Condicional: if

- If:
  - if sintaxis:
  - Las clausulas elif y else son opcionales.
  - No existe el operador case en python; se puede usar la estructura if/elif/else.

```
if <condition>:
    <statements>
[elif <condition>:
    <statements>]
[elif <condition>:
  • pass]
...
[else:
    <statements>]
```

Condicional: if

- Ejemplos:

-
```
a=7
if a>6:
    print("Es mayor que 6")
```
-
```
if 1: # 1 significa verdadero
    print("sip")
```

-
```
if a == 1:
    print("1")
elif a == 2:
    print("2")
else:
    print("Mayor a 2")
```

Condicional: if

- **if anidados:**

```
a=1
b=2
if a==1:
    if b==2:
        print("a es 1 y b es 2")
```

- **if en una línea**

```
if a>4: print("Greater")
```

Bucle: for

```
for <variable> in <secuencia>:
    <sentencias>
[else:
    <sentencias>]
```

- La sentencia else es opcional y siempre se ejecuta al menos que se ejecute la sentencia break dentro del bucle.
- Se itera por cada valor en la secuencia, en cada iteración la variable tomando el valor correspondiente en la secuencia.
- Las secuencias pueden ser: una lista, una tupla, un diccionario, un conjunto o un string.

Bucle: for

- Ejemplos:
- #Iterando con una lista:

```
frutas = ["banana", "uva"]
for x in frutas:
  print(x)
```

- #Iterando con cadenas:

```
for x in "banana":
  print(x)
```

- #usando la función range (genera una secuencia de números):

```
for i in range(6):
    print(i, end=', ')
```

Bucle: while

```
while <condición>:
    <sentencias>
[else:
    <sentencias>]
```

- La sentencia else es opcional y siempre se ejecuta al menos que se ejecute la sentencia break dentro del bucle.
- Se itera mientras la condición sea verdadera.
- Es importante modificar dentro del bucle los elementos que forman la condición para finalizar las iteraciones.

Bucle: while

- Ejemplos:
- ```
  i = 1
  while i < 6:
    print(i)
    i += 1
  ```
- ```
  #En una línea
  a=3
  while a>0: print(a); a-=1
  ```
- ```
  a=3
  while(a>0):
      print(a)
      a-=1
  else:
      print("a llego a 0")
  ```
  ```

Bucle anidados

- #Bucles for anidados:

```
for i in range(1,6):
        for j in range(i):
            print("*",end=' ')
        print()
```

- #Bucles while anidados:

```
i=6
while(i>0):
        j=6
        while(j>i):
            print("*",end=' ')
            j-=1
        i-=1
        print()
```

Control de bucles

- Modificar el comportamiento normal de los bucles en python: continue y break.
- **continue**:
  - detiene la iteración actual y continua con la siguiente.
- **break**:
  - detiene la iteración actual y todas las restantes.
- En python no se puede especificar que bucle anidado se pretende controlar.

Control de bucles

- #break

```
for i in 'break':
        print(i)
        if i=='a': break;
```

- # continue

```
i=0
while(i<8):
        i+=1
        if(i==6): continue
        print(i)
```

Control de bucles

```
#Con bucles anidados
for x in range(10):
    for y in range(10):
        print (x*y)
        if x*y > 50:
            break
    else:
        continue
    break
```