Universidad San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ingeniería en Ciencias y Sistemas

Laboratorio Lenguajes Formales Y De Programación, Sección B+

Ing. David Estuardo Morales Ajcot

Auxiliar: Francisco Magdiel Asicona Mateo

Segundo Semestre 2023

Manual Técnico

# Gestión de Inventario y Registro de Movimiento de Productos

Nombre: Carlos Manuel Lima y Lima

Registro Académico: 202201524

CUI: 3009368850101

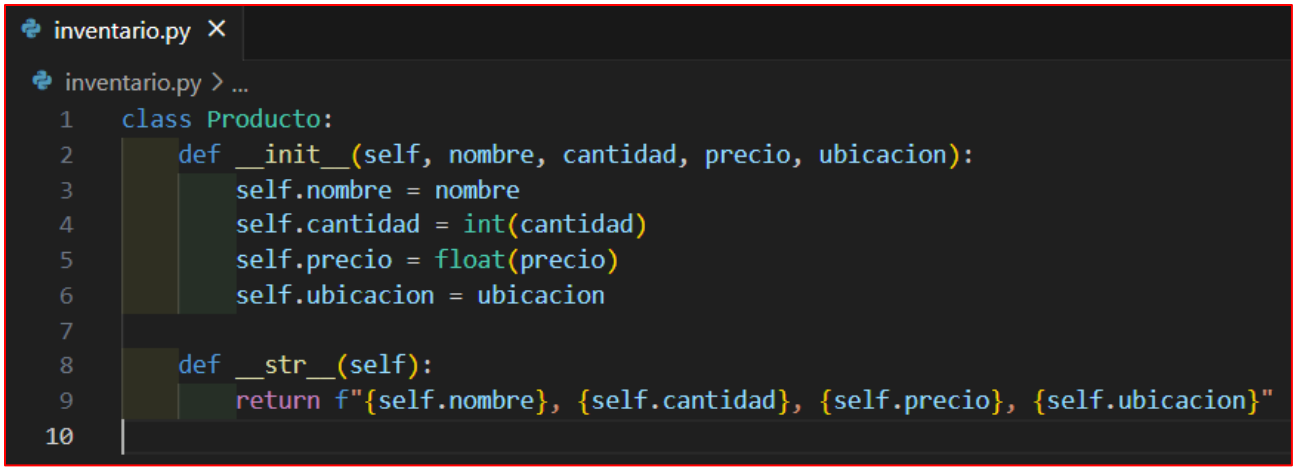
Guatemala, 13 de agosto del 2023.

# Inventario.py

- ***Clase Producto***
  La clase producto contiene 2 funciones:

  La primera función *init* define todos los atributos que tiene un producto, los cuales son nombre, cantidad, precio, ubicación. En dicha función se declara que los valores definidos del objeto producto serán iguales a los valores que se le pasan.

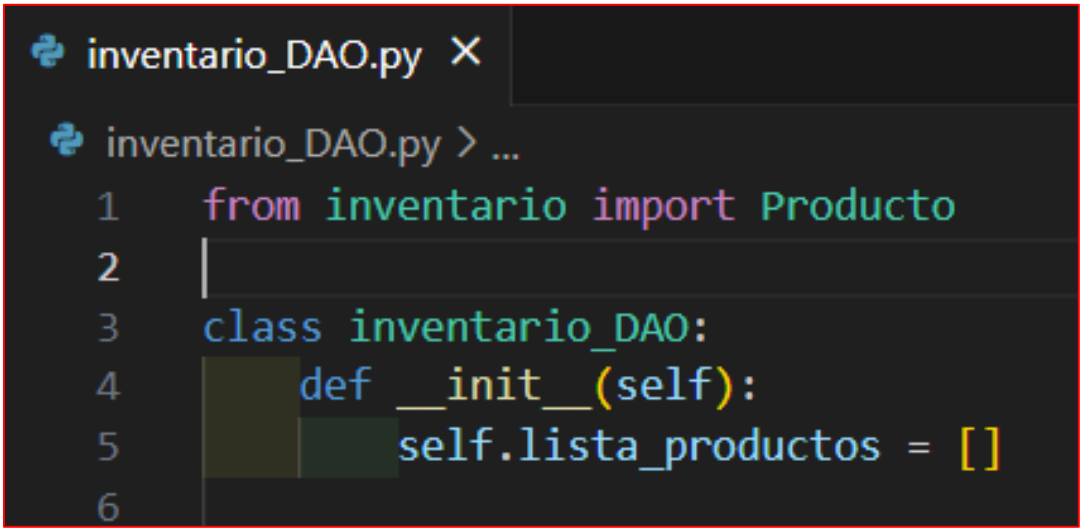
  La segunda función *str* es un método especial en Python que permite definir una representación legible en forma de cadena del objeto producto de una clase Producto.



