tuplas



Las tuplas son estructuras que, una vez creadas o definidas, NO permiten modificarse a lo largo de la ejecución de un programa, lo cual les da la caracterización de ser **estructuras inmutables**.





Tuplas – Métodos – Práctica IDE Python

x
x
x

```
>>> tupla=(1,"Juan",5,"Pedro",10.5,[2-3],"Sergio")
```

```
>>> tupla
```

```
(1, 'Juan', 5, 'Pedro', 10.5, [-1], 'Sergio')
```

```
>>> tupla[1]
```

```
'Juan'
```

```
>>> tupla.append(15)
```

```
Traceback (most recent call last):
```

```
  File "<pyshell#3>", line 1, in <module>
```

```
    tupla.append(15)
```

```
AttributeError: 'tuple' object has no attribute 'append'
```

```
>>> tupla.extend([30,40])
```

```
Traceback (most recent call last):
```

```
  File "<pyshell#4>", line 1, in <module>
```

```
    tupla.extend([30,40])
```

```
AttributeError: 'tuple' object has no attribute 'extend'
```

```
>>> tupla.remove(5)
```

```
Traceback (most recent call last):
```

```
  File "<pyshell#5>", line 1, in <module>
```

```
    tupla.remove(5)
```

```
AttributeError: 'tuple' object has no attribute 'remove'
```

Tuplas No se pueden modificar

Estructura de Datos



El futuro digital
es de todos

MinTIC



Conjuntos y Diccionarios



Conjuntos



Un **conjunto** es una colección no ordenada de objetos únicos. Python provee este tipo de datos «por defecto» al igual que otras colecciones más convencionales como las listas, tuplas y diccionarios.

Los conjuntos son ampliamente utilizados en lógica y matemática, y desde el lenguaje podemos sacar provecho de sus propiedades para crear código más eficiente y legible en menos tiempo.



Conjuntos – Creación



Un conjunto se crea colocando todos los elementos (elementos) entre llaves {}, separados por comas o usando la función incorporada set(). Puede tener cualquier número de elementos y pueden ser de diferentes tipos (entero, flotante, tupla, cadena, etc.). Sin embargo, un conjunto no puede contener elementos mutables como listas, conjuntos o diccionarios.

```
Conjunto = {12, 22, 33}
```

Es importante recordar que, los elementos de un conjunto no tiene un orden predefinido como las listas o las tuplas; por lo tanto, las posiciones de los elementos no importan y **pueden variar entre la definición y la impresión del conjunto.**



Conjuntos – Métodos



```
>>> conjunto1={1,2,4,"Pedro","Luis"}
>>> conjunto1
{1, 2, 4, 'Luis', 'Pedro'}
>>> conjunto2=set([2,5,6,"Mara","Luis"])
>>> conjunto2
{2, 5, 6, 'Luis', 'Mara'}
>>> conjunto1.add("Hugo")
>>> conjunto1
{1, 2, 4, 'Hugo', 'Luis', 'Pedro'}
>>> conjunto1.remove(2)
>>> conjunto1
{1, 4, 'Hugo', 'Luis', 'Pedro'}
>>>
```

Métodos

Add: Adicionar elementos

Remove: Eliminar elementos



Conjuntos – Métodos: operaciones conjuntos

x
x
x

```
>>> A={1, 3, 5, 7, 9, 12}
>>> B={5, 7, 9, 15, 20, 30}
>>> C=A | B
>>> C
{1, 3, 5, 7, 9, 12, 15, 20, 30}
>>> D=A & B
>>> D
{9, 5, 7}
>>> E=A - B
>>> E
{1, 3, 12}
>>> F=B - A
>>> F
{20, 30, 15}
```

Operaciones
Conjuntos

Unión: |

Intersección: &

Diferencia: -



Diccionarios



Los diccionarios son estructuras de datos que permiten almacenar **valores** indexados, a través de **claves**, lo cual permite ordenar datos a través de la clave y realizar una búsqueda más eficiente

Para crear un diccionario, empaquetamos entre llaves { } cada par de elementos **Clave:Valor** separadas por comas, así:

```
empleado = {'nombre':"Sergio", 'apellido':"Medina", 'edad':57, 'salario':3500000}
```





Diccionarios

x
x
x

Una vez que almacenamos los datos en el diccionario, vamos a acceder a ellos

Con el **método get()** de un diccionario, podemos obtener el valor de una clave, pero si no existe la clave devolver un mensaje en string como respuesta.

```
>>> empleado={'nombre':"Sergio Medina",'cargo':"Programador",'salario':4000000}
>>> empleado
{'nombre': 'Sergio Medina', 'cargo': 'Programador', 'salario': 4000000}
>>> empleado['nombre']
'Sergio Medina'
>>> empleado.get('nombre')
'Sergio Medina'
>>> empleado['email']
Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    empleado['email']
KeyError: 'email'
>>> empleado.get('email',"NO ENCONTRADO")
'NO ENCONTRADO'
```

