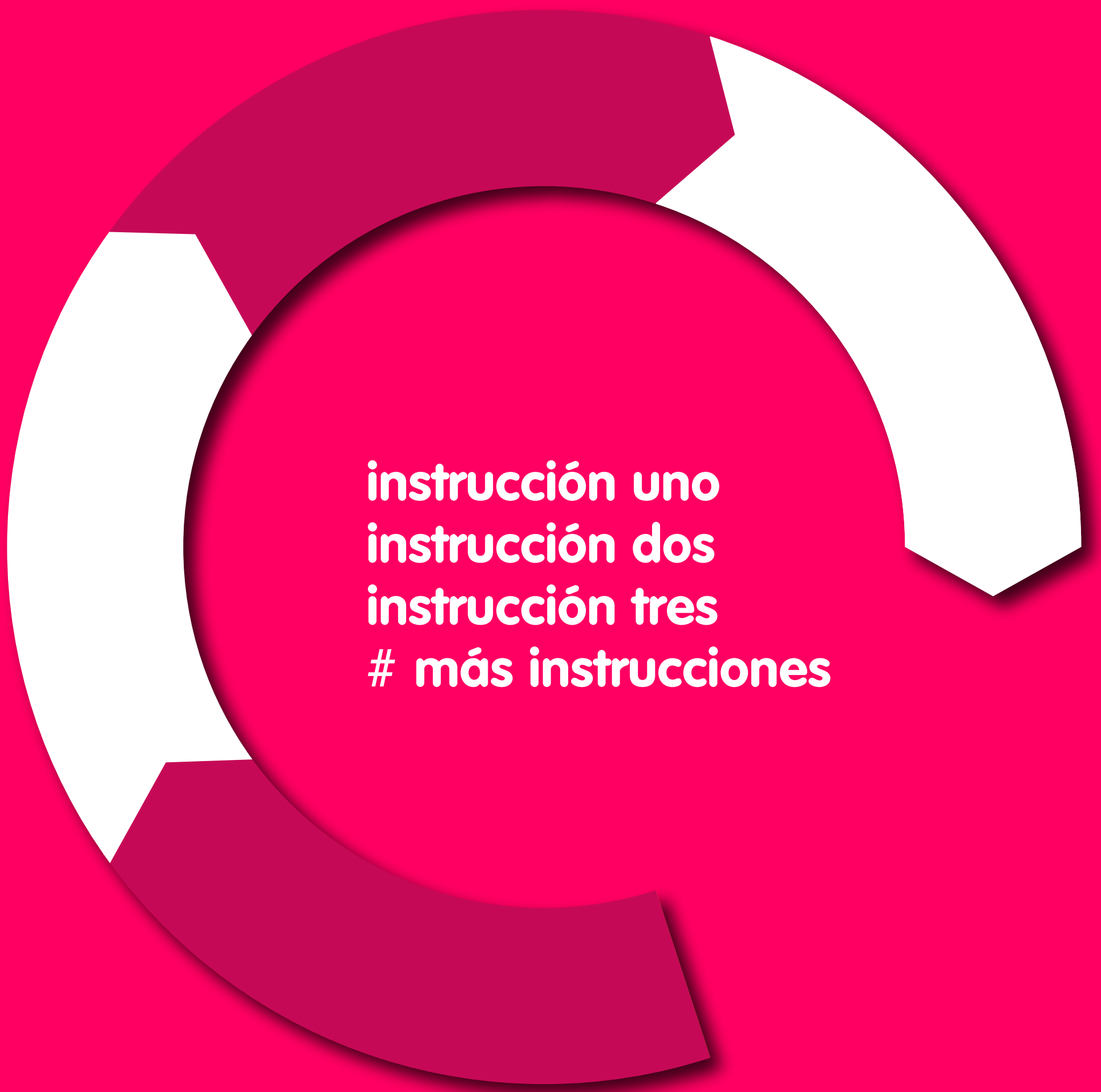
El futuro digital es de todos

MinTIC



Control de flujo: ciclos

Universidad de Caldas

Hola todos

En este video en revista, vamos a ver un tema que nos va a permitir construir programas más interactivos que nos van a ayudar a resolver problemas más complejos, estamos hablando de los ciclos, bucles o loops.

