

Clase 21-10-2024

LINK: <https://ellibrodepython.com/paso-por-valor-y-referencia>

<https://www.programiz.com/python-programming/online-compiler/>

<https://pythontutor.com/visualize.html#mode=edit>

- Append es agregar

```
x = [10, 20, 30]
def funcion(entrada):
    entrada = []

funcion(x)
print(x)
# [10, 20, 30]
```

-
- If es si
- El programa se divide en cajas y en la página de python tutor lo refleja gráficamente.
- Luego de crear una clase, se comienza a dibujar
- Hay dos clases instancias y clase (es común para todos los objetos)
- Es importante la programación orientada a objeto.
- Init es para llamar y luego se comienza a clasificar y se nombran los atributos-

- Python tutor

Python 3.6
[known limitations](#)

```
1 # Creando una clase vacía
→ 2 class Perro:
3     pass
```

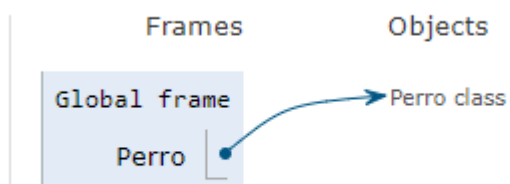
[Edit this code](#)

It just executed
Click here to execute

<< First < Prev Next > Last >>

Step 1 of 1

Follow our [YouTube](#), [TikTok](#), and [Instagram](#) for free tutorials



Write code in Python 3.6 [reliable stable version, select 3.11 for newest] ▼

```
1 # Creando una clase vacía
2 class Perro:
3     pass
```

self.en_marcha

```
main.py [Icons] [Share] [Run] [Clear]
1 # Creando una clase vacia
2 class Perro:
3     pass
4 mi_perro = Perro()
5
6 class Perro:
7     # El método __init__ es llamado al crear el
      objeto
8     def __init__(self, nombre, raza):
9         print(f"Creando perro {nombre}, {raza}")
10
11     # Atributos de instancia
12     self.nombre = nombre
13     self.raza = raza
14
15 mi_perro = Perro("Toby", "Bulldog")
16 print(type(mi_perro))
17 # Creando perro Toby, Bulldog
18 # <class '__main__.Perro'>
19
20 print(mi_perro.nombre) # Toby
21 print(mi_perro.raza) # Bulldog
22
23 class Perro:
24     # Atributo de clase
25     especie = 'mamífero'
26
27     # El método __init__ es llamado al crear el
      objeto
28     def __init__(self, nombre, raza):
29         print(f"Creando perro {nombre}, {raza}")
30
31     # Atributos de instancia
32     self.nombre = nombre
33     self.raza = raza
```

Output

```
Creando perro Toby, Bulldog
<class '__main__.Perro'>
Toby
Bulldog

=== Code Execution Successful ===
```

- Self es porque la lista puede ser desordenada, es para tipear algo.

```

1 #Definir la clase coche
2 class coche:
3     #El metodo __init__ es el constructor, se llama al crear un objeto de la
4     def __init__(self,marca,modelo,color):
5         self.marca = marca #atributo de marca
6         self.modelo = modelo #atributo modelo
7         self.color = color #atributo color
8         self.en_marcha = False #atributo para saber si el coche esta en ma
9
10    #Metodo para arrancar el coche
11    def arrancar(self):
12        if not self.en_marcha:
13            self.en_marcha = True
14            print(f"El coche {self.marca} {self.modelo} ha arrancado.")
15
16    #Metodo para detener el coche
17    def detener(self):
18        if not self.en_marcha:
19            self.en_marcha = False
20            print(f"El coche {self.marca} {self.modelo} se ha detenido.")
21        else:
22            print(f"El coche {self.marca} {self.modelo} ya esta detenido.")
23
24    #Metodo para mostrar el estado del coche
25    def mostrar_estado(self):
26        print(f"Coche {self.marca} {self.modelo} color:{self.color}, e
27
28    #Crear objetos (instancias) de la clase coche
29
30    mi_coche = coche ("Toyota", "Corolla", "Rojo")
31    tu_coche = coche ("Honda", "Civic", "Azul")
32
33    #Mostrar el estado de los coches después de arrancar
34
35    mi_coche.mostrar_estado()
36    tu_coche.mostrar_estado()
37
38    #Detener los coches
39
40    mi_coche.detener()
41    tu_coche.detener()
42
43    #Mostrar el estado final de los coches
44
45    mi_coche.mostrar_estado()
46    tu_coche.mostrar_estado()
47

```

#Definir la clase coche

class coche:

```
#El metodo_init_ es el constructor, se llama al crear un objeto de la clase
def _init_(self,marca,modelo,color):
    self.marca = marca #atributo de marca
    self.modelo = modelos #atributo modelo
    self.color = color #atributo color
    self.en_marcha = False #atributo para saber si el coche esta en marcha
```

```
#Metodo para arrancar el coche
def arrancar(self):
    if not self.en_marcha:
        self.en_marcha = true
        print(f"El coche {self.marca}{self.modelo} ha arrancado.")
```

```
#Metodo para detener el coche
def detener(self):
    if not self.en_marcha:
        self.en_marcha = false
        print(f"El coche {self.marca}{self.modelo} se ha detenido.")
    else:
        print(f"El coche {self.marca}{self.modelo} ya esta detenido.")
```

```
#Metodo para mostrar el estado del coche
def mostrar_estado(self):
    print(f"Coche {self.marca}{self.modelo} color:{self.color},
estado{estado}")
```

```
#Crear objetos (instancias) de la clase coche
```

```
mi_coche = coche ("Toyota", "Corolla", "Rojo")
tu_coche = coche ("Honda", "Civic", "Azul")
```

```
#Mostrar el estado de los coches después de arrancar
```

```
mi_coche.mostrar_estado()
tu_coche.mostrar_estado()
```

```
#Detener los coches
```

```
mi_coche.detener()  
tu_coche.detener()
```

#Mostrar el estado final de los coches

```
mi_coche.mostrar_estado()  
tu_coche.mostrar_estado()
```