

Estructuras de control



¿Qué son?

Un programa se ejecuta mediante una serie de instrucciones, que por norma general son ejecutadas una tras otra de manera secuencial.

```
costo = float(input("Ingrese el costo del producto: "))  
precio = float(input("Ingrese el precio de venta del producto: "))  
  
utilidad = precio - costo  
print("Tuviste una utilidad de: ", utilidad)
```

¿Qué son?

En muchas ocasiones no basta con seguir una estructura secuencial, puede ser que ciertas instrucciones se tengan que ejecutar si y sólo si se cumple una determinada condición.

También se puede necesitar repetir un determinado bloque de código más de una vez.

Condicionales

if - elif - else

Condicionales **if** - **elif** - **else**

Permite cambiar el flujo de ejecución de un programa, haciendo que ciertos bloques de código se ejecuten si y sólo si se dan determinadas condiciones

Estructuras Simples

Las **estructuras simples** cuentan con solo una condición (**if**), cuando esta se cumple se ejecuta el bloque de código que esté dentro de la estructura.

```
numero = 10
if numero > 5:
    print("Mayor")
```

Estructuras dobles

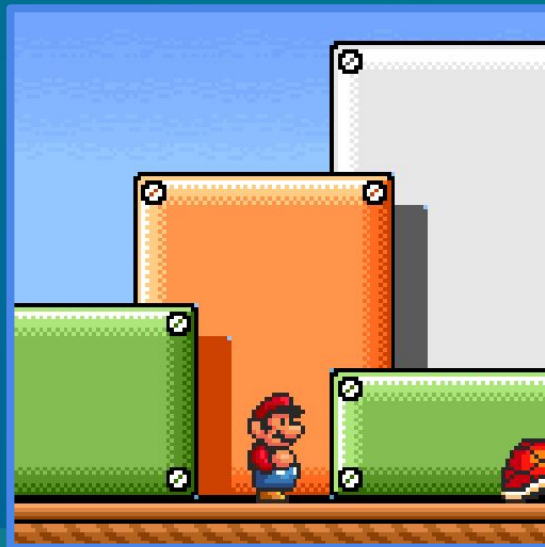
Las **estructuras dobles** constan de una condición (**if**) y una excepción a la condición (**else**). En caso de cumplirse la condición se ejecuta un bloque de código determinado, y en caso de NO cumplirse se ejecuta un bloque de código diferente.

```
numero = 5
if numero == 1:
    print("El número es 1")
else:
    print("El número no es 1")
```