Bienvenidos

Si cuando estás programando descubres que empiezas a escribir código repetido y ya lo has escrito antes y una y otra vez, debemos analizar por qué está pasando dicha situación.

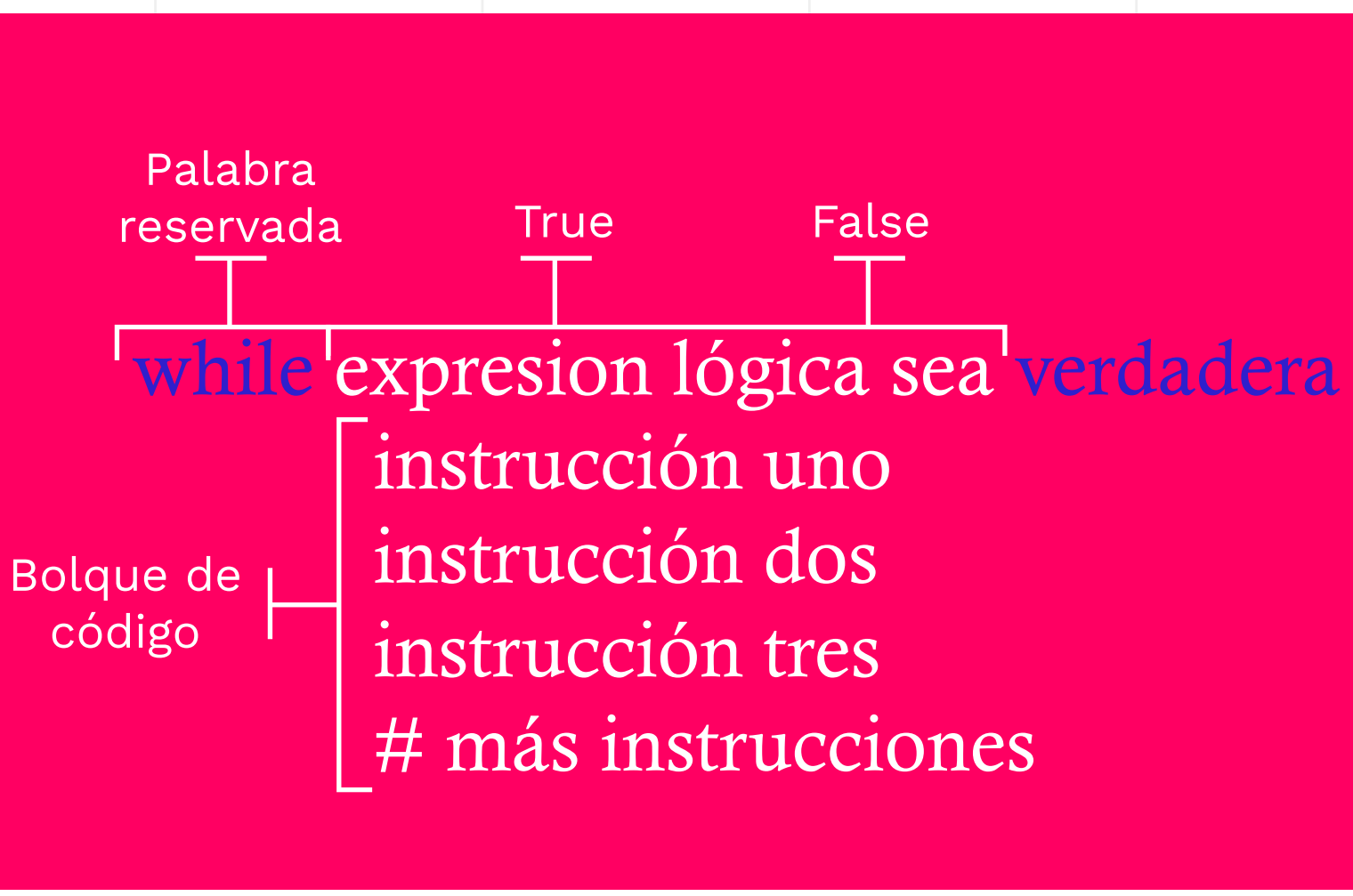
Uno de los principios básicos del programador es **“don’t repeat yourself”**, es decir “no te repitas a ti mismo”. Un ejemplo de esta situación lo vemos cuando copiamos y pegamos código y más aún si es en el mismo programa, lo que realmente puede pasar es que nuestra solución consista en resolver el problema copiando y pegando pero que no obedezca a un proceso de análisis y buenas prácticas de programación.

