

RESPUESTAS DEL CUESTIONARIO

1. ¿Qué estructura condicional utilizaste para validar que el ID del producto sea único? Explica el proceso de validación.

El ID de los productos se maneja mediante un ID global que comienza en el valor de ID del último producto en inventario. Este ID global aumenta en 1 cada que se agrega un nuevo producto, y se asigna a ese nuevo producto. Al eliminar productos el ID no se modifica, sigue en la última posición donde se quedó.

2. En el sistema de ventas, ¿cómo implementaste la condición para verificar si hay suficiente stock antes de realizar una venta?

```
for id_p , p in inv.items():
    if id_p == id_prod:
        if p["stock"] < 1:
            print(f"{AMARILLO}No hay suficiente inventario para agregar
...{BLANCO}")
```

Ya que verifiqué que el producto existiera en inventario, busco que el ID del producto a agregar coincida con el ID en inventario. Una vez obtenido, reviso si el atributo "stock" del producto en inventario sea mayor a 0, caso contrario se activará una alerta.

3. Proporciona un ejemplo de uso de elif en tu código y explica por qué fue necesario.

```
if opt == 1:
    inv = ordenar_inventario(inv, "nombre")
elif opt == 2:
    inv = ordenar_inventario(inv, "nombre", True)
elif opt == 3:
    inv = ordenar_inventario(inv, "proveedor")
elif opt == 4:
    inv = ordenar_inventario(inv, "proveedor", True)
elif opt == 5:
    inv = ordenar_inventario(inv, "categoria")
...
```

Durante el ordenamiento de inventario, usé una cantidad grande de elif, pero es necesario ya que se necesita una acción distinta a cada una de las opciones de ordenamiento.

4. ¿Qué tipo de ciclo utilizaste para el menú principal y por qué elegiste ese tipo?

El ciclo es un **while True:**. Elegí este tipo de ciclo porque no necesita un valor inicial, ni tampoco requiere de un valor de salida, el ciclo se rompe hasta que el usuario lo decida.

5. Explica la diferencia entre usar *for* y *while* en el contexto de tu proyecto. Da un ejemplo de cada uno.

For se utiliza durante todo el proyecto para recorrer los productos del inventario y del carrito de compras, así como para imprimir en consola los productos

```
for prod in carr:
    imprimir_producto_carrito(prod, c_aux)
```

While se utiliza cuando se necesita que el ciclo dure una cantidad indefinida de tiempo, es el ciclo que se utiliza durante la ejecución de la mayoría de los menús. O también en la verificación de entradas, donde la función no se interrumpe hasta que se ingrese un valor correcto.

```
while True:
    if mensaje_entrada != "":
        print(mensaje_entrada)

    cad = input("> ")

    if cad.strip() == "":
        print(f"{ROJO}>>> E405: Se necesita ingresar un valor
...{BLANCO}")
        continue

    return cad
```

6. ¿Cómo implementaste la búsqueda de productos usando ciclos? ¿Qué pasaría si tuvieras 10,000 productos?

```
for id_prod, prod in inv.items():
    if texto in prod[criterio].lower():
        productos.append((id_prod, prod))
```

Lo que hago es hacer una lista auxiliar, donde voy guardando cada producto en inventario que coincida con el criterio de búsqueda. Es un ciclo muy simple, así que, en caso de 10,000 productos, solo podría tardar un poco más dependiendo del dispositivo donde se ejecuta el código.

7. Lista las cinco funciones más importantes de tu sistema y describe brevemente qué hace cada una.

```
def ver_productos_inventario(inv,titulo=True):
```

Muestra todos los productos en inventario

```
def detectar_valor_entero_positivo(mensaje_entrada,valor_maximo = 0):
```

Garantiza que se elijan opciones válidas en lo menús

```
def detectar_existencia_inventario(inv,id_prod):
```

Garantiza que el producto con el ID especificado sí exista en inventario

```
def finalizar_compra(carr,c_aux):
```

Muestra el total de compra y genera los tickets de compra

```
def menu_producto(prod):
```

Permite la modificación y eliminación de productos

8. ¿Por qué es importante usar funciones en lugar de escribir todo el código en el programa principal?

