Lógica de programación I

Juan Pablo Restrepo Uribe

Ing. Biomedico

MSc. Automatización y Control Industrial

jprestrepo@correo.iue.edu.co

2023-2

Institución Universitaria de Envigado

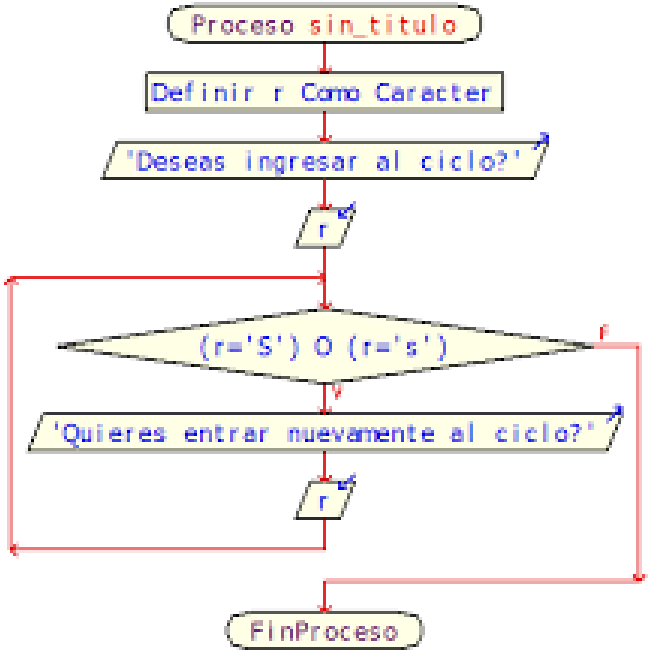
Unidad 2: Estructuras selectivas, condicionales o de decisión. Repetitivas, cíclicas o iterativas

Definición de variables

- Contador
- Acumulador
- Ciclo para, desde o for

Estructuras iterativas

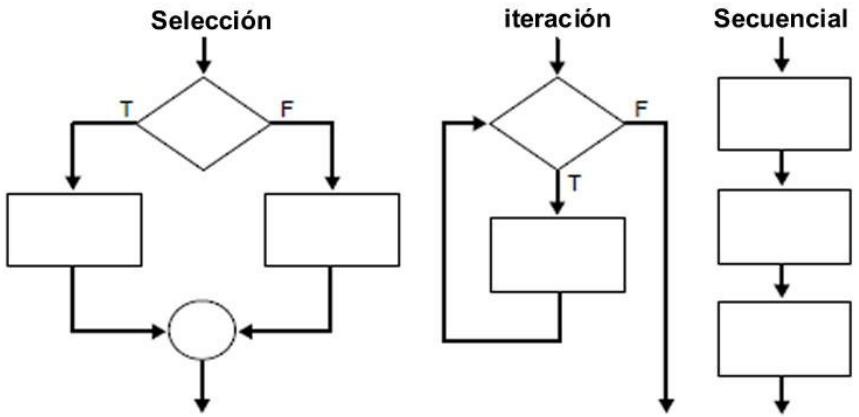
Las estructuras de control iterativas se utilizan para resolver problemas donde sea necesario repetir un número de veces un conjunto de instrucciones. También se les conoce como estructuras repetitivas.



Estructuras iterativas

Un ciclo iterativo es la repetición de operaciones dependiendo de una condición. Las operaciones o instrucciones son las mismas pero los datos que se procesan pueden cambiar en la ejecución del ciclo.

Los ciclos deben terminar después de un cierto número (finito) de repeticiones, y el conjunto de operaciones a repetir se conocen como bucle.



Estructuras iterativas

En ocasiones, sabemos el número de veces que el ciclo se repetirá (para lo cual usamos estructuras FOR), en otras, no sabemos a ciencia cierta cuántas repeticiones se harán (para lo cual usamos cualquier otra de las 3 estructuras restantes). Para poder trabajar con estas estructuras, es necesario comprender antes las variables especiales de ciclos: contadores, acumuladores, centinelas.

Variables contador

Un contador es una variable de tipo entero que, durante el proceso o ejecución de un programa, va aumentando su valor progresivamente. Generalmente un contador va incrementando su valor en 1, pero puede ser un contador de 2 en 2, o de n en n.

CONTADOR = CONTADOR + CONSTANTE

Variables contador

Un contador es una variable de tipo entero que, durante el proceso o ejecución de un programa, va aumentando su valor progresivamente. Generalmente un contador va incrementando su valor en 1, pero puede ser un contador de 2 en 2, o de n en n.

