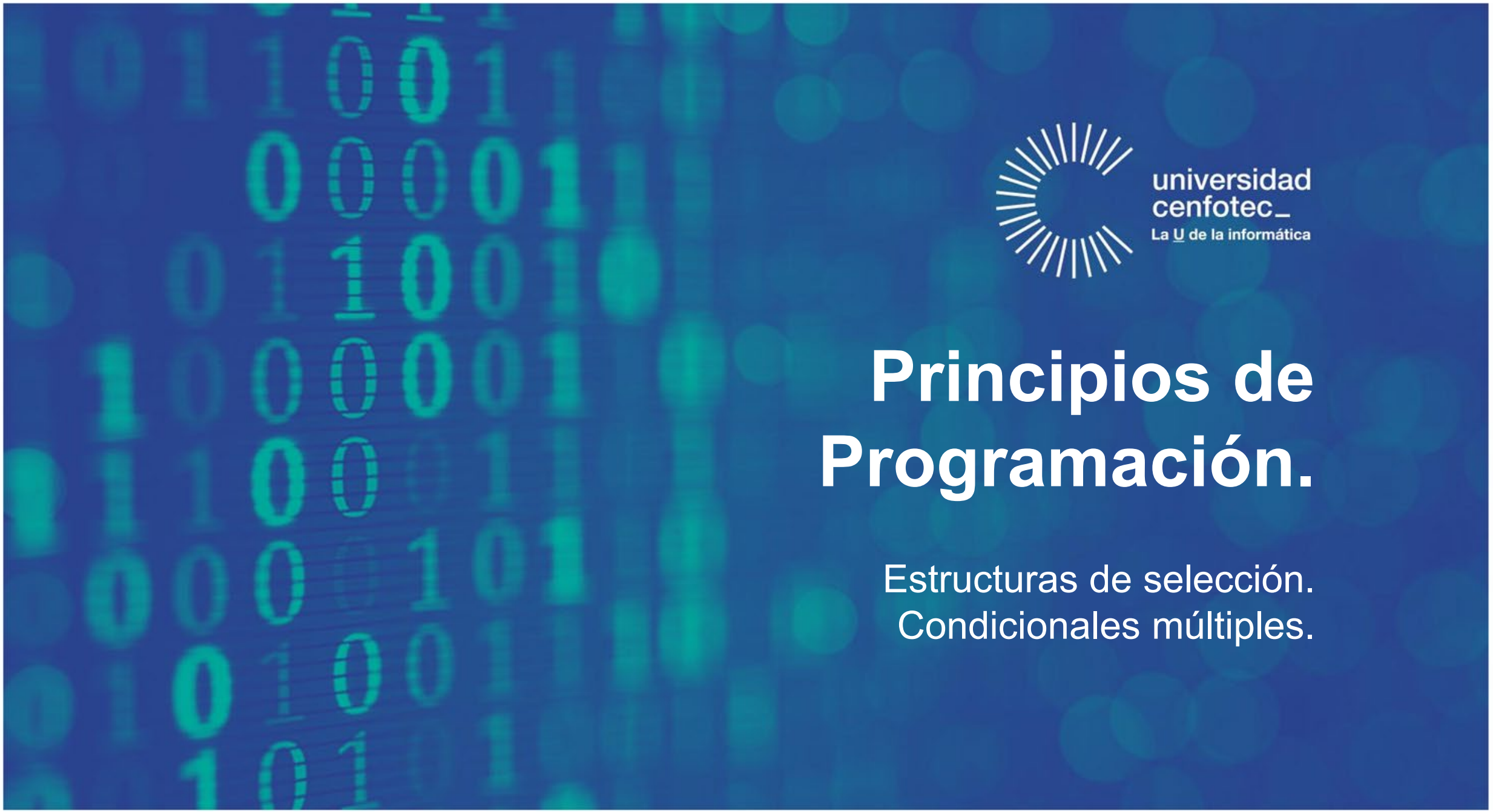


universidad cenfotec_
La U de la informática

Principios de Programación.

Estructuras de selección.
Condicionales múltiples.

