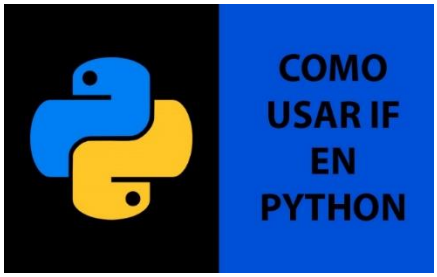
ESCUELA ESPECIALIZADA EN INGENIERIA
REGIONAL SANTA ANA
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA Y ELECTRONICA
CREACIÓN DE PROGRAMAS COMPUTACIONALES (CPC)

Guía Práctica N.º 6: Estructuras Condicionales (If, If-Else) en Python

Resultado De Aprendizaje:

- Identificar los distintos tipos de Estructuras Condicionales en la manipulación de datos en modo consola en Python.
- Resolver ejercicios de la vida practica utilizando estructuras condicionales en Python modo consola.



Comparaciones

Para comparar se usan los siguientes operadores básicos:

Operador	Función
>	Para comparar si un valor es mayor que otro.
>=	Para comparar si un valor es mayor o igual que otro.
<	Para comparar si un valor es menor que otro.
<=	Para comparar si un valor es menor o igual que otro.
==	Para comparar si un valor es igual a otro.
!=	Para comparar si un valor es distinto a otro.
is	Para comparar si un objeto es igual a otro.
is not	Para comparar si un objecto es distinto de otro.

Sentencia IF

La sentencia if en Python se utiliza para evaluar una condición y ejecutar un bloque de código si la condición es verdadera.

Ejemplo:

```
1   edad = 18
2   if edad >= 18:
3       print("Es mayor de edad")
```

En este caso, se compara si la edad es mayor o igual a 18. Si es así, se imprime el mensaje "Es mayor de edad".

Es importante recordar los dos puntos : al final de la sentencia if y la indentación del bloque de código que se ejecutará si la condición es verdadera.

Sentencia IF-ELSE

La sentencia **else** se utiliza junto con **if** para ejecutar un bloque de código si la condición evaluada en la sentencia **if** es falsa.

Ejemplo:

```
edad = 18
if edad >= 18:
    print("Es mayor de edad")
else:
    print("No es mayor de edad")
```

En este caso, si la edad es menor a 18, se imprimirá el mensaje "No es mayor de edad". Al igual que con if, ***es importante recordar los dos puntos : al final de la sentencia else y la indentación del bloque de código que se ejecutará si la condición es falsa.***

En Python solo existe "**==**" así que siempre compara **valor y tipo**. Por lo tanto, operaciones de números con strings siempre devolverán false, aunque tengan el mismo valor, ya que tienen distinto tipo.

Ejemplo:

```
if 5 == '5':
    print("Aquí no va a entrar nunca porque sus tipos de dato son distintos")
else:
    print("Siempre entrará aqui")
```

Sentencias con strings

Para comparar **strings**, se puede usar "**==**" para saber si son iguales o "**!=**" para comprobar si son distintos. Sin embargo, si se quiere comprobar si una variable es una cadena vacía, el manual de estilos de Python recomienda utilizar **if not** .

Ejemplo:

```
a = ''
if not a:
    print("Esta es una cadena vacía")
```

Sentencia elif

La sentencia elif (una contracción de "else if") se utiliza para verificar múltiples condiciones después de una sentencia if inicial.

Ejemplo:

```
comida_preferida = "Tacos"
if comida_preferida == "Tacos":
    print("Te gustan los Tacos")
elif comida_preferida == "Pizza":
    print("Te gusta la Pizza")
elif comida_preferida == "Hamburguesa":
    print("Te gustan las hamburguesas")
else:
    print("Te gusta otra comida")
```

En este caso, se pueden tener múltiples sentencias **elif** después de una sentencia **if**, y no es necesario tener una sentencia **else** al final, aunque se recomienda.

Encadenar varias sentencias

También puede surgir el caso en el que se quiera realizar distintas acciones según el valor de una variable. Para solucionar esto se puede usar la sentencia **elif** todas las veces que necesitemos después de una sentencia **if**. De esta forma el código irá comprobando todas las sentencias hasta que encuentre la que cumpla la condición. Opcionalmente se puede añadir un **else** al final para realizar una acción si no se cumplen ninguna de las expresiones de cada sentencia.

Ejemplo:

```
command = input("Ingrese una letra:")

if command == 'a':
    print('Acciones del comando a')
elif command == 'b':
    print('Acciones del comando b')
elif command == 'c':
    print('Acciones del comando c')
else:
    print('Comando inválido')
```

Como utilizar más de una condición en una misma sentencia

Cuando se necesita usar más de una condición se puede utilizar los operadores **and** y **or**. El significado literal del operador **and** sería "**y**", se utilizaría en el caso de que se quiera que se cumplan todas las condiciones añadidas. En el caso del operador **or** significa "**o**", se utilizaría cuando solo se necesite que se cumpla una de las condiciones.

Ejemplo de **and**:

```
edad = 25
if edad >=18 and edad <= 64:
    print('Eres adulto pero no estás jubilado')
```

En este caso al cumplirse ambas condiciones, entraría dentro del bloque y realizaría el print.

Ejemplo de **or**:

```
edad = 65
if edad < 18 or edad > 64:
    print('Tienes descuento')
```

En este caso al cumplirse una de las dos condiciones, entra dentro del bloque y ejecuta el print.