CONTADOR = CONTADOR + CONSTANTE

Variables acumulador

Es una variable numérica (puede ser de tipo entero o real) que durante la ejecución de un programa va sumándose así misma valores contenidos en otras variables.

ACUMULADOR = ACUMULADOR + VARIABLE

Ejemplos

Tenemos que C es el contador, y queremos que incremente de 1 en 1, por lo tanto, la expresión (operación) que se utiliza es C = C + 1, donde 1 es la constante que se sumará al contador cada vez que se ejecute esta operación dentro de un ciclo.

- Si inicialmente tenemos que C = 1 cuando entremos al ciclo y ejecutemos la expresión C = C + 1 tenemos que: C = 1 (valor actual del contador) + 1 (constante) = 2 (nuevo valor del contador)
- Ahora C = 2, si volvemos a entrar al ciclo y repetimos la operación, tenemos que:
- C = 2 (valor actual del contador) + 1 (constante) = 3 (nuevo valor del contador)
- Durante las ejecuciones de la expresión tuvimos que C = 1, luego C = 2, y posteriormente C = 3, como podemos observar C “contó” de 1 en 1.

Ejemplos

Tenemos que A es el acumulador, y queremos que sume las calificaciones que vamos introduciendo de los alumnos de una clase, por lo tanto, la expresión (operación) que utilizamos es A = A + CALIF, donde CALIF es una variable que va almacenando la calificación que vamos introduciendo.

- En este caso A se inicializa a 0, para que la suma sea precisa. A = 0. Cuando entremos al ciclo, primero debemos solicitar CALIF y supongamos que introducimos 7.5, CALIF = 7.5, por lo tanto, al ejecutar la operación tenemos que:
- A = 0 (valor actual del acumulador) + 7.5 (variable CALIF) = 7.5 (nuevo valor del acumulador)
- Si volvemos a entrar al ciclo e introducimos el valor 9.3 para CALIF, y repetimos la operación, tenemos que: A = 7.5 (valor actual del acumulador) + 9.3 (variable CALIF) = 16.8 (nuevo valor del acumulador)
- Si volvemos a entrar al ciclo e introducimos el valor 8.1 para CALIF, y repetimos la operación, tenemos que: A = 16.8 (valor actual del acumulador) + 8.1 (variable CALIF) = 24.9 (nuevo valor del acumulador)

	DIAGRAMA DE FLUJO	EXPLICACIÓN DEL DIAGRAMA
1	 	1. Inicio
2	 	2. Declaración de variables
3	 	3. Inicio del primer for, finaliza después de 11 iteracciones (del 0 al 10, hay 11 repeticiones).
4	 	4. Inicio del segundo for, finaliza después de 4 iteraciones (del 0 al 3 hay 4 repeticiones). En el ejemplo A, hay un error al emplear el mismo contador que en el primer for.
5	 	5. Muestra los números del segundo contador (del 0 al 3). Pero en el ejemplo A, al emplear la misma variable para ambos ciclos, el procedimiento no finaliza, y se "encicla" el mensaje de mostrar los números del 0 al 3.
6	 	6. Fin del segundo ciclo.
7	 	7. Fin del primer ciclo.
8	 	8. Fin del diagrama de flujo.

FOR

Este tipo de ciclos se utiliza cuando se conoce de antemano el número de veces que un conjunto de operaciones deberá ser ejecutado repetitivamente.

Generalmente utilizamos una variable de control como contador, que al llegar a un número predeterminado (condición de ejecución) termina la ejecución del ciclo.

Ordinariamente hacemos uso de las letras *i, j, k, x, y* para representar estas variables de control.