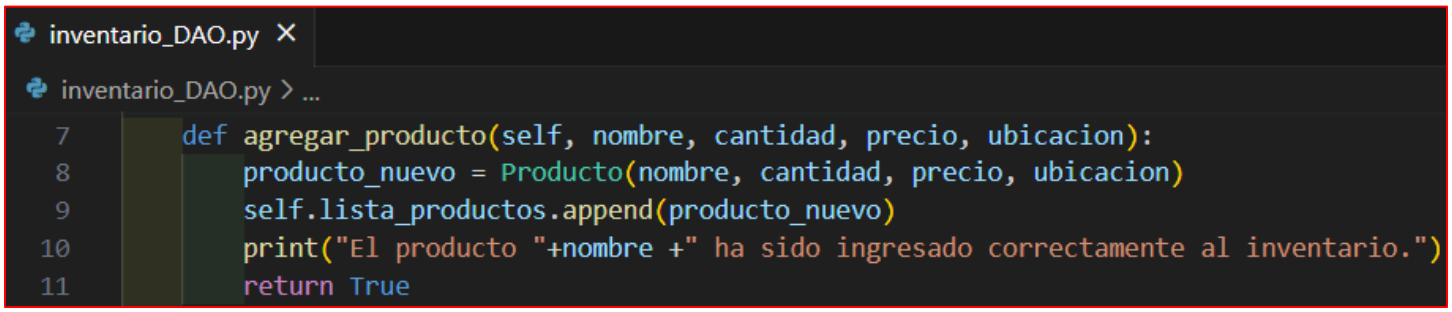

# Inventario_DAO.py

- ***Clase Inventario DAO***
  Esta clase contiene todas los métodos de la clase Producto. Para acceder a la clase Producto se debe importar dicha clase desde el archivo inventario.

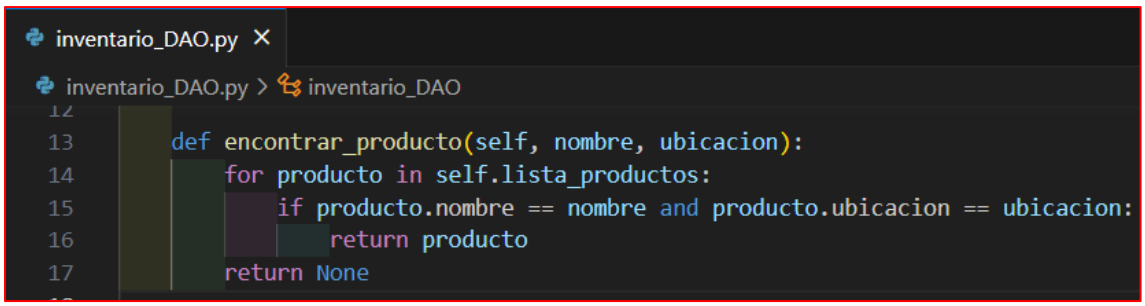
  La función *init* declara una lista con el nombre lista productos, de esta forma la lista será global y todos los cambios que se podrían producir en un objeto se podrán realizar desde cualquier función.



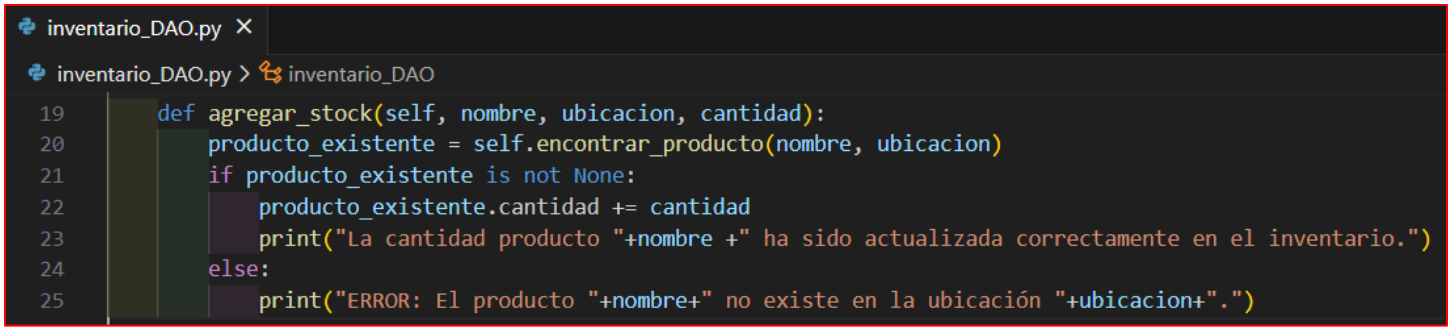
- ***Función Agregar Producto***
  Esta función se declara con los parámetros de nombre, cantidad, precio y ubicación. Lo primero que realiza esta función es crear un nuevo producto el cual será objeto igual al objeto Producto con los mismos atributos. Luego se agrega el nuevo producto a la lista productos con el método append. Finalmente se muestra un mensaje de confirmación en consola y se devuelve una variable booleana true.