Objetivos

universidad cenfotec
La U de la informática

- Describir las estructuras de selección múltiples.
- Resaltar la diferencia entre una condicional simple o doble en oposición con la múltiple.
- Aplicar el principio DRY en la construcción de estructuras de selección como buena práctica.
- Programar las estructuras de selección múltiple en el lenguaje Python.

Estructuras de selección múltiple

universidad cenfotec
La U de la informática

Existen situaciones en las que se tienen múltiples alternativas al tomar una decisión; tome por ejemplo la elección de películas a la hora de ir al cine, tipos de helados en una heladería, la elección de colores de pintura o la elección de un corte de cabello.

Las **estructuras de selección múltiple** permiten representar un conjunto de opciones como las que se mencionaron anteriormente separándolas en casos.

Diferencias entre las estructuras de selección

universidad cenfotec
La U de la informática

Simple o doble.

- Separan el algoritmo en uno o dos posibles flujos.
- El flujo se controla con una condición de verdadero o falso.
- No existe una posibilidad adicional a los casos de verdadero o falso de la condición.

Múltiple.

- Separan el algoritmo en múltiples posibles flujos.
- El flujo se controla de acuerdo a un valor dado, por ejemplo, el valor de una variable.
- Se construye con un caso 'predeterminado' adicional para contemplar cuando el valor de la variable no corresponde a ninguno previsto.