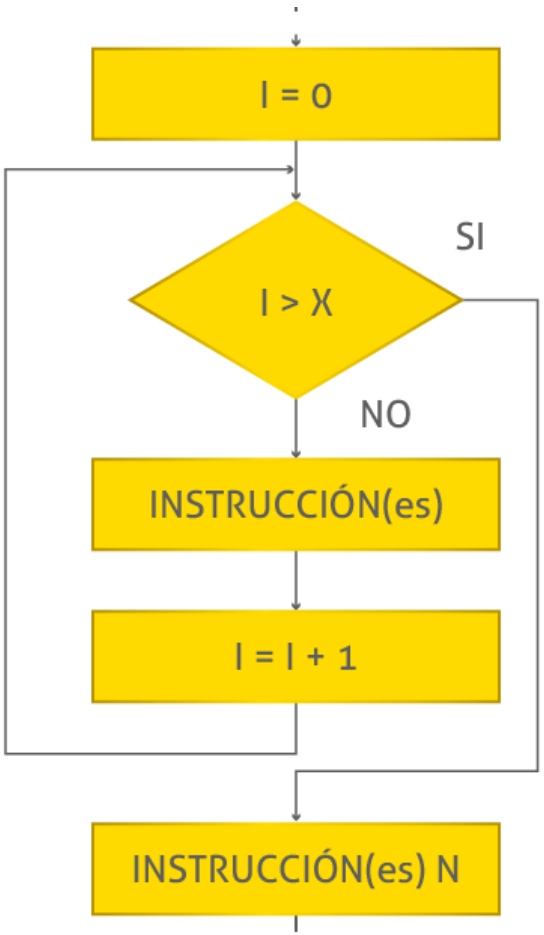
FOR

- La declaración del ciclo *for* en Python comienza con la palabra clave "**for**".
- Lo sigue el nombre de la **variable de control**.
- Lo sigue la palabra reservada "**in**".
- Lo sigue **la lista o colección a recorrer** (sobre los que se desea iterar).
- Luego siguen dos puntos "**:**".
- Finalmente, el **cuerpo** del *for* separado por sangría.

Estructura for

```
for elemento in lista:
  print(elemento)
```

FOR



Donde:

I es la variable de control (contador), inicia en 0, se incrementa en 1 y finaliza en X + 1.

I > X es la condición que debe cumplirse para terminar la ejecución.

INSTRUCCIÓN (es) expresa las acciones que se van a realizar dentro del ciclo.

INSTRUCCIÓN (es) N exspresa las acciones que se van a realizar después de terminar la ejecución del ciclo.

FOR

- Escriba un programa que pida dos números enteros y escriba qué números son pares y cuáles impares desde el primero hasta el segundo.
- Escriba un programa que pida un número entero mayor que cero y que escriba sus divisores.
- Escriba un programa que pregunte cuántos números se van a introducir, pida esos números, y muestre un mensaje cada vez que un número no sea mayor que el primero.
- Escriba un programa que pregunte cuántos números se van a introducir, pida esos números, y muestre un mensaje cada vez que un número no sea mayor que el anterior.
- Escriba un programa que pregunte cuántos números se van a introducir, pida esos números y escriba cuántos negativos ha introducido.

FOR

- Crear un ciclo for que cuente de 0 a 100
- Haz una tabla de multiplicar utilizando el ciclo for
- Imprima los números del 1 a 10 al revés utilizando el ciclo for
- Crear un bucle que cuente todos los números pares hasta el 100
- Cree un bucle que sume los números del 100 al 200
- Dado un número, cuente el número total de dígitos de un número

  Por ejemplo, el número es 75869, por lo que la salida debería ser 5.
- Mostrar series de Fibonacci hasta 10 términos
- Use un bucle para mostrar elementos de una lista dada que estén presentes en posiciones pares

FOR

- Nivel de complejidad 1 - Principiantes:
  - Imprime los números del 1 al 10 utilizando un bucle for.
  - Imprime los números pares del 2 al 20 utilizando un bucle for.
  - Calcula la suma de los números del 1 al 100 utilizando un bucle for.
- Nivel de complejidad 2 - Intermedio:
  - Crea un bucle for que imprima la tabla del 5 (del 5 al 50).
  - Escribe un programa que calcule el factorial de un número ingresado por el usuario utilizando un bucle for.
  - Escribe un bucle for que imprima los caracteres de una cadena de texto uno por uno.

FOR

- Nivel de complejidad 3 - Avanzado:
  - Escribe un programa que encuentre todos los números primos menores que 100 utilizando un bucle for.
  - Crea un bucle for anidado para imprimir un patrón de asteriscos en forma de pirámide.
  - Implementa un juego de adivinanza donde el programa elige un número aleatorio y el usuario debe adivinarlo. El bucle for se utiliza para limitar el número de intentos.
- Nivel de complejidad 4 - Experto:
  - Escribe un programa que calcule e imprima la secuencia de Fibonacci hasta el término n utilizando un bucle for.
  - Crea un programa que simule una carrera entre varios corredores. Utiliza un bucle for para avanzar a los corredores en cada iteración.