

Evaluación Final Modulo 1 Data Analytics

letras en el
(pp) x 3
igual el mismo
poco.

A lo largo de esta evaluación tengo que crear una clase llamada:

Tienda Online

Class TiendaOnline: → Estoy creando una tienda, donde guardo mis productos

Método constructor, se ejecuta automáticamente.

`def __init__(self)` → Self es como una etiqueta de identificación

Para que se guarde dentro de la tienda

`self.inventario = []` → estantería de la tienda

`self.ventas_totales = 0.0` → caja registradora

Método 1: agregar producto

productos a la tienda

`def agregar_producto(self, nombre, precio, cantidad)`

`for producto in self.inventario:` → Bucle for para

comprobación

Modificación de cantidad

Actualización del precio

Esto lo

hago por si no encuentro el

producto, para crear uno nuevo

Para ello se usa el `append()`

`if producto["nombre"] == nombre:`

`producto["cantidad"] = producto["cantidad"] + cantidad`

`producto["precio"] = precio`

`return` → Lo uso para salir del método cuando el producto ya existe (anti-duplicado)

`nuevo_producto = {"nombre": nombre, "precio": precio, "cantidad": cantidad}`

`self.inventario.append(nuevo_producto)`

Método 2: ver inventario

`def ver_inventario(self)` → Nos sirve para ver información,

del dato

`for producto in self.inventario:` → Bucle for para recorrer la lista inventario

Es primordial para ver la info

`print("Nombre:", producto["nombre"], ", ", "Precio:", producto["precio"],
", ", "Cantidad:", producto["cantidad"])`

Método 3: buscar producto

`def buscar_producto(self, nombre):`

`for producto in self.inventario:` → bucle for busca un producto concreto

Si encuentro, imprimo los datos y uso return para salir del método.

`if producto["nombre"] == nombre:` → Comparo
`print("Nombre:", producto["nombre"], ", ", "Precio:", producto["precio"],
", ", "Cantidad:", producto["cantidad"])`
`return`

`print("Producto no encontrado")` return es como decir: Para aquí.

Método 4:

Actualizar stock

```
- stock(self, nombre, cantidad):
```

→ Este método lo vamos a usar para sumar o restar unidades a un producto que ya existe.

```
for producto in self.inventario:
```

if producto["nombre"] == nombre:

producto["cantidad"] = producto["cantidad"] + cantidad → Así actualizamos el stock

return → Cuando encuentra el producto y lo actualiza, hacemos el stock para el método

print("Producto no encontrado")

Método 5:

Eliminar producto

```
eliminar_producto(self, nombre):
```

for producto in self.inventario: → Lo usaremos para borrar productos de la tienda

if producto["nombre"] == nombre:

self.inventario.remove(producto) → remove es para quitar el producto de la lista del inventario

return → Por el método cuando lo encuentras

print("Producto no encontrado") → Solo lo mostraremos si el for termina y no encuentra nada

Método 6:

Calcular el valor del inventario

```
valor_inventario(self)
```

valor_total = 0 → Creo la variable para ir guardando la suma total

for producto in self.inventario: → Recorro la lista de inventario (cada producto es un dic)

precio = producto["precio"] → copio el precio y lo guardo en una variable

cantidad = producto["cantidad"] → copio la cantidad y la guardo en una variable

valor_producto = precio * cantidad → Para hacer el cálculo

valor_total = valor_total + valor_producto → sumo el valor a lo que ya tengo

return valor_total → Me devuelve el resultado aún estando fuera del método