En efecto al analizar podría llegarse a pensar, que todo ese código lo podemos unificar, es decir si corresponde al mismo código lo podemos unificar y evaluar, sin embargo, ¿cómo hacemos para integrarlo en sitio? si estamos haciendo el mismo código todos 2, 3, 4, 5 veces, quizás lo que necesitemos es repetir este código, un múltople número de veces y no escribirlo tantas veces; de esta manera podremos lograr un programa más legible más entendible y dinámico porque ya no sólo lo repetiré, 4 o 5 veces sino una mayor cantidad de veces.

¿Cómo se comportan los ciclos en nuestro código?, al ejecutar las instrucciones se da el flujo del programa, existe un sitio en el cual se hace necesario repetir esas instrucciones, así que inicia la sentencia “repite”, esta hace que las instrucciones vuelvan a regresar al punto, repite y así sucesivamente.

Esto se conoce como control de flujo, es decir del orden en que se ejecutan las instrucciones y que hay puntos específicos en el código donde van a ver bloques de código, o series de instrucciones que el programa debe repetir, por ejemplo en un programa para el control de mercancía en un almacén, si el programa pide explícitamente que una instrucción se repita 5 veces o 500.000 veces o que se repita por cada producto del almacén, en este caso se hace necesario aplicar el concepto de los ciclos.

Supongamos que tenemos ese grupo de instrucciones que se requieren repetir, la forma correcta de escribirlo normalmente o de definirlo es utilizar la instrucción WHILE, la cual literalmente traduce **mientras que**; luego viene una expresión lógica y básicamente esto puede leerse de la siguiente manera: **mientras, que** una expresión lógica sea verdadera realice la instrucción uno, dos y tres, posteriormente el programa vuelve a verificar que esa expresión lógica sea verdadera, en general el ciclo **mientras que** (WHILE) es el más común y el más utilizado en programación incluso en lenguajes diferente a Python, en C#, en Java.



Entonces ¿qué sucede?
Si tenemos un bloque de código que queremos o que necesitamos repetir, para ello contamos la palabra reservada while, luego tenemos una expresión lógica la cual se evalúa como verdadera o falsa (recordemos que está formada por expresiones relacionales), entonces, mientras que esa expresión lógica sea verdadera, todas las instrucciones que se encuentren en el bloque de código se van a ir repitiendo.

En el siguiente ejemplo, mientras que la distancia objeto sea menor que 5.000 (que sea verdadero), se estarán ejecutando esas instrucciones, a continuación, vuelve, repite y pregunta, (es importante tener mucho cuidado con dichas instrucciones) debe existir algo que ayudan a que esa condición lógica que se encuentra en la parte superior, se vuelva falsa porque de lo contrario nos vamos a enfrentar a uno de los casos típicos de programación que se llama un bucle o un ciclo infinito, lo cual significa que nunca va a terminar, a menos que se interrumpa violentamente la aplicación.

Es importante tener claro que dentro del conjunto de instrucciones, debe existir alguna que esté cambiando el valor de la variable de la condición, entonces cuando esto por fin se vuelva falso termina la ejecución del ciclo.

```
while distancia_objeto < 5000
    instrucción uno
    instrucción dos
    instrucción tres
    # más instrucciones
```

Debe cambiarse algo que afecte el valor de verdad

Las expresiones lógicas son muy dinámicas, recordemos que se pueden tener allí muchas cosas por ejemplo mientras que el estado del sensor esté activo sigo repitiendo, es decir, a veces es posible identificar, cuántos números o cuántas veces o cuantas interacciones es necesario repetir, pero a veces no.

