



CICLO 1

[FORMACIÓN POR CICLOS]

Fundamentos de **PROGRAMACIÓN**



Ingeni@
Soluciones TIC



UNIVERSIDAD
DE ANTIOQUIA

Facultad de Ingeniería

Lectura

INSTRUCCIONES

de lectura y escritura





Como vimos en la lectura anterior, las estructuras lógicas para la construcción de un algoritmo son:

- 1. Estructura secuencia**
- 2. Estructura decisión**
- 3. Estructura ciclo**

Cada estructura consta de un conjunto de instrucciones. Las instrucciones correspondientes a la estructura secuencia son:

- 1. Instrucciones de lectura**
- 2. Instrucciones de escritura**
- 3. Instrucciones de asignación**
- 4. Las instrucciones correspondientes a la estructura decisión**
- 5. Las instrucciones correspondientes a la estructura ciclo**

InSTRUCCIÓN DE LECTURA.

Para que el computador pueda procesar datos, estos deben estar en la memoria principal (RAM). La instrucción de lectura consiste en llevar los datos con los cuales se desea trabajar desde un medio externo hacia la memoria principal. Los medios externos en los cuales pueden residir los datos son: disco duro, disco removible, diskette, cinta, etc., o entrarlo directamente a través del teclado.

La forma general de la instrucción de lectura en Python es:

```
variable = input("mensaje de entrada de dato: ")
```

Tenga presente que todo dato que se lea usando este formato se almacena como si fuera una hilera de símbolos.

Si se quiere leer el nombre y el apellido de una persona, las instrucciones son:

```
nombre = input("Por favor teclee su nombre ")
apellido = input("Ahora teclee su apellido ")
```



Al ejecutar esas dos instrucciones, en la variable **nombre** queda almacenado el nombre tecleado, y en la variable **apellido** queda almacenado el apellido tecleado.

Instrucción de escritura.

La instrucción de escritura consiste en llevar datos desde la memoria hacia un medio externo, el cual puede ser un disco duro, una cinta, una impresora, etc.

La forma general de la instrucción de escritura en Python es:

```
print([lista de mensajes y variables separados por comas])
```

Los mensajes sirven para describir al usuario los datos que se le están presentando. Si el dato que se imprime es la estatura de una persona, es conveniente que esté precedido por un mensaje que diga: estatura. Si el dato que se está presentando es una edad, es conveniente que esté precedido por un mensaje que diga: edad. Y así sucesivamente.

Cuando vayamos a escribir un mensaje en una instrucción de lectura, lo escribiremos encerrado entre comillas.

Con base en lo anterior, si queremos mostrar el nombre y el apellido que se leyeron con las dos instrucciones **input** anteriores, escribimos:

```
print("Nombre ", nombre)
print("Apellido ", apellido)
```

Un programa que resume lo anterior es:

```
nombre = input("Teclee su nombre ")
apellido = input("ahora teclee su apellido ")
print("Nombre ", nombre)
print("Apellido ", apellido)
```

Como resultado de ejecutar esas instrucciones y tecleando el nombre "Pepito" y el apellido "Pinilla", el programa escribe:

```
Nombre Pepito
Apellido Pinilla
```

Si queremos que el nombre y el apellido se escriban en la misma línea, nuestro programa podrá ser:

```
nombre = input("Teclee su nombre ")
apellido = input("ahora teclee su apellido ")
print("Nombre ", nombre, "Apellido ", apellido)
```

Al ejecutar este programa y teclear el mismo nombre y el mismo apellido, la salida que produce es:

Nombre Pepito Apellido Pinilla

Ahora, si hacemos la siguiente variación al programa:

```
nombre = input("Teclee su nombre ")
apellido = input(f"ahora {nombre} teclee su apellido ")
print("Nombre ", nombre, apellido)
```

el efecto en la ejecución es que cuando ejecute la segunda instrucción de lectura el mensaje que aparece es:

ahora **Pepito** teclee su apellido
Nombre Pepito Pinilla

Esto se logra porque la instrucción **input**, después del paréntesis, tiene una **f**, la cual indica que los nombres de variables que aparezcan entre llaves mostrarán su contenido cuando se ejecute la instrucción, y con la instrucción **print** se logró escribir el nombre y el apellido en la misma línea. Otra forma de lograr que escriba el nombre y el apellido en la misma línea es usando las instrucciones **print**, así:

```
print("Nombre", nombre, end="")
print(apellido)
```

De esta forma, con dos instrucciones **print** se logra escribir en una sola línea.

Ahora, si hacemos la siguiente modificación:

```
nombre = input("Teclee su nombre ")
apellido = input(f"ahora {nombre} teclee su apellido ")
edad = input(f"Muy bien {nombre} {apellido} ahora teclee
su edad")
qq = edad + 2
print(qq)
```

al ejecutar este programa, en la primera instrucción se teclea el nombre (Pepito), en la segunda se teclea el apellido (Pinilla) y en la tercera instrucción despliega el mensaje:

```
Muy bien Pepito Pinilla ahora teclee su edad
```

Y digamos que la edad tecleada fue 16. En la siguiente instrucción edad se le suma 2 a la edad. Esta instrucción generará un error ya que la **variable edad** el programa la trabaja como si fuera una **hilera de caracteres**, no un número entero.

En general, todo dato que se lea con la instrucción **input** el programa Python lo almacena como una hilera de caracteres (**tipo string**).

Si se desea que lo almacene como un dato numérico entero, se debe escribir la instrucción de esta manera:

```
edad = int(input("Muy bien {nombre} {apellido} ahora  
teclee su edad"))
```

De este modo se instruye al programa para que el dato que almacene en la variable edad sea numérico entero. Si lo que se desea es que lo almacene como numérico real, en vez de la palabra clave int se debe colocar la palabra clave **float**.