- ***Función Encontrar Producto***
  La función encontrar producto recorre la lista de productos de forma ordenada. Luego evalúa si el nombre y la ubicación del producto actual de la lista es igual al nombre y ubicación que la función está pasando. Si la condición se cumple durante la iteración se devolverá el producto que coincida. Si al finalizar la interacción no se cumple la condición, la función devolverá None.




- ***Función Agregar Stock***
  La función agregar producto primero evalúa si existe un producto en la lista con el mismo nombre y ubicación, para esto se crea un objeto producto existente que será igual la función encontrar producto anteriormente declarada a la cual se le pasan los parámetros de nombre y ubicación. Si el producto existente no es nulo, se actualiza la cantidad sumando la cantidad actual con la nueva cantidad. Si el producto es nulo, es decir que no existen coincidencias durante la iteración, se devuelve un mensaje de error.

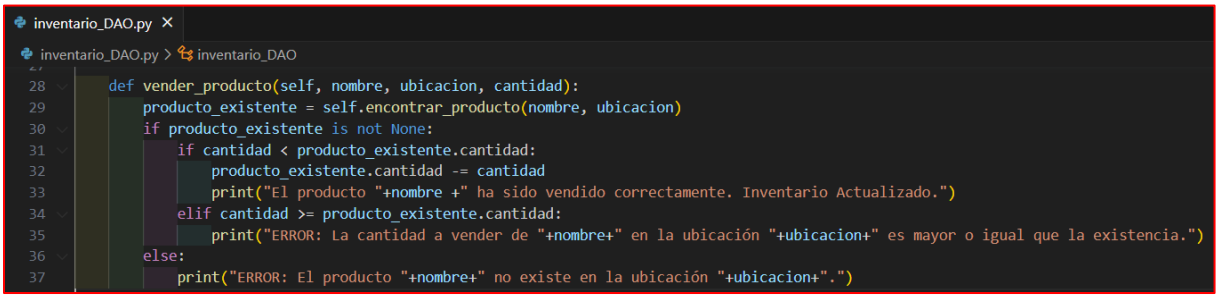


- ***Función Vender Producto***

  La función vender primero evalúa si existe un producto en la lista con el mismo nombre y ubicación, para esto se crea un objeto producto existente que será igual la función encontrar producto anteriormente declarada a la cual se le pasan los parámetros de nombre y ubicación.

  Si el producto existe, es decir que el producto no es nulo, se procede a realizar las siguientes validaciones:
  Si la cantidad a vender es menor a la cantidad del producto existente, se actualiza la cantidad restando la cantidad del producto existente de la cantidad a vender y se muestra un mensaje en consola. Si la cantidad a vender es mayor o igual a la cantidad del producto existe, se muestra un mensaje error.

  Si el producto no existe, se muestra un mensaje de error.




- ***Función Crear Informe***

  Esta función utiliza una estructura try-catch para crear un archivo con un nombre archivo, el cual será el parámetro que se le pasará a la función.

  En el try, se utiliza el with open, el cual abre y cierra automáticamente el archivo. Se utiliza del modo de apertura *w* el cual abre el archivo en modo de escritura, además crea el archivo si no existe o sobrescribe su contenido. Luego de esto se crea una variable linea1 la cual contiene un string junto con 2 saltos de línea y se procede a escribir dicha variable con el método write. Luego se procede a recorrer la lista de productos de forma secuencial, y se crea una variable valor total la cual será igual a la multiplicación de la cantidad del producto por el precio del producto. Posteriormente se crea una variable linea la cual será igual a un string que contendrá todos los atributos del producto (nombre, cantidad, precio, valor total, ubicación) y un salto de línea. Finalmente se escribe la variable linea en el archivo con el método write. Al finalizar la iteración de la lista productos, se muestra e un mensaje en pantalla.

  En el catch se muestra un mensaje de error y la excepción del error. Esto sin caso se produjera un error.