En ocasiones el control del ciclo, es simplemente un estado que va a cambiar, por ejemplo mientras que el voltaje sea menor el límite y la energía, se encuentre estable haga se realiza cierto conjunto de instrucciones.

Existen varios elementos que se pueden ubicar allí y al evaluarse verdaderas van a permitir que la instrucción se repita, desde la estrategia nos debe ser muy claro que se requiere repetir, no es solo tomar un valor arbitrario como contador es menor que 5; esto es necesario saberlo, pero desde la estrategia debimos haber detectado que para solucionar este problema se necesita un ciclo, eso es muy importante.

```
contador = 0
while contador < 5:
        print("el contador va en:", contador)
        contador = contador + 1
print ("terminamos de contar")
```

En el siguiente ejemplo contamos con un contador que empieza en cero, estamos diciendo que mientras esa variable contador sea menor que 5 imprima “el contador va en..:” e imprime la variable y luego lo incrementa en 1, es muy importante, aquí estamos dando la instrucción que de alguna manera va a ayudar a que esta instrucción que está arriba se vuelva falsa, dicho control es muy importante en el ciclo, de modo que se cuente con algo que garantice que en algún momento, éste va a terminar. Por ejemplo si ejecutamos este programa va a salir así :

Console

```
El contador va en : 0
El contador va en : 1
El contador va en : 2
El contador va en : 3
El contador va en : 4
terminamos de contar
>
```

Y en algún momento dará 5 y como 5 no es menor que 5 terminará de contar.

Qué elementos son importantes en el manejo del ciclo while

1: la inicialización
2: la expresión lógica
3: cambio al valor de verdad

```
contador = 0          ← Inicialización
while contador < 5:   ← Expresión lógica
        print("el contador va en:", contador)
        contador = contador + 1   ← Cambio al valor de verdad
print ("terminamos de contar")
```

Aquí tenemos otro gran ciclo que se maneja en Python, en ciclo FOR, Python va a tener un significado diferente al que por experiencia se haya conocido, en este tipo de ciclo se utiliza la instrucción **for** la cual representa otro tipo de ciclo, algo que nos va a ayudar a repetir pero lo va a repetir desde el punto de vista for elemento In colección, es decir, por cada elemento en la colección va a hacer algo.

```
for elemento in colección
    instrucción uno
    instrucción dos
    instrucción tres
    # más instrucciones
```

Ejemplo: por cada producto que se tenga en un inventario, se realiza un conjunto de instrucciones, se actualizan precios, e inventarios.

```
for producto in inventario
    instrucción uno
    instrucción dos
    instrucción tres
    # más instrucciones
```

También podríamos tener otras cosas por ejemplo por cada número en un rango, esto lo vamos a utilizar más adelante, lo veremos en laboratorios o en talleres y nos va a servir un poco como para contar, como por ejemplo si se dice que para cada número en un rango del 1 al 10 entonces el programa va a mostrar los números del 1 al 10, (se estudiará en laboratorios)..

En resumen, el control de flujo, es la estructura en la cual se ejecutan las instrucciones, ya habíamos visto el IF, (instrucción condicional), pero acá vimos uno diferente que son los ciclos, bucles y la iteración, de ellas la instrucción más empleada es el While (mientras que), donde analizamos los elementos, condiciones iniciales, expresión lógica y algo dentro del cuerpo del ciclo que debe trabajar para que la condición se vuelva falsa y el ciclo pueda terminar, ya que los siglos que no terminan se llaman ciclos infinitos, después viene otro ciclo el cual profundizaremos más adelante, llamado el ciclo FOR (para), que en Python se usa normalmente para realizar una acción en un conjunto de elementos.

Muchas gracias por su atención hasta un próximo video en revista


Mision
TIC2022

Universidad de Caldas