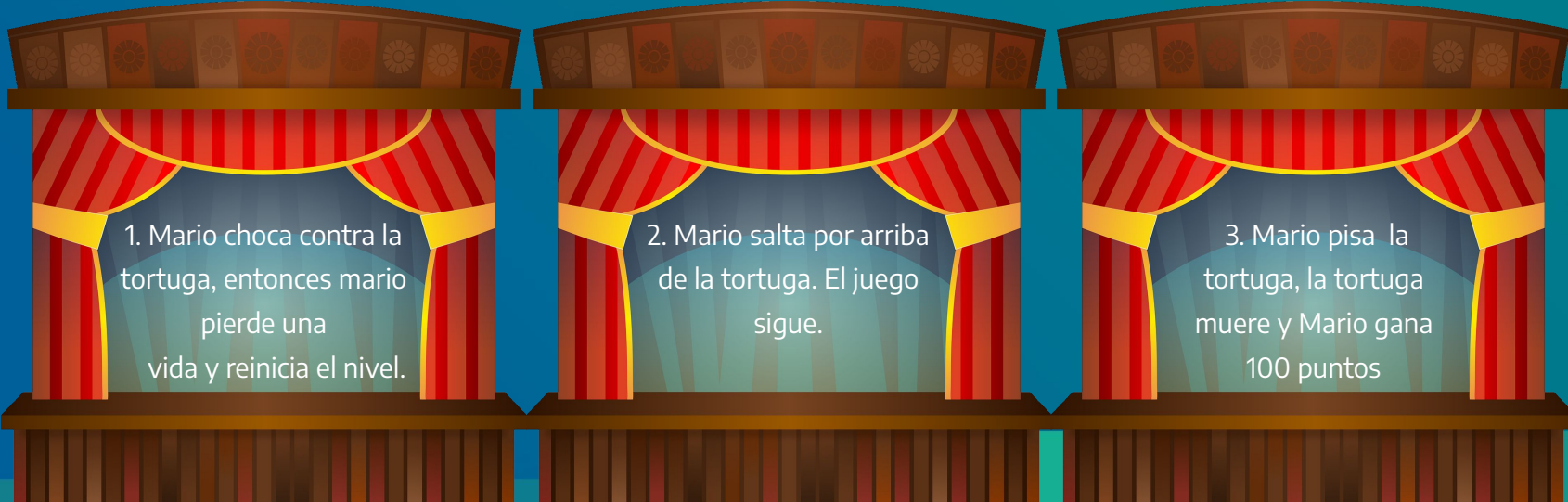
Estructuras múltiples

Las **estructuras múltiples** son necesarias cuando debemos evaluar varias condiciones, todas excluyentes entre sí. al tener escenarios establecidos o previamente definidos para una variable, nos permite tomar la ruta de decisión en la que se encuentre una coincidencia entre el valor de la variable y los escenarios.

¿Qué podría pasar si Mario se encuentra con la tortuga?



Escenarios posibles



1. Mario choca contra la tortuga, entonces mario pierde una vida y reinicia el nivel.

2. Mario salta por arriba de la tortuga. El juego sigue.

3. Mario pisa la tortuga, la tortuga muere y Mario gana 100 puntos

Estructuras múltiples (con anidamiento)

```
if "Mario choca la tortuga":  
    vidas -= 1  
    reiniciar_nivel()  
else:  
    if "Mario salta por arriba de la tortuga":  
        mario_avanza()  
    else: #Mario pisa la tortuga, seria por defecto  
        matar_tortuga()  
        puntaje += 100
```

Estructuras múltiples (con elif)

```
if "Mario choca la tortuga":  
    vidas -= 1  
    reiniciar_nivel()  
elif "Mario salta por arriba de la tortuga":  
    mario_avanza()  
else: #Mario pisa la tortuga, seria por defecto  
    matar_tortuga()  
    puntaje += 100
```

Ejemplo

Una empresa desea saber si tuvo pérdida o utilidad en base al precio de costo de un producto, y al precio al que fue vendido. En caso de que los precios sean iguales, saldrá empatado

```
costo = float(input("Ingrese el costo del producto: "))
precio = float(input("Ingrese el precio de venta del producto: "))
if costo < precio:
    utilidad = precio - costo
    print("Tuviste una utilidad de: ", utilidad)
elif costo > precio:
    perdida = precio - costo
    print("Tuviste una pérdida de: ", perdida)
else:
    print("Saliste empatado")
```

Selección múltiple:

match

Selección múltiple: **match**

Una sentencia match recibe una expresión y compara su valor con patrones sucesivos dados en uno o más bloques case.

```
que_paso = encontrar_tortuga()
match que_paso:
    case 1:
        vidas -= 1
        reiniciar_nivel()
    case 2:
        mario_avanza()
    case 3:
        matar_tortuga()
        puntaje += 100
```

Selección múltiple: **match**

También es posible combinar varios literales en un solo patrón usando | («ó»):