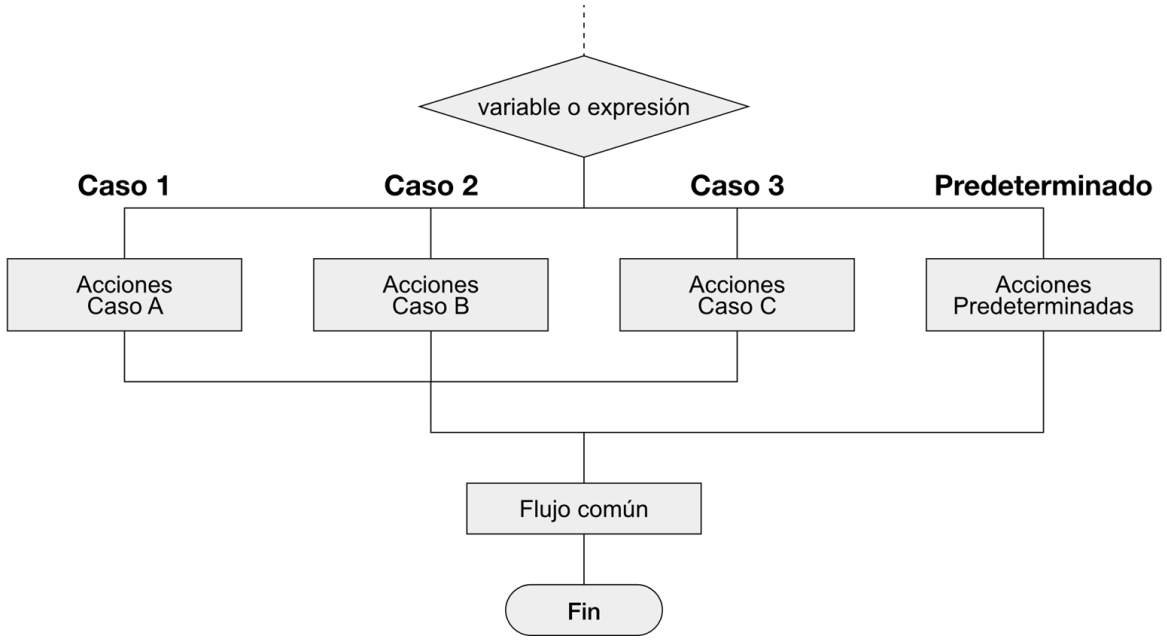
Descripción de la estructura

universidad cenfotec
La U de la informática

Los casos

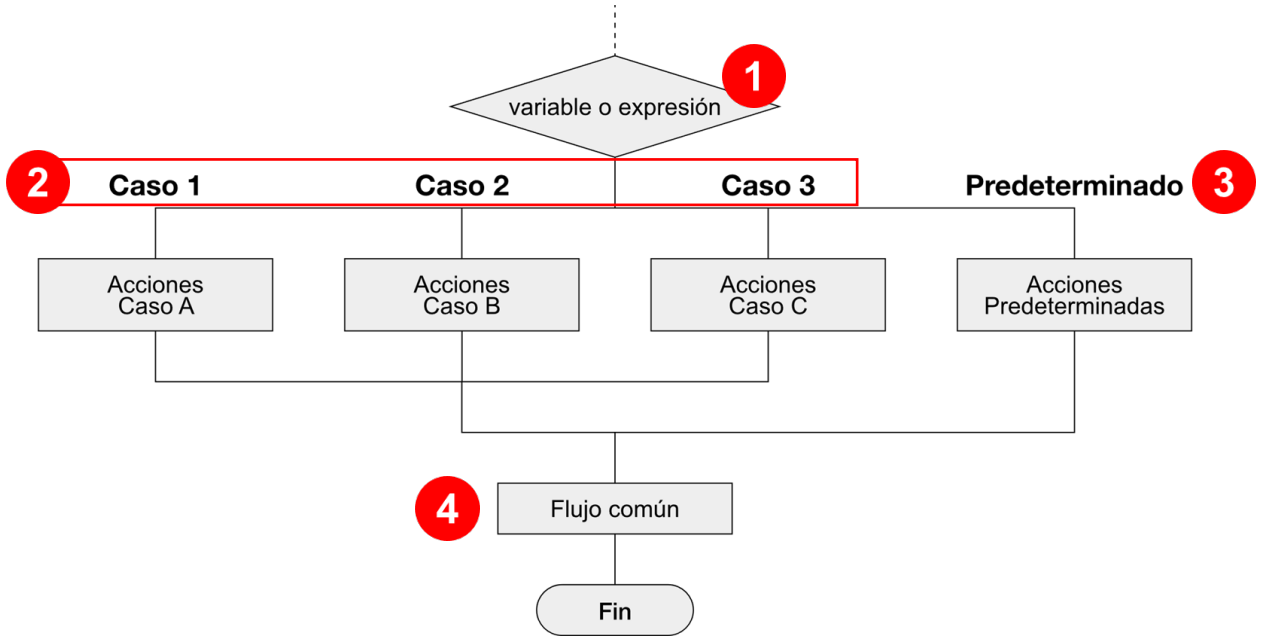
En esta estructura la condición no depende de un valor *verdadero* o *falso*, sino que toma el valor de una variable o una expresión y de acuerdo con ese valor se verifica cada caso.

Es importante tener un caso predeterminado que va a abarcar todos los otros valores que pueda tomar la variable o la expresión, que no estén contemplados en los casos definidos.



Descripción de la estructura

1. La variable o expresión que define el flujo a ejecutar.
2. Los casos que se definen para resolver el problema
3. Predeterminado o *default* es el nombre del caso por defecto que se ejecuta si la variable o expresión no concuerda con los casos definidos. Este caso va siempre al final.
4. Después de ejecutar las acciones del caso, el flujo vuelve a un punto en común.

