UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Estructuras de Control

Programación y Laboratorio I

Versión 22.1

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

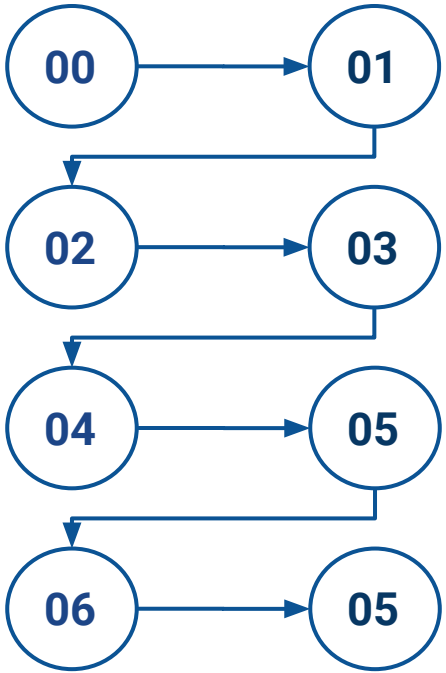
Estructuras de Control

¿Qué son?

Bucle while

Bucle for

Continue



Condicional

Funcion range

Break

Selección

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

¿Qué son?

Un código es una secuencia de instrucciones, que por norma general son ejecutadas una tras otra de manera **secuencial**

```
costo = 20
precio = 23
utilidad = precio - costo
print("La utilidad es: ", utilidad)
```

¿Qué son?

En muchas ocasiones no basta, 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.

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Condicional: if, elif y else

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

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Condicional: if, elif y else

```
costo = 20
precio = 23
if (costo < precio):
  utilidad = precio - costo
  print("La utilidad es: ", utilidad)
elif (costo > precio):
  perdida = precio - costo
  print("La pérdida es: ", perdida)
else:
  print("Saliste empatado")
```

Bucle: while

El uso del while permite ejecutar una sección de código mientras una condición determinada se cumpla.

Cuando se deje de cumplir, se saldrá del bucle y se continuará la ejecución normal.

Bucle: while

```
respuesta = 's'
while(respuesta == 's'):
   respuesta = input("¿Desea continuar? (s/n) ")
```

Funcion: range

El **range**() genera una secuencia de números que van desde cero por defecto hasta el número que se pasa como parámetro menos uno.

```
lista_numeros = list(range(5))
print(lista_numeros) # [0,1,2,3,4]
```

Funcion range

También se le pueden pasar hasta tres parámetros separados por coma (inicio, fin y salto)

```
lista_numeros = list(range(10, 20, 2))
print(lista_numeros) # [10, 12, 14, 16, 18]
```

Bucle: for

En este tipo de bucle el número de iteraciones de un está definido de antemano, mientras que en un **while** no.

En Python el **for** permite recorrer los elementos de un objeto iterable.

Bucle: for

En este tipo de bucle el número de iteraciones de un está definido de antemano, mientras que en un **while** no.

En Python el **for** permite recorrer los elementos de un objeto iterable.

Bucle: for

```
lista_numeros=[1,2,4,5,77,-1]
for numero in lista_numeros:
  print(numero,end=" ")

#Salida: 1 2 4 5 77 -1
```

Bucle: break

La sentencia break nos permite alterar el comportamiento de los bucles **while** y **for**. Permite terminar con la ejecución del bucle.

```
lista_numeros=[1,2,4,5,77,-1]
for numero in lista_numeros:
  if(numero==5):
     break
  print(numero,end=" ")

#Salida 1 2 4
```

Bucle: continue

La sentencia **continue** se salta todo el código restante en la iteración actual y vuelve al principio en el caso de que aún queden iteraciones por completar.

```
lista_numeros=[1,2,4,5,77,-1]
for numero in lista_numeros:
  if(numero==5):
     continue
  print(numero,end=" ")

#Salida 1 2 4 77 -1
```

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Seleccion: match

La sentencia **match** permite organizar bloques de códigos, de forma que se ejecuten cuando se cumple cierta condición o caso.

```
match status:
        case 400:
            print("Error de request")
        case 404:
            print("No encontrado")
        case _:
            print("Algo salio mal")
```

Selección: match

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

```
match status:
        case 400 | 401:
            return "Error de request"
        case 404:
            return "No encontrado"
        case other:
            return "Algo salio mal"
```