**CÓMO CONTAR OPERACIONES**

1. **Instrucciones simples**

Inmediatamente después de la instrucción, aumenta en 1 la variable num\_op. Es bueno que comentes cada incremento de la variable num\_op para que tu mism@ lo recuerdes después y para que nosotros podamos chequear si tu número de operaciones no coincide con los nuestros. Por ejemplo:

|  |
| --- |
| cambio=true  num\_op=num\_op+1 // incrementa num\_op por asignación a variable cambio |

Si tienes varias instrucciones seguidas (por ejemplo, al hacer swap), puedes hacer una sola suma. Con mayor razón, es bueno que agregues el comentario. Por ejemplo:

|  |
| --- |
| tmp=A[i]  A[i]=A[i+1]  A[i+1]=tmp  num\_op=num\_op+3 // incrementa num\_op en 3 por swap |

En general, no importa cómo hagas el swap, aumenta num\_op en 3 operaciones.

1. **Condicionales**

Un condicional tiene dos partes: la evaluación de la condición (esa parte se ejecuta siempre) y las instrucciones a ejecutar si la condición es verdadera (esa parte sólo se ejecuta si la condición es verdadera).

Por ejemplo:

|  |
| --- |
| num\_op=num\_op+1 //aumenta 1 por la evaluación de la condición del if  **if** (A[i]>A[i+1]) **then**  x=2  y=x+3  num\_op=num\_op+2 //aumenta 2 por las instrucciones dentro del if |

1. **Ciclos for**

Los ciclos for tienen 3 partes (en C se ve muy claro, en Python está más escondido por la manera en que se escribe el ciclo):

* la inicialización de la variable de control
* la evaluación de la condición para continuar con el ciclo
* la actualización del valor de la variable de control

Por ejemplo, la instrucción en C:

for (i=0; i<N, i++)

printf(“hola”)

* la parte “i=0” corresponde a la inicialización de la variable de control i
* la parte “i<N” corresponde a la evaluación de la condición que permite seguir con el ciclo
* la parte “i++” corresponde a la parte de la actualización de la variable de control i

Si el ciclo no se ejecuta (por ejemplo, cuando N es igual a 0), al menos se realizan dos operaciones:

* Se inicializa la variable de control i en 0
* Se evalúa la condición

Al evaluar la condición, se verifica que no se cumple y no se ejecuta nada más del ciclo.

En cambio, si el ciclo se ejecuta N veces, entonces las operaciones que se realizan son:

Inicializa la variable i en 0 (1 operación)

Evalúa la condición (i<N) N+1 veces (N+1 operaciones)

Incrementa i N veces (N operaciones)

Ejecuta printf N veces (N operaciones)

Por lo tanto, en un ciclo se ejecutan siempre al menos 2 operaciones (inicializar variable de contro y evaluar condición) y luego, dentro del ciclo se ejecutan 2 operaciones propias del ciclo (evaluar condición y actualizar variable) más las operaciones de las instrucciones dentro del ciclo.

|  |
| --- |
| num\_op=num\_op+2 //inicialización y evaluación inicial de condición  for (i=0; i<N, i++)  num\_op=num\_op+2 //evaluación de condición y actualización i  printf(“hola”)  num\_op=num\_op+1 //por la instrucción print |

1. **Ciclos while**

Con los ciclos while es más fácil visualizar el número de operaciones. Un ciclo while tiene siempre la siguiente estructura (igual a la del ciclo for):

* Inicialización variable de control
* Evaluación de condición
* Actualización de la variable de control

Al igual que el ciclo for, si la condición del ciclo while no se cumple desde el principio, se ejecutan al menos dos operaciones (inicialización y evaluación de condición).

A continuación se muestra un ejemplo del conteo de operaciones de un ciclo while:

|  |
| --- |
| cambio=true  num\_op=num\_op+1 //inicialización variable de control  num\_op=num\_op+1 //evaluación de condición  while(cambio)  A[i]=A[i]+1  num\_op=num\_op+1 //asignación A[i]  num\_op=num\_op+1 //evaluación condición if  if(A[i]%2==0)  cambio=false  num\_op=num\_op+1 //actualización variable de control |

En un caso más simple:

|  |
| --- |
| i=1  num\_op=num\_op+2 //inicialización variable y evaluación condicion  while(i<10)  print(i)  i=i+1  num\_op=num\_op+2 //actualización variable y print |