Por último, también se tiene la posibilidad de combinarlos:

```
edad = 18
tiene_pase_vip = True
tiene_ticket = False
if edad > 17 and (tiene_pase_vip or tiene_ticket):
    print('Puede pasar')
```

Al cumplirse la condición de la edad y una de las dos dentro del paréntesis, se ejecuta el bloque de código.

Sentencias con booleanos

En el caso de que se quiera comprobar si una variable es verdadera, se puede añadir simplemente la variable o podemos utilizar el operador **is** más el valor **True**:

```
is_valid = True

if is_valid:
    print('Es válido')

if is_valid is True:
    print('Es válido')
```

En el caso de que se quiera comprobar si un valor el falso, se puede utilizar el operador **not** o **is** más el valor **False**:

```
is_valid = False

if is_valid is False:
    print('No es válido')

if not is_valid:
    print('No es válido')
```

Sentencias con listas

Para comprobar si un valor está dentro de una lista, se puede utilizar el operador **in** de la siguiente forma:

```
mi_lista = ['manzana', 'pera', 'naranja']

fruta = 'naranja'

if fruta in mi_lista:
    print('Tenemos naranjas')
```

Sentencias con diccionarios

Para comprobar si existe una clave en un diccionario, también se puede utilizar el operador **in** como se muestra en el siguiente ejemplo:

```
usuario = 'usuario'

datos = {
    'usuario': 'Alber',
    'fecha': '2021-04-27'
}
if usuario in datos:
    print('Tenemos la clave usuario')
```

En el caso de querer buscar un valor dentro de un diccionario, se puede usar el método **values()** accesible en todas las variables del tipo diccionario y que devuelve una lista con todos los valores. De esta forma se puede usar el operador **in** para comprobar si existe el valor o no:

```
datos = {
    'usuario': 'Alber',
    'fecha': '2021-04-27'
}
if 'Alber' in datos.values():
    print('Tenemos el usuario Alber')
```

Sentencia IF ternaria

Por último, se pueden crear operadores ternarios. Estos son básicamente operaciones de una sola línea. De esta forma se puede ahorrar líneas de código cuando se tengan condiciones muy sencillas.

Estas sentencias tienen la siguiente estructura: ***'valor if' if expresion else 'valor else'***

Ejemplo:

```
edad = 24
persona = 'Es adulto' if edad > 17 else 'Es menor de edad'
print(persona)
```

Si cumple la condición del **if** guardará el valor "**Es adulto**", en el caso contrario, guardará el valor en el **else**.

GUIA DE EJERCICIOS.

1. **Verificación de mayoría de edad**: Escribe un programa que pida al usuario su edad e imprima si es mayor de edad o no. Si la edad es 18 o más, el programa debe imprimir "Eres mayor de edad". De lo contrario, debe imprimir "No eres mayor de edad".

2. **Comparación de números**: Escribe un programa que pida al usuario dos números e imprima cuál es el mayor. Si el primer número es mayor que el segundo, el programa debe imprimir "El primer número es mayor". Si el segundo número es mayor, debe imprimir "El segundo número es mayor". Si los números son iguales, debe imprimir "Los números son iguales".

3. **Determinación del cuadrante de un punto**: Escribe un programa que pida al usuario las coordenadas de un punto en el plano (x, y) e imprima en qué cuadrante se encuentra el punto. Recuerda que el primer cuadrante es donde $x > 0$ e $y > 0$, el segundo cuadrante es donde $x < 0$ e $y > 0$, el tercer cuadrante es donde $x < 0$ e $y < 0$, y el cuarto cuadrante es donde $x > 0$ e $y < 0$.

4. **Verificación de divisibilidad**: Escribe un programa que pida al usuario dos números e imprima si el primer número es divisible por el segundo. Si el primer número es divisible por el segundo, el programa debe imprimir "El primer número es divisible por el segundo". De lo contrario, debe imprimir "El primer número no es divisible por el segundo".

5. **Determinación de la paridad de un número**: Escribe un programa que pida al usuario un número e imprima si el número es par o impar. Si el número es par, el programa debe imprimir "El número es par". Si el número es impar, debe imprimir "El número es impar".

6. **Determinación del tipo de triángulo**: Escribe un programa que pida al usuario las longitudes de los tres lados de un triángulo e imprima si el triángulo es equilátero, isósceles o escaleno. Recuerda que un triángulo es equilátero si sus tres lados son iguales, isósceles si dos de sus lados son iguales, y escaleno si ninguno de sus lados es igual.

7. **Determinación de la estación del año**: Escribe un programa que pida al usuario un mes (en forma de número del 1 al 12) e imprima la estación del año correspondiente. Recuerda que diciembre (12), enero (1) y febrero (2) corresponden al invierno; marzo (3), abril (4) y mayo (5) a la primavera; junio (6), julio (7) y agosto (8) al verano; y septiembre (9), octubre (10) y noviembre (11) al otoño.

8. **Determinación de la validez de una fecha**: Escribe un programa que pida al usuario un día, un mes y un año (todos en forma de números) e imprima si la fecha es válida o no. Recuerda que los meses tienen diferentes números de días (por ejemplo, febrero tiene 28 o 29 días, mientras que abril tiene 30 días), y debes tener en cuenta los años bisiestos.