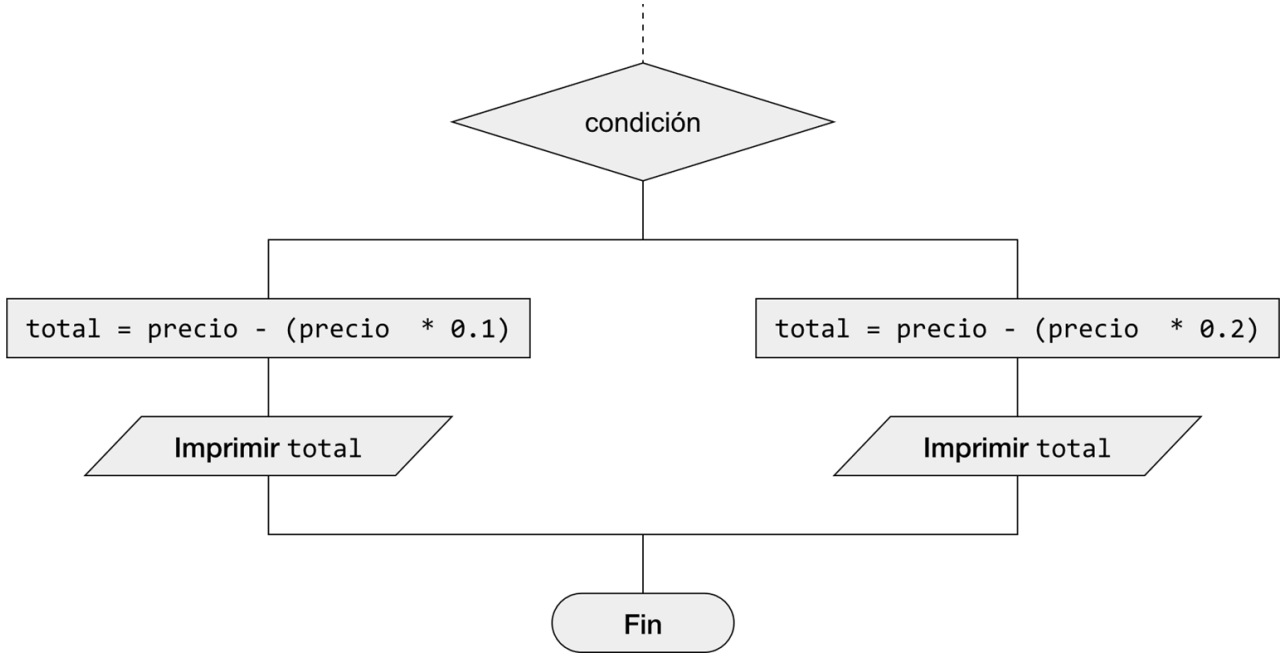


El principio DRY *(Don’t repeat yourself)*

universidad cenfotec
La U de la informática

Este principio lleva su nombre por las siglas en inglés de '*Don’t repeat yourself*' (no se repita), este consiste en que si existen líneas de código que hacen lo mismo en dos flujos distintos, estas líneas de código se pueden escribir cuando se retorne al flujo común.

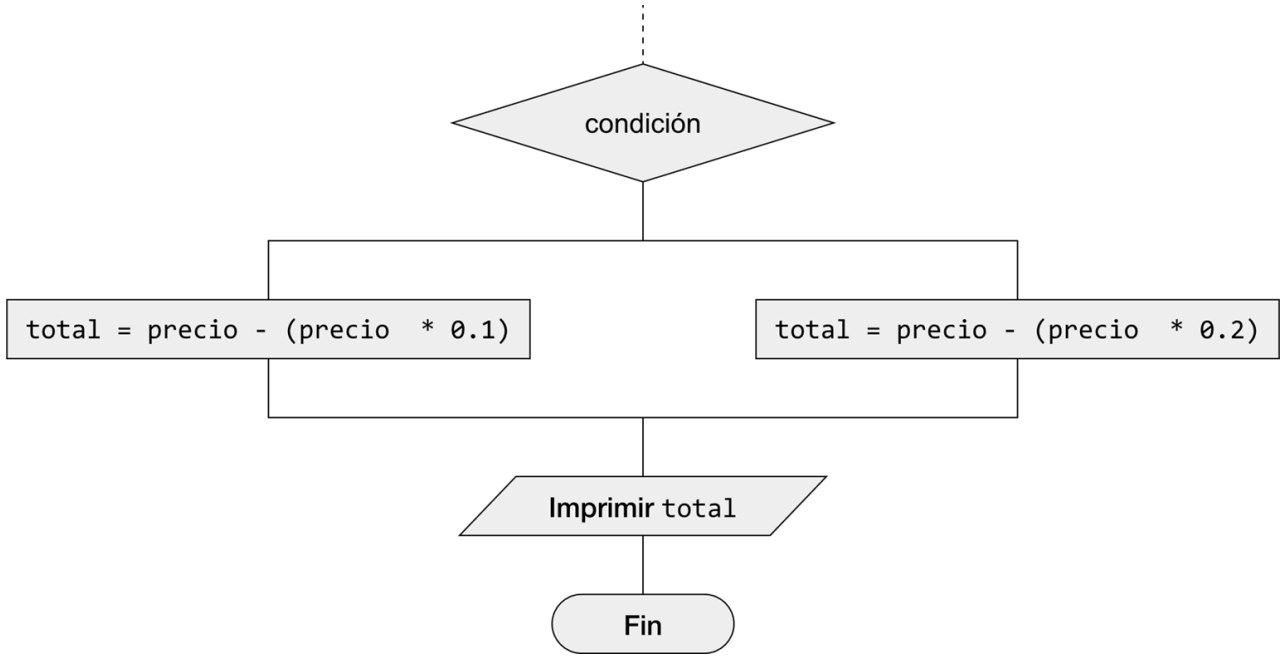
En el ejemplo a continuación puede observar que se imprime la variable **total** en ambos casos, cuando la condición es verdadera y cuando es falsa.



