

Introducción a la Programación

Estructuras de control
Repetitivas

Objetivos

- Definiciones
- Estructura Para (for)
- Acumuladores
- Actividades

Ciclos (Para/for)

Su función consiste en ejecutar un número determinado de veces una secuencia de instrucciones.

Para ello utilizaremos la función llamada **range()**

Ejemplo:

`range(10)` es una lista de diez valores
del 0 al 9

```
for i in range (10):  
    print ("i=", i)
```

Devuelve:

i= 0
i= 1
i= 2
i= 3
i= 4
i= 5
i= 6
i= 7
i= 8
i= 9

Ejecuta
10
iteraciones
desde 0
hasta 9

Ciclos (Para/for)

El problema de Arnaldito (antes:)

```
i=1
while i<=500
    print ("No debo tirar avioncitos")
    i=i+1
```

Ahora lo resolvemos con for

```
for i in range (500):
    print ("No debo tirar avioncitos")
```

Ventaja!!

No tener que incrementar la variable de control en cada iteración!!

Ciclos (Para/for)

Ciclo con while (antes:)

```
i=1  
while i<=500  
    print ("No debo tirar avioncitos")  
    i=i+1
```

Ciclo con for (ahora)

```
for i in range (500):  
    print ("No debo tirar avioncitos")
```

i
es la
Variable
de
control

En ambos casos el ciclo ejecuta 500 iteraciones

Ciclos (Para/for)

Ciclo con while (antes:)

```
i=1  
while i<=500  
    print ("No debo tirar avioncitos")  
    i=i+1
```

Variable de control:

se inicializa

y

se incrementa a mano

Ciclo con for (ahora)

```
for i in range (500):  
    print ("No debo tirar avioncitos")
```

Variable de control:

La administra Python

En ambos casos el ciclo ejecuta 500 iteraciones

Ciclos (Para/for)

Ciclo con while (antes:)

```
i=1
while i<=500
    print ("No debo tirar avioncitos")
    i=i+1
```

Variable de control:

i=1
i=2
...
i=500 >>>> *Último valor que toma i =500, ejecuta el ciclo y lo finaliza porque el sgte valor 501 no es <=500*

Ciclo con for (ahora)

```
for i in range (500):
    print ("No debo tirar avioncitos")
```

Variable de control:

i va desde 0 hasta 499
>>> *Último valor de i=499*
Con i=500 no entra al ciclo for

En el ciclo con **for** no se debe alterar el valor de la variable de control. La administra Python

Ejemplos

```
for x in range(3):  
    print("x= ", x)
```

```
for x in range(1, 10):  
    print("x= ", x)
```

```
for x in range(3,50,3):  
    print("x= ", x)
```

```
for x in range(55,2,-2):  
    print("x= ", x)
```

```
for x in range(50,8,1):  
    print("x= ", x)
```

En
todos los
casos se
usó la
variable
x
como
variable
de
control

Acumulador

En el siguiente ejemplo se intenta realizar la suma de los primeros n números naturales.

Si n es 10 los primeros 10 números naturales son:

1 2 3 4 5 6 7 8 9 10

Y la respuesta debería ser 55 ya que:

$$1+2+3+4+5+6+7+8+9+10 = 55$$

Para resolverlo voy acumulando los resultados parciales:

$$\begin{array}{rcl}
 1+2 & & \\
 \textcolor{red}{3} & + & 3 \\
 \textcolor{red}{6} & + & 4 \\
 \textcolor{red}{10} & + & 5 \dots\dots\dots \\
 & & \dots\dots\dots\textcolor{red}{45} + 10 = 55
 \end{array}$$

El resultado lo tuve que ir guardando (acumulando) en algún lado para no perderlo

Acumulador

```

1  n=int(input("Indique hasta que numero quiere sumar"))
2  suma=0
3  for i in range(1,n+1,1):
4      suma=suma+i
5  print("La suma de los primeros",n, "naturales es:",suma)

```

Ciclo de n iteraciones

Analicemos el código anterior:

En la línea 2 (**suma=0**) se le asigna a la variable “suma” el valor 0.

En la línea 3 se define un *ciclo for*, para la variable de control **i**, que toma valores desde 1 hasta $n+1$, en pasos de a 1, y cuando la variable **i** llegue a $(n+1)$, NO entra en el ciclo for, lo da por finalizado y continúa en la línea 5.

Acumulador

```

1 n=int(input("Indique hasta que numero quiere sumar"))
2 suma=0
3 for i in range(1,n+1,1):
4     suma=suma+i
5 print("La suma de los primeros",n, "naturales es:",suma)

```

Continuamos analizando el código anterior:

En la línea 4 (**suma=suma+i**) el = es una asignación, no es el igual de matemática. Sino esa afirmación sería verdadera solamente si $i=0$. Lo que hace el programa es asignar a la variable **suma** el resultado de calcular lo que tenía suma antes de esta instrucción y sumarle el contenido de la variable **i**

La variable suma está **acumulando** las sumas parciales en cada iteración y de allí su nombre **acumulador**.

Acumulador

```

1 n=int(input("Indique hasta que numero quiere sumar"))
2 suma=0
3 for i in range(1,n+1,1):
4     suma=suma+i
5 print("La suma de los primeros",n, "naturales es:",suma)

```

Continuamos analizando el código anterior:

Un cuidado que se debe tener al utilizar **acumuladores** es **inicializarlos** como en la línea 2. (**suma= 0**). Sino no puede hacer **suma+i** ya que no conoce el valor de suma.

Las variables que intervienen en las expresiones a la derecha de un **=** deben contener valores ya generados en instrucciones previas.

Actividades

1. Realizar un programa que imprima la tabla del número que el usuario desee (desde el 1 al 10). Hacer ahora una versión con ciclos for.
2. Escribir un programa que le pregunte al usuario un número entero positivo N , y que calcule su factorial ($N!$)
(ej: $5! = 5 * 4 * 3 * 2 * 1$)
3. Escribir un programa que le pida al usuario dos números. Devuelve el primer número elevado al segundo. $()$.
4. Escribir un programa que le pida al usuario números (la cantidad que el usuario desee) y calcule el promedio.
5. Escribir un programa que le pregunte al usuario un número entero positivo n y que calcule la parte entera de $\sqrt[n]{n}$.
6. (solo para ingeniosos) Escribir un programa que le pregunte al usuario un número entero positivo N , y que muestre por pantalla la cantidad de cifras que tiene. (Ej: 1492 tiene 4 cifras)