- ***Función Imprimir Productos***
  En esta función, se imprime la lista del productos con un for que recorrerá la lista de forma iterada.

```python
# inventario_DAO.py > inventario_DAO
    def imprimir_productos(self):
        for producto in self.lista_productos:
            print(producto)
```

## Main.py

Este archivo será el principal, el cual contendrá el menú principal del programa. Para acceder a todas las funciones del inventario, se importa del archivo inventario DAO la clase inventario DAO. Y se crea una variable manejador lista, la cual servirá para llamar a todas las funciones.

```python
# main.py > ...
from inventario_DAO import inventario_DAO
manejador_lista = inventario_DAO()
```

- ***Función Menú Principal***
  La función menú principal imprime en consola todas las opciones a las cuales se puede acceder. Se declara una variable opción la cual pide al usuario que ingrese una opción. Con una estructura if-else se valida la opción que el usuario ingresó y se mandan a llamar a las funciones. Si la opción no es válida, se vuelve a llamar la función menú principal.

```python
# main.py > crear_informe_inventario
def menu_principal():
    print("-------------------------------------------------")
    print("Menú Principal - Sistema de Inventario          ")
    print("-------------------------------------------------")
    print("1. Cargar Inventario Inicial                    ")
    print("2. Cargar Instrucciones de Movimiento           ")
    print("3. Crear Informe de Inventario                  ")
    print("4. Salir                                        ")
    print("-------------------------------------------------")
    opcion = input("Ingrese Una Opción: ")
    if opcion == "1":
        cargar_inventario_inicial()
    elif  opcion == "2":
        cargar_instrucciones_movimiento()
    elif opcion == "3":
        crear_informe_inventario()
    elif opcion == "4":
        salir()
    else:
        print("-------------------------------------------------")
        print("OPCIÓN NO VÁLIDA")
        menu_principal()
```

- ***Función Cargar Inventario Inicial***

  Esta función primero guarda la ruta del archivo a cargar en una variable llamada ruta. Luego por medio de un try-catch accede al archivo.

  En el try, se utiliza el with open, el cual abre y cierra automáticamente el archivo. Se utiliza del modo de apertura *r* el cual abre el archivo en modo de lectura, además genera un error si el archivo no existe.

  Luego por medio de un for se recorre todas las líneas del archivo en modo lectura. Se crea la lista llamada palabras que será igual a la línea del archivo.

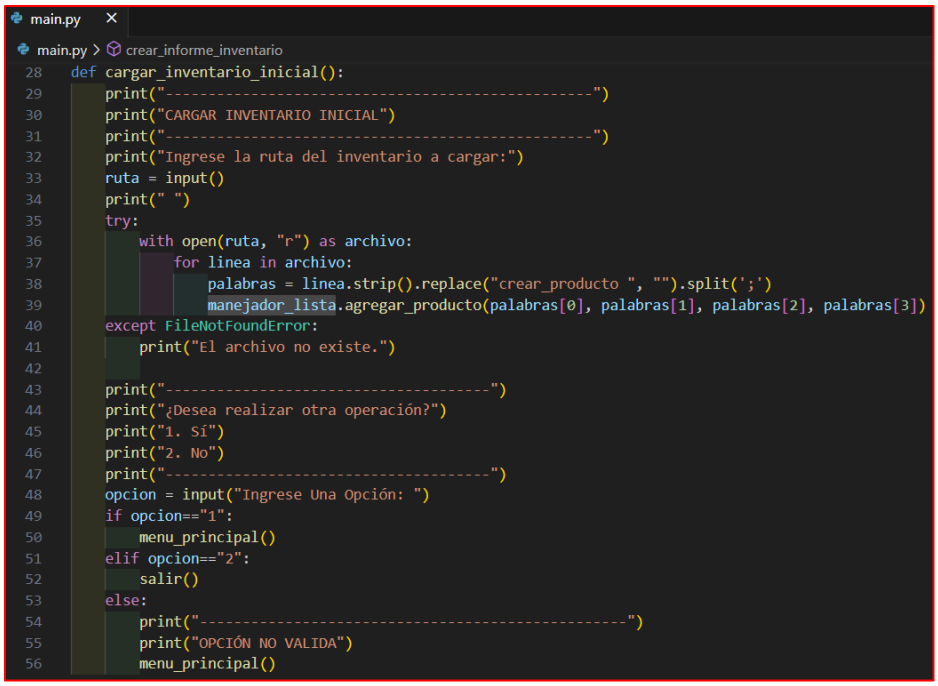
  El método .strip() se utiliza para eliminar los espacios en blanco y caracteres de nueva línea al principio y al final de la línea del archivo.

  El método .replace() se utiliza para eliminar la palabra crear_producto con todo y el espacio en blanco que está después de dicha palabra, además crea una cadena con todos los caracteres que están después de la palabra eliminada.

  El método split() se utiliza para dividir una cadena en una lista de subcadenas utilizