Taller de Física Computacional

Condicionales (control de flujo)

Cristián G. Sánchez y Carlos J. Ruestes

2021

Condicionales

Condicionales

Los condicionales son ubicuos en los lenguajes de programación y permiten dividir la ejecución en *ramas* distintas de acuerdo al resultado de la evaluación de una o más expresiones Booleanas. En Python existen tanto **expresiones** como como **declaraciones** condicionales.

Declaraciones y expresiones

Nota

El Manual de Referencia de Python distingue entre distintos tipos de **expresiones** y **declaraciones**. Sin entrar en definiciones estrictas (porque la categorización *no es estricta*) podemos decir que:

- Las **expresiones** son combinaciones de elementos sintácticos como identificadores, literales, llamados a funciones, operadores, etc. que poseen al menos un valor.
- Las **declaraciones** se pueden pensar como la mínima unidad imperativa, las declaraciones hacen algo. Las asignaciones, por ejemplo, son un tipo de declaración.

La expresión condicional

La expresión condicional

La mínima unidad de código de ejecución condicional en python es la siguiente:

```
a = X if Z else Y
```

donde Z es una variable o expresión Booleana y X e Y son dos variables, expresiones o literales. Si Z es `True` la variable a tomará el valor X, si es `False` tomará el valor Y.

Declaraciones condicionales

La declaración condicional más simple

La declaración condicional más simple en python es la siguiente:

```
if X: (declaración)
```

donde `(declaración)` es una única línea de código que es ejecutada si la expresión X es `True`.

Declaraciones condicionales

La declaración condicional

La declaración condicional que sigue en complejidad es la siguiente:

```
if X:
    (declaración 1)
    (declaración 2)
    # y sigue...
```

donde la serie de declaraciones indentadas son ejecutadas si la expresión X es `True`. Como vimos en el caso de la definición de funciones la indentación es una parte fundamental de la estructura.

Declaraciones condicionales

La declaración condicional

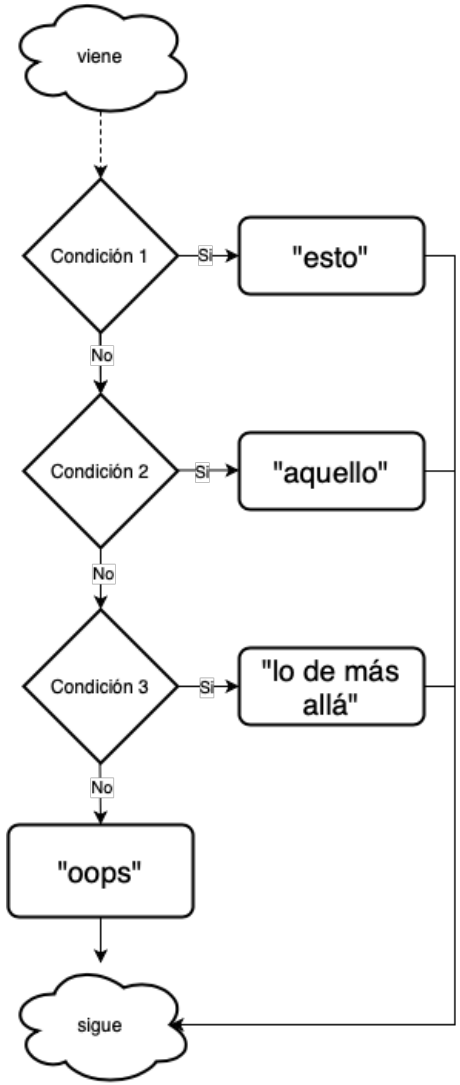
La declaración condicional que sigue en complejidad es la siguiente:

```
if X:
    (declaración 1)
    (declaración 2)
    # y sigue...
else:
    (declaración 1)
    (declaración 2)
    # y sigue...
```

donde la serie de declaraciones indentadas luego de la línea que contiene al `if` son ejecutadas si X es `True` y si X es `False` se ejecutan las declaraciones indentadas luego de la línea que contiene al `else`.

La declaración condicional más completa

```
if (condición 1):
    pasar.cosas("esto")
elif (condición 2):
    pasar.cosas("aquello")
elif (condición 3):
    pasar.cosas("lo de más allá")
else:
    pasar.cosas("oops")
```



Condicionales y eficiencia

Los condicionales son de extrema utilidad en programación. Sin embargo al causar lo que se denomina *branching* (ramificación), tienen una penalidad en términos de eficiencia. El *branching* hace que el procesador deje de ejecutar una secuencia de instrucciones y deba saltar una secuencia diferente de instrucciones.

Condicioneales y eficiencia

Mucha de la ingeniería de eficiencia en los procesadores modernos se basa en la predictibilidad de un programa. El *branching* rompe la predictibilidad y por lo tanto disminuye la eficiencia. Los condicionales deben ser evitados particularmente dentro de los bucles, en donde se ejecutan, potencialmente, muchísimas veces. De ser posible el código debe ser estructurado para que las declaraciones condicionales se ejecuten fuera de los bucles. Las estructuras condicionales dentro del código paralelo imponen también penalidades en términos de eficiencia.

Síntesis y recursos:

- Condicionales en Wikipedia
- Manual de Referencia de Python