El principio DRY *(Don’t repeat yourself)*

universidad cenfotec
La U de la informática

Siguiente la recomendación del principio DRY en este ejemplo se ejecuta la impresión de la variable total en el flujo común, una vez se finalice con el cálculo del valor de la variable.



Implementación en Python

universidad cenfotec
La U de la informática

- A diferencia de la mayoría de lenguajes de programación Python no posee una estructura específica para implementar una **condición múltiple**. Por lo que se va a utilizar una adaptación de las estructuras condicionales dobles llamada elif.

- A continuación, se deben definir los valores de los **casos** a tener en cuenta para resolver el problema. Dentro de cada caso se define el **grupo de acciones** a ejecutar cuando la variable o expresión concuerde con el valor definido del case.

- Es importante tener en cuenta el **valor predeterminado** o por *default*, cuando la variable o expresión no concuerde con alguno de los casos definidos anteriormente.

Implementación en Python

universidad cenfotec_
La U de la informática

1. La variable o expresión que controla la condición debe tener un valor concreto que se evalúa para definir el caso a ejecutar.

1. A cada caso se asocia a un **valor** y cuando la variable o expresión concuerde con este, se ejecutarán las acciones definidas dentro del escenario.

1. La estructura *else* en éste caso representa el caso **predeterminado**, es decir cuando la variable o expresión no concuerde con alguno de los casos definidos anteriormente. En Python lo podemos representar con un *else* al final.

```
tipo_cliente = 5
precio = 25000
total = 0

if(tipo_cliente == 1):
   total = precio - (precio * 0.1)
elif(tipo_cliente == 2):
   total = precio - (precio * 0.2)
else:
   total = precio

print(total)
```

1 2 3

Implementación en Python

Puntos importantes.

1. Es posible ejecutar varias instrucciones dentro de un caso, no solo una.

1. El flujo se retoma normalmente una vez que la estructura se cierra.

```
tipo_cliente = 5
precio = 25000
total = 0

if(tipo_cliente == 1):
   print('Ejecutando el escenario 1')
   total = precio - (precio * 0.1)
elif(tipo_cliente == 2):
   print('Ejecutando el escenario 2')
   total = precio - (precio * 0.2)
else:
   total = precio

print(total)
```

1

2

Implementación en Python

El valor de la condición puede ser de cualquier tipo de dato básico, no solo números enteros.

Es importante que los casos tengan valores que coincidan con el tipo de dato de la variable de control para que la comparación pueda llevarse a cabo.

```
tipo_sangre = 'B'
if(tipo_sangre == 'A'):
   print('Tipo de sangre A')

elif(tipo_sangre == 'B'):
   print('Tipo de sangre B')
elif(tipo_sangre == 'O'):
   print('Tipo de sangre O')
```

```
idioma = 'español'
if(idioma == 'español'):
   print('¡Hola!')
elif(idioma=='inglés'):
   print('Hello!')
elif(idioma=='portugués'):
   print('Oi!')
```


universidad
cenfotec_
La U de la informática