Cuando la clave
NO es encontrada
en el diccionario



Diccionarios - Operaciones

x
x
x

```
>>> articulos={1:"Lapiz",2:"Borrador",3:"Cuadernos"}
>>> articulos
{1: 'Lapiz', 2: 'Borrador', 3: 'Cuadernos'}
>>> articulos[4]="Calcualdora"
>>> articulos
{1: 'Lapiz', 2: 'Borrador', 3: 'Cuadernos', 4: 'Calcualdora'}
>>> articulos[4]="Calculadora"
>>> articulos
{1: 'Lapiz', 2: 'Borrador', 3: 'Cuadernos', 4: 'Calculadora'}
>>> articulos[5]="Refresco"
>>> articulos
{1: 'Lapiz', 2: 'Borrador', 3: 'Cuadernos', 4: 'Calculadora', 5: 'Refresco'}
>>> del articulos[5]
>>> articulos
{1: 'Lapiz', 2: 'Borrador', 3: 'Cuadernos', 4: 'Calculadora'}
```

Agregar

Modificar

Eliminar

Operaciones



Diccionarios - Ejercicio



Se realiza la compra de N artículos, en donde se ingresa el código del artículo y la cantidad y mediante el uso de diccionarios para los nombres y valores unitarios de los artículos, el programa debe obtener el nombre de cada artículo, cantidad comprada, valor unitario, valor total de acuerdo a la cantidad comprada y finalmente calcular el valor total de la compra.



Se suministra el diccionario de nombres de artículo y otro con los valores unitarios.

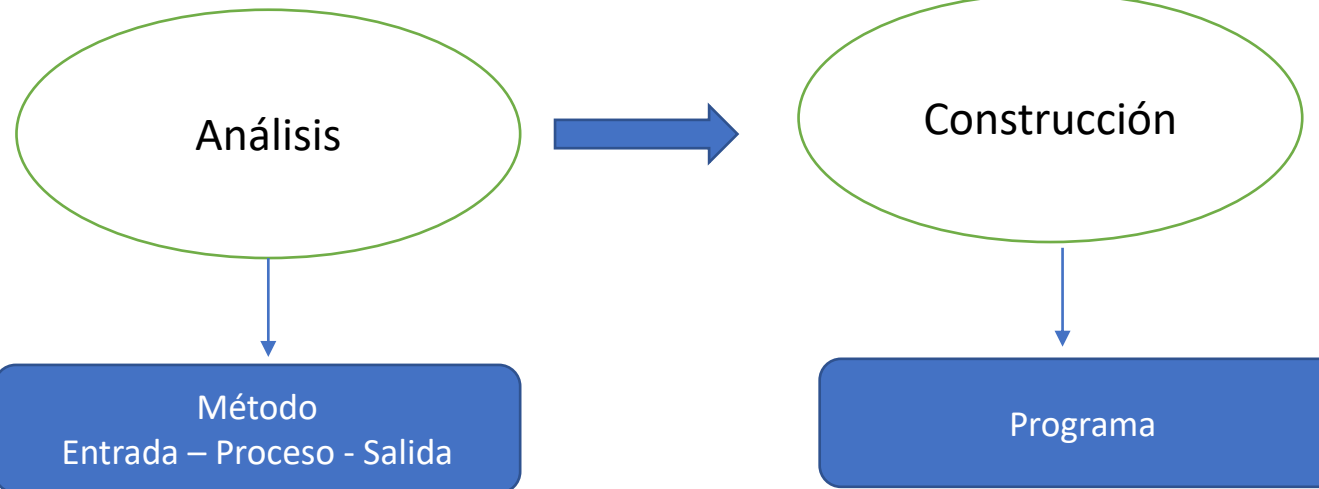
`articulos={1:"Lapiz",2:"Cuadernos",3:"Borrador",4:"Calculadora",5:"Escuadra"}`