El uso de funciones permite tener un mejor control de la ejecución del programa, para poder reutilizar un misma en diferentes partes del código y para detectar más fácilmente errores de ejecución. También le da un buen apartado visual.

9. Describe una función que reciba parámetros y retorne un valor. ¿Qué ventaja tiene usar return?

```
def obtener_mayor_id(inv):  
    if not inv: # si está vacío  
        return -1  
    return max(inv.keys())+1
```

Tiene la ventaja de poder hacer operaciones o modificar valores dentro de la función, y regresar un valor específico al lugar donde fue llamada la función.

10. ¿Qué estructura de datos elegiste para almacenar los productos (lista de diccionarios, diccionario de diccionarios, etc.)? Justifica tu elección

Elegí diccionarios de diccionarios porque permite asignar una llave-valor que se puede usar como ID-producto para garantizar que no existan productos con un mismo ID y facilitar la obtención de productos desde el inventario.

11. Explica cómo agregaste un nuevo producto al inventario usando operaciones con arreglos.

```
inv[identificador] = {  
    "ID": identificador,  
    "nombre": nombre,  
    "categoria": categoria,  
    "precio": precio,  
    "proveedor": proveedor,  
    "stock": stock  
}  
  
identificador += 1
```

Después de validar todas las entradas del usuario para cada atributo del nuevo producto, hago referencia al inventario con el identificador global, el cual siempre tendrá el valor del siguiente ID libre para productos nuevos. Al hacer referencia a una llave que no existe en el diccionario, Python simplemente crea un nuevo valor con esa llave y los datos del valor. Al final aumento en 1 el identificador global para que siga manteniendo su propósito.

12. ¿Cómo implementaste la eliminación de un producto? ¿Qué método de listas o diccionarios utilizaste?

```
def menu_producto(prod):  
    ...  
    opt = input(f"{AZUL}¿Seguro desea eliminar {prod['nombre']} del  
    inventario? [1) Si | Otro) No] {BLANCO}\n> ")  
    if opt == "1":  
        inventario.pop(prod["ID"])  
        print(f"{VERDE} ; Producto eliminado ! {BLANCO}\n")
```

Primero hago uso de verificadores para garantizar que el producto exista en inventario, y guardo el producto en una variable. Dentro del menú del producto, utilizo la opción de eliminar producto. Para eliminar el producto llamo al inventario global (es la única vez que se utiliza el inventario de manera global, y es solo para ahorrar unas líneas de código), y hago uso de la función `.pop()` del diccionario, y el parámetro que utilizo lo consigo del atributo "ID" del mismo producto que guarde.

13. ¿Cuál fue el mayor desafío técnico que enfrentaste durante el desarrollo y cómo lo resolviste?

Lograr que los filtros se pudieran utilizar sobre otros filtros, después de un tiempo de prueba y error intenté utilizar recursividad para llamar a la misma función, pero con el inventario ya filtrado, y gracias a la modularidad de Python logré que se pudiera operar sobre los productos ya filtrados.

14. Si tuvieras que agregar una nueva funcionalidad (por ejemplo, sistema de descuentos), ¿cómo la integrarías al código existente?

Pudiera ser una función para descuento para compras por mayoreo en ciertos productos. Constaría en verificar si el producto es apto para descuento por mayoreo, cual es la cantidad mínima para comenzar a aplicar el descuento, y cuáles son los porcentajes asignados dependiendo de la cantidad de productos comprados.

15. ¿Qué aprendiste sobre trabajo en equipo durante este proyecto? ¿Cómo se organizaron para evitar conflictos en el código?

Que el trabajo en equipo es una muy buena práctica en proyectos grande, ya que disminuye la carga de trabajo al dividir por partes el proyecto, y a su vez ayuda a la improvisación y integración de nuevas ideas al haber distintos puntos de vista. Para evitar conflictos se repartieron las áreas a trabajar desde un inicio, y una vez completadas dos áreas que fueran complementarias, se unieron con cautela para detectar errores o incoherencias en la ejecución.