Las sentencias condicionales hacen que la programación tenga gran importancia. Esta es la capacidad de comparar el valor de una variable contra otra variable o una constante y ejecutar un bloque de instrucciones si se cumple una condición o de otra si no se cumple.

En este componente formativo se explicará el uso de las sentencias condicionales simples y las sentencias condicionales anidadas, utilizando expresiones booleanas simples y compuestas.

Existen operadores booleanos (también llamados lógicos) y operadores de comparación.

En Python, los tres operadores booleanos son: **“and” (y), “or “(o), “not”(no)**.

Find More Candidates with Boolean Search, 2015

Para Buttu(2016) una sentencia u operación lógica puede ser evaluada Verdadera o Falsa de la siguiente manera:

El operador and genera como resultado el valor True, solamente cuando son ciertos sus dos operandos.

El operador or da como resultado el valor True, si cualquiera de sus operandos es True, y False sólo cuando ambos operandos son False.

El operador not es unario, y proporciona el valor True si su operando es False y viceversa.

Salazar(2019) menciona que es importante conocer las prioridades existentes al usar los operadores lógicos: en primer lugar se tiene el operador de negación **(not)**, luego el de conjunción ***(and)*** y por último el de disyunción ***(or)***.

En general, los elementos nulos o vacíos se consideran False y el resto se consideran ***True***.

Si se desea verificar si un elemento es ***True*** o ***False***, se puede convertir a su valor booleano mediante la función **bool()**.

```
>>> bool(0)

False
```

```
>>> bool(0.0)

False

>>> bool("")

False

>>> bool(5)

True

>>> bool(3.2)

True

>>> bool("SENA")

True

>>>
```

En Python se usan los siguientes operadores de comparación:

```
Ejemplo:

>>> 2==3

False

>>> 2>1

True

>>> 2>8

False

>>> 3>=4

False

>>> 8>=5

True
```

```
>>> 1<=2< /p>

True

>>> 5!=6

True
```

Una sentencia condicional consta básicamente de los siguientes componentes:

Una prueba que evalúa verdadero o falso.

Un bloque de código que se ejecuta si la prueba es verdadera.

Un bloque opcional de código si la prueba es falsa.

1. Condicionales simples

Cuando se evalúa una condición, existen dos respuestas posibles: verdadero (True) o falso (False). Si la condición es verdadera, el flujo del programa continúa en el bloque de instrucciones de la respuesta verdadera, de lo contrario, el programa continúa en el bloque de instrucciones de la respuesta falsa.

Una característica valiosa en la sintaxis de Python es el uso de la indentación, a diferencia de otros lenguajes como C que utilizan llaves { }. Salazar (2019) lo confirma en su libro.

La indentación consiste en dejar una sangría de cinco (5) espacios para señalar un bloque completo de instrucciones que se deben ejecutar, en caso que la sentencia condicional sea verdadera o falsa.

La indentación es un rasgo muy particular del código Python, y permite una lectura más agradable del programa y una fácil identificación de las distintas partes del programa.

En caso de un error de indentación, aparece el siguiente mensaje:

IndentationError: expected an indented block

Python usa la siguiente sintaxis:

```
Si (condición):

hacer esto solo para ‘Verdadero’

bloque de instrucciones

de otro modo:

hacer esto solo para ‘Falso’

bloque de instrucciones

Siguiente instrucción después de la condicional

bloque de instrucciones
```

Programas ramificados:

En las instrucciones secuenciales, las sentencias se ejecutan en el orden en el que aparecen, en cambio, los programas ramificados permiten ejecutar las sentencias sin importar el orden, basándose en la toma de decisiones.

Según Marzal (2014), las sentencias condicionales se aplican dentro de este tipo de programación; si una sentencia condicional ha sido ejecutada, la ejecución del programa continúa en la siguiente línea de código condicional.

1.1 Uso de IF

La sentencia condicional ***if*** se usa para tomar una decisión. Esta sentencia realiza una operación lógica que debe dar como resultado True o False, y ejecuta el bloque de código siguiente, siempre y cuando el resultado sea verdadero.

La sintaxis es la siguiente:

```
if (condición):

hacer esto solo para ‘Verdadero’

bloque de instrucciones

Siguiente instrucción después de la condicional
```

```
bloque de instrucciones

Ejemplo:

numero= - 2

if numero < 0:

print ("El número ingresado es negativo: ", numero, " \n")

print ("El número será cambiado a cero. \n")

numero = 0

print ("El número ingresado ahora es ", numero, " \n")

El número ingresado es negativo : -2

El número será cambiado a cero.

El número ingresado ahora es 0.
```

1.1 Uso de IF-ELSE

Una segunda forma de la sentencia condicional ***if*** es la ejecución con dos posibilidades. La condición, dependiendo si es verdadera o falsa, determina cuál de los dos bloques de instrucciones se ejecuta. El programa continúa después de la última instrucción con indentación.

La sintaxis es la siguiente:

```
if (condición):

hacer esto solo para ‘Verdadero’

bloque de instrucciones

else:

hacer esto solo para ‘Falso’

bloque de instrucciones

Siguiente instrucción después de la condicional
```

```
bloque de instrucciones

Ejemplo:

x=7

if x % 2 == 0:

print (x, "el número secreto es par")

else:

print (x, "el número secreto es impar")

print ("-------------------------")

print ("hemos terminado la verificación para este número")
```

La operación matemática (x % 2) calcula el valor del residuo de dividir un número por 2. El anterior código valida el valor del residuo, si el residuo es cero significa que se trata de un número par, de lo contrario, significa que es un número impar.

Este programa dará como resultado:

```
7 el número secreto es impar

-------------------------

hemos terminado la verificación para este número.
```

El uso de else es opcional. Si no se coloca, nada sucederá cuando la condición sea ‘Falsa’.

Si por algún motivo no se quisiera ejecutar instrucción alguna en uno de los bloques, se debe usar la instrucción pass (esta orden significa que no tiene que hacer nada).

```
Ejemplo:

edad = int(input("¿Cuántos años tiene? "))

if edad < 99:

pass

else:

print("¡No creo que usted tenga esa edad!")
```

```
print("Usted dice que tiene {edad} años.")

¿Cuántos años tiene? 130

¡No creo que usted tenga esa edad!

Usted dice que tiene 130 años.
```