Tipos de datos

- Tipos básicos:
  - Números
  - Cadenas de texto
  - Booleanos
- Colecciones de datos (estructuras de datos)
  - Una estructura de datos es una forma de organizar un conjunto de datos con el objetivo de facilitar su manipulación

Tuplas

- **Una tupla es una variable que permite almacenar varios datos inmutables de distintos tipos:**

tupla = (5, 5.3, “hola”, 3)

t=3,'a',7

print(type(t))

- **Como vemos se pueden combinar de distintas formas y tipos; tanto enteros, como decimales, como cadenas de texto, etc…**

Tuplas

- Accediendo a cada elemento:

  print(tupla[1])

- No puede modificarse:

  tupla[0] = 7 ERROR!!!

- Intervalo:

  tupla[1:3]

Listas

- Una lista es una estructura de datos que contiene una colección o secuencia de datos. Los datos o elementos de una lista deben ir separados con una coma y todo el conjunto entre corchetes. Se dice que una lista es una estructura mutable porque además de permitir el acceso a los elementos, pueden suprimirse o agregarse nuevos.

```
ListaEstaciones = ["Invierno", "Primavera",
   "Verano", "Otoño"]  # Declara lista
```

Listas

- Otro ejemplo:

```
lista = [22, True, "una lista", [1, 2]]
```

- Acceso a los elementos de una lista

```
lista = [11, False]
mi_var = lista[0] # mi_var vale 11
print(mi_var)
>>11
```

Listas

- Modificar los elementos de la lista:

```
lista=[22, False]
lista[0]=54
```

- Agregar un elemento al final de la lista:

```
nombres = ["Jose", "Miguel", "David"]
nombres.append("Juan")
print(nombres)
>>['Jose', 'Miguel', 'David', 'Juan']
```

Listas

- Agregar varios elementos al final de la lista

```
nombres.extend(["Jose", "Gerardo"])
print(nombres)
>>['Jose', 'Miguel', 'David', 'Juan', 'Jose', 'Gerardo']
```

- Agregar un elemento en una posición determinada

```
nombres.insert(0, "Dani")
print(nombres)
>>['Dani', 'Jose', 'Miguel', 'David', 'Juan', 'Jose',
  'Gerardo']
```

Listas

- Extraer el último elemento de la lista

```
ultimoNombre=nombres.pop()
print(nombres)
>>['Dani', 'Jose', 'Miguel', 'David', 'Juan', 'Jose']
```

- Extraer un elemento por su índice

```
nom2=nombres.pop(2)
print(nombres)
>>['Dani', 'Jose', 'David', 'Juan', 'Jose']
```

Listas

- Eliminar el último elemento de la lista

```
del(nombres[-1])
print(nombres)
>>['Dani', 'Jose', 'Miguel', 'David', 'Juan', 'Jose']
```

- Eliminar un elemento por su índice

```
del(nombres[2])
print(nombres)
>>['Dani', 'Jose', 'David', 'Juan', 'Jose']
```

Listas

- Mostrar el número de elementos de una lista. Longitud lista

```
nombres = ['Dani', 'Jose', 'Miguel', 'David', 'Juan',
  'Jose']
print(len(nombres))
>>6
```

- También sirve para cadenas de texto

```
cadena="Estructuras de datos"
print(len(cadena))
>>20
```

Listas

- Mostrar el contenido de una lista. Dos formas:

```
cont=0
while(cont<len(nombres)):
    print(nombres[cont])
    cont=cont+1

for nombre in nombres:
    print(nombre)
```

Diccionarios

- Creación:

```
dic={} #utilizando llaves
dic2={1:'a',2:'b'}
print(dic2)
```

- Lectura de elementos:

```
print(dic2[1])
```

- Modificación:

```
dic2[1]='c'
print(dic2[1])
```

Diccionarios

- No se puede acceder a través de la posición:

```
print(dic2[0]) #ERROR!!!
```

- Añadir elementos:

```
dic2[3]='f'
print(dic2)
```

- Borrar:

```
del(dic2[1]) #un elemento
print(dic2) #todo el diccionario
```

Diccionarios

- Recuperar todas las claves:

```
dic={1:'a',2:'b'}
print(dic.keys())
```

- Recuperar los valores:

```
print(dic.values())
for i in dic.values():
    print(i)
```

Clases

- Se pueden ver como tipos de datos que crea el usuario
- Como los otros datos, agrupan varios tipos de datos simples y compuestos
- Las clases son los modelos sobre los cuáles se construirán nuestros objetos
- Propiedades o atributos: son las características intrínsecas del objeto, los datos que queremos guardar. Éstas, se representan a modo de variables, solo que técnicamente, pasan a denominarse propiedades
- Ejemplo:

Clases

```
class Antena():
    color = ""
    longitud = ""

class Pelo():
    color = ""
    textura = ""

class Ojo():
    forma = ""
    color = ""
    tamanio = ""

class Objeto():
    color = ""
    tamanio = ""
    aspecto = ""
    antenas = Antena() # propiedad compuesta por el objeto objeto Antena
    ojos = Ojo()       # propiedad compuesta por el objeto objeto Ojo
    pelos = Pelo()     # propiedad compuesta por el objeto objeto Pelo
```

Clases

- Los métodos son funciones que técnicamente se denominan métodos, y representan acciones propias que puede realizar el objeto (y no otro):
- Ejemplo:

```
class Objeto():
    color = "verde"
    tamanio = "grande"
    aspecto = "feo"
    antenas = Antena()
    ojos = Ojo()
    pelos = Pelo()

    def flotar(self):
        print 12
```

Objetos

- Las clases por sí mismas, no son más que modelos que nos servirán para crear objetos en concreto. Podemos decir que una clase, es el razonamiento abstracto de un objeto, mientras que el objeto, es su materialización. A la acción de crear objetos, se la denomina instanciar una clase y dicha instancia, consiste en asignar la clase, como valor a una variable:
- Ejemplo:

Objetos

```
class Objeto():
    color = "verde"
    tamanio = "grande"
    aspecto = "feo"
    antenas = Antena()
    ojos = Ojo()
    pelos = Pelo()

    def flotar(self):
        print 12

et = Objeto()
print et.color
print et.tamanio
print et.aspecto
et.color = "rosa"
print et.color
```

Creación más actual

```
class clase1():
    def __init__(self, name="", salary=0):
        self.name = name
        self.salary = salary
class clase2():
    def __init__(self):
        self.at1 = 1
        self.at2 = 'valor'
```

Métodos especiales

- Constructor

  __init__

- Destructor

  __del__

- Impresión

  __str__

```
class clase2():
    def __init__(self):
        self.at1 = 1
        self.at2 = 'valor'
```

Métodos especiales

```
1  class clase1():
2      empCount = 0
3
4      def __init__(self, name="", salary=0):
5          self.name = name
6          self.salary = salary
7          clase1.empCount += 1
8
9      def displayCount(self):
10         print("Total Employee %d", clase1.empCount)
11
12     def displayEmployee(self):
13         print("Name : ", self.name,  ", Salary: ", self.salary)
14
15     def __del__(self):
16         class_name = self.__class__.__name__
17         print(class_name, "destroyed")
18
19     def __str__(self):
20         return "blablq"+self.name+" "+str(self.salary)
21
```