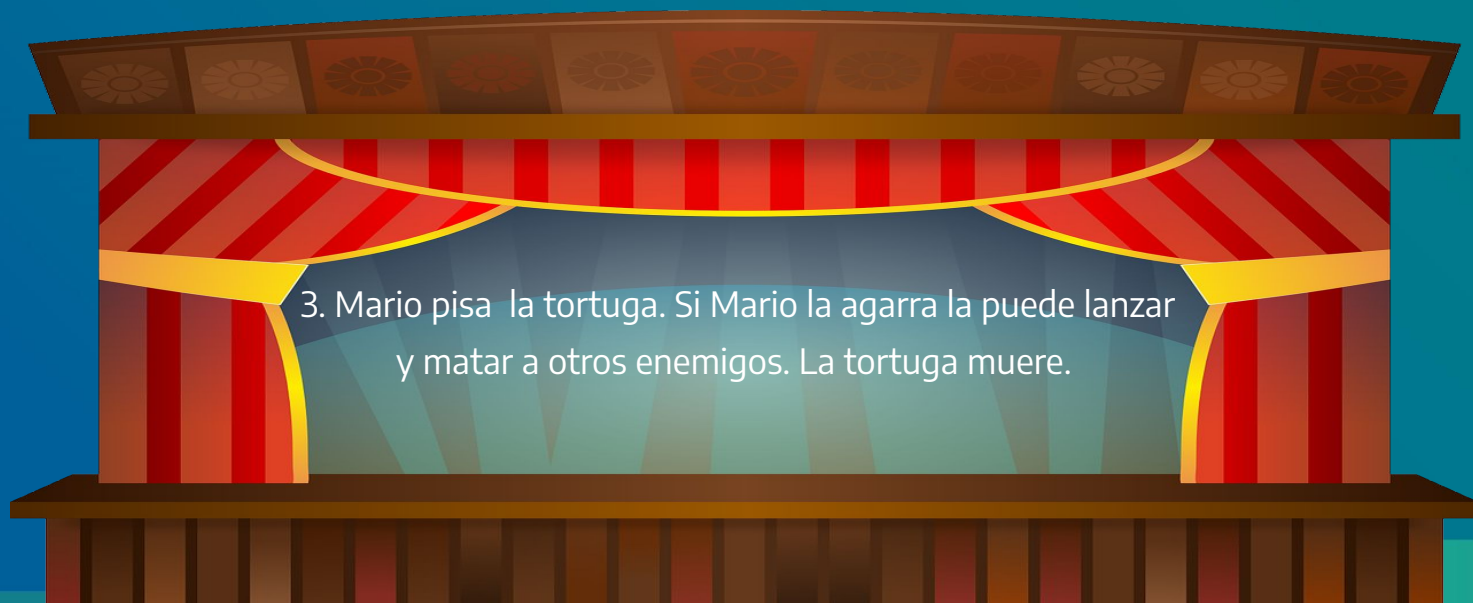
```
match comida_seleccionada:  
    case "Pizza":  
        print("¡Excelente elección, la pizza nunca falla!")  
    case "Ensalada" | "Vegetales":  
        print("Buena elección para una comida saludable.")  
    case "Sushi":  
        print("Una opción sofisticada y deliciosa.")
```


Selección múltiple: **match**

Y podemos tener un caso *por defecto* usando « _ »

```
que_paso = encontrar_tortuga()
match que_paso:
    case 1:
        vidas -= 1
        reiniciar_nivel()
    case 2:
        mario_avanza()
    case _: #case default
        matar_tortuga()
        puntaje += 100
```

Si modificamos el tercer caso del ejemplo de Mario...



Sólo con if - elif - else

```
if "Mario choca la tortuga":  
    vidas -= 1  
    reiniciar_nivel()  
elif "Mario salta por arriba de la tortuga":  
    mario_avanza()  
elif "Mario pisa la tortuga":  
    if "Mario Agarra la tortuga":  
        lanzar_tortuga()  
        matar_enemigos()  
    else:  
        matar_tortuga()  
        puntaje += 100
```

Usando **if - else** junto con **match**

```
que_paso = encontrar_tortuga()
match que_paso:
    case 1:
        vidas -= 1
        reiniciar_nivel()
    case 2:
        mario_avanza()
    case 3: # en el caso que pise la tortuga
        if "Mario agarra la tortuga":
            lanzar_tortuga()
            matar_enemigos()
        else:
            matar_tortuga()
            puntaje += 100
```