Diccionario

- **objeto**: en python: TODO. se representan con Clases. **objeto** es una **instancia** de una **clase.** Es el encapsulamiento de datos
- **instancia**: concretización, realización en memoria de lo que yo quiero crear
- **referencia**: en cada creación se crea una referencia, es el identificador o nombre de la instancia y no son equivalentes.
- **clase**: templates que permiten generalizar tipos de datos, se componen de tipos de datos nuevos que se construyen con las clases. Tienen atributos o componentes, también incluyen funciones que son parte del dato y sólo se activan cuando un dato los llama: **método**. También permiten redefinir **operadores**. **Son componentes + funciones + operadores**
- **atributo** (de dato): variables u objetos que definen el estado para una instancia, atributo de instancia o atributo de clase
- **método**: funciones asociadas a las clases, accesibles siempre a través del punto "." (de instancia, getters y setters, @classmethod, @staticmethod). **Función** que pertenece a un **objeto.** Utiliza **self** como primer argumento
- **operador**: operador punto: toma algo definido por una clase y accede a sus componentes. Operaciones matemáticas.
- **namespace**: espacio de nombre que existe cuando hay una lista de referencias que está asociada a una lista de objetos, este espacio tiene tiempo de vida y puede desaparecer. ej: (len, print)
- **scope**: ámbito, manera de encontrar namespace dentro del intérprete
- **polimorfismo**: Capacidad de los objetos de tomar muchas formas. Tener métodos distintos pero con mismo nombre en diferentes objetos. Permite que una función o método se aplique a objetos de diferentes clases, siempre que esas clases compartan una interfaz común (por ej, un método con el mismo nombre).
- **herencia**: crear objetos a partir de otros, capacidad de definir un objeto a partir de otro que está definido, de modo que la nueva hereda atributos y métodos de la original. Responde al verbo to be y establece jerarquía entre las clases
- **encapsulamiento**: Propiedad o práctica de restringir el acceso a componentes de un objeto, permite ocultar los detalles.

Important sentences

- **objeto** es una **instancia** de una **clase** definida (**referencia** es el nombre, **instancia** es el contenido)
- referencia **no** es equivalente a una instancia
- las **referencias** (identificador) apuntan a objetos (**instancias**)
- cada **objeto** tiene asociado un tipo: la **clase** de la instancia
- **atributos y métodos** de los objetos son accesibles con punto . desde sus **referencias**
- en C: **id()** es como el **&** que me dice dónde está físicamente
- en C: la **referencia** es como el **puntero***
- **listas** pueden contener objetos, como todo es un obj, puede contener todo, no importa si es int o str o whatever
- lista no es lo mismo que las referencias de listas
- scope level superior: variable global, **NO** hacer variables globales
- Para que las **instancias** tengan **atributos** declarados en un estado inicial, se utiliza el **método** especial **__init__(self)** (instanciación)
- **clase** es mi template para generar **instancias** del tipo de esa clase
- todo lo que tiene **self** pertenece a la instancia, **self**: referencia a la instancia que se crea usando esta clase
- los **métodos** también pueden ser considerados **atributos**

Important commands

- **dir():** dice cuáles son mis referencias
- **id():** identificador basado en la dirección de memoria donde se almacena
- _name: variables ocultas de python
- _ _name: referencias que guarda cosas de lib
- _ _ name _ _ **:** nombre de la rutina que está ejecutándose en este momento
- s. (**doble tab**): muestra los no ocultos
- **dir(s):** me muestra los ocultos _ _
- **__ str __:** sirve para darle atributos especiales a la fn print, genera una impresión de str de la instancia que llama al método
- **__ repr __:** representación oficial del objeto, análoga a la sintaxis de creación
- **vim** : ***te sales del intérprete y pones / para **buscar** , como el cmd+F
- constructor **%init%** destructor **%del%**