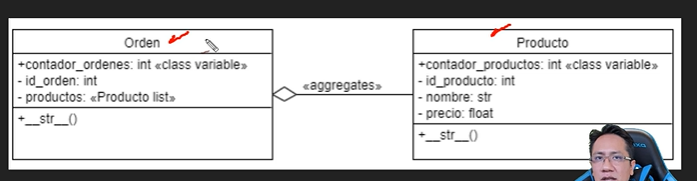
**DISEÑO DE CLASES EN PYTHON**



Este es el diagrama de clases para esta sección

(el símbolo – antepuesto a una variable de instancia determina que la misma debe estar encapsulada)

La idea es crear objetos de tipo Producto y agregarlos a la lista para la creación de objetos de tipo Orden.

Para iniciar se recomienda primero crear la clase producto, que es la que “no depende de otros objetos” para su creación.

La creación es una creación corriente, con la sobreescritura del método str para imprimir las características del producto.

Para la creación de la clase Orden, este es el código

    def \_\_init\_\_(self, \*productos):

        Orden.contador\_ordenes += 1

        self.\_id\_orden = Orden.contador\_ordenes

        self.\_productos = list(productos)

en el \_\_init\_\_ recibimos una lista y realizamos el casteo correspondiente para almacenarle en el atributo **\_productos**

La lista deberá recibir objetos de tipo Producto para el correcto funcionamiento del programa, para esto deben realizarse comprobaciones al momento de usar los métodos **calc\_precio() y add\_product()**

Add\_product() permite agregar un producto a la orden “fuera” del init:

    def add\_product(self, producto\_nuevo):

        if (isinstance(producto\_nuevo, Producto)):

            self.\_productos.append(producto\_nuevo)

        else:

            alerta = f'ALERT - no es posible agregar el producto "{producto\_nuevo}"'

            print(

                alerta.center(100, '-'))

recibe como parámetro el nuevo producto y se comprueba que el mismo sea de instancia de la clase Producto, en ese caso de agrega a la lista guardada en el atributo **.\_productos**

calc\_precio() nos permite obtener el precio de la orden en cierto momento:

    def calc\_precio(self):

        precio = float(0)

        contador = 0

        for e in self.\_productos:

            if(isinstance(e, Producto)):

                precio += e.precio

                contador += 1

        else:

            precio = float(precio)

            return(f'el precio de la orden es ${precio}. La cantidad de productos es: {contador}')

como se ve, se recorre la lista **\_productos** y en cada caso se comprueba que cada valor sea un objeto de tipo Producto para asi acceder a su atributo precio.

Por ultimo, se almacena en una variable el total y se retorna junto con un indicativo del total de productos.

El método \_\_str\_\_ de la clase Orden, se encarga de imprimir el numero de orden, el precio y la descripción de cada producto.

    def \_\_str\_\_(self):

        str\_productos = f'\n'

        count = 0

        for e in self.\_productos:

            str\_productos += f'{e.\_\_str\_\_()}\n'

            count += 1

        return f'[Numero de orden: {self.\_id\_orden}, Precio: {self.calc\_precio()}, Productos ({count} en total) :{str\_productos}'

se usa un for para recorrer el atributo **\_productos** y se almacena una llamada al método \_\_str\_\_(), el de la clase producto, para obtener la descripción del mismo y almacenar la de todos los productos en una variable “str\_productos” que luego se imprime en el retorno del metodo