`valores={1:2500,2:3800,3:1200,4:35000,5:3700}`



Diccionarios - Ejercicio

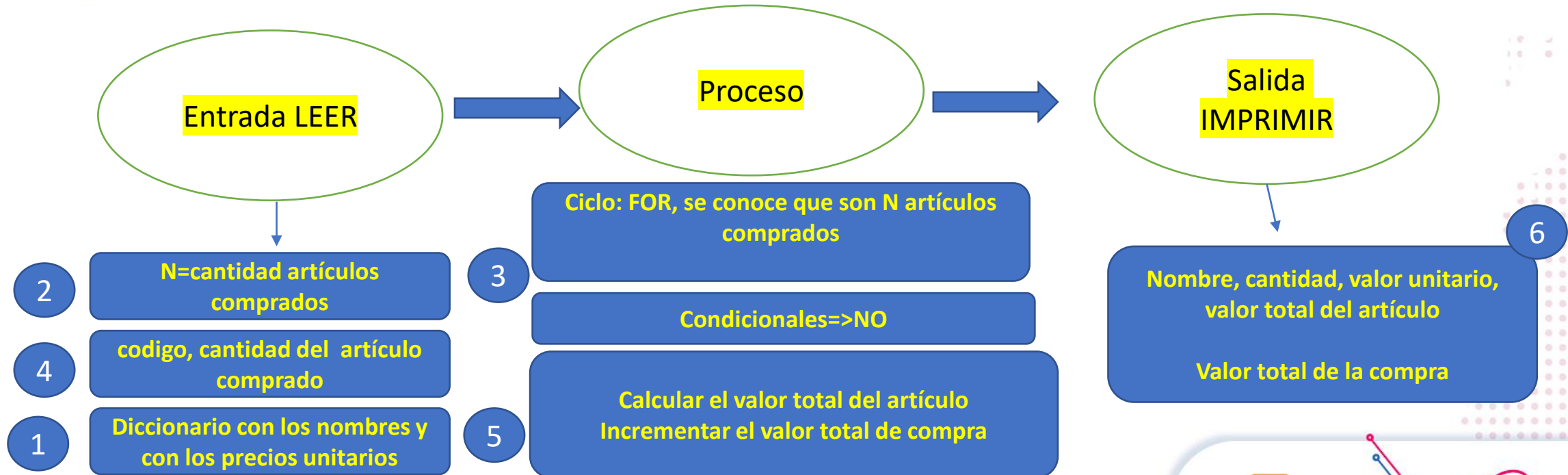
Metodología -> Pensamiento lógico estructurado





Diccionarios - Ejercicio

Análisis → Ejercicio Listas





Diccionario - Ejercicio

Construcción - Programa – Ver 1



```
# Programa para manejo de diccionarios
# Autor: Sergio Medina}
# Fecha: 14/05/2022

# Diccionarios
articulos={1:"Lapiz",2:"Cuadernos",3:"Borrador",4:"Calculadora",5:"Escuadra"}
valores={1:2500,2:3800,3:1200,4:35000,5:3700}

N=int(input("cantidad de artículos comprados: "))
total_compra=0
for i in range(N):
    codigo=int(input("Código artículo: "))
    cantidad=int(input("Cantidad comprada: "))
    valor_articulo=cantidad*valores.get(codigo)
    total_compra+=valor_articulo
    print("Artículo: ",articulos.get(codigo))
    print("Cantidad comprada: ",cantidad)
    print("Valor unitario: ", "{:,.2f}".format(valores.get(codigo)))
    print("Valor artículo: ", "{:,.2f}".format(valor_articulo))
print("Valor total compra: ", "{:,.2f}".format(total_compra))
```



Diccionario - Ejercicio

x
x
x

Análisis - Modularidad

Parámetros de entrada

nombre



MODULO (FUNCION)
calcular_valor_articulo:

Calcular el valor del articulo con la
cantidad y valor unitario



Parámetros de salida

Cantidad_palabras

FUNCIÓN retorna o regresa un solo valor

Parte

5



El futuro digital
es de todos

MinTIC

```
# Programa para manejo de diccionarios
# Autor: Sergio Medina}
# Fecha: 14/05/2022

X
X # Diccionarios
X articulos={1:"Lapiz",2:"Cuadernos",3:"Borrador",4:"Calculadora",5:"Escuadra"}
valores={1:2500,2:3800,3:1200,4:35000,5:3700}

# Funciones
def valida_entero(etiqueta):
    while True:
        try:
            dato=int(input(etiqueta))
            break
        except ValueError:
            print(etiqueta," debe ser un número entero")
    return dato

def valida_codigo(etiqueta):
    while True:
        try:
            dato=int(input(etiqueta))
            if articulos.get(dato,"ERROR")== "ERROR":
                print(etiqueta," NO existe en el diccionario")
                continue
            break
        except ValueError:
            print(etiqueta," debe ser un número entero")
    return dato

def calcular_valor_articulo(cantidad,valor_unitario):
    valor_articulo=cantidad*valor_unitario
    return valor_unitario
```

Con validación de entrada de datos
y de existencia de clave en lista



```
N=valida_entero("Cantidad de artículos comprados: ")
total_compra=0
for i in range(N):
    codigo=valida_codigo("Código artículo: ")
    cantidad=valida_entero("Cantidad comprada: ")
    valor_articulo=calcular_valor_articulo(cantidad, valores.get(codigo))
    total_compra+=valor_articulo
    print("Artículo: ", artículos.get(codigo))
    print("Cantidad comprada: ", cantidad)
    print("Valor unitario: ", "{:,.2f}".format(valores.get(codigo)))
    print("Valor artículo: ", "{:,.2f}".format(valor_articulo))
    print("Valor total compra: ", "{:,.2f}".format(total_compra))
```

Con validación de entrada de datos y de existencia de clave en lista



El futuro digital
es de todos

MinTIC

» Estructura de Datos



Ejercicios Propuestos



El futuro digital
es de todos

MinTIC

Estructuras de Datos:

Listas dentro de Listas

Tuplas

Diccionarios

»
Misión TIC 2022

xxx



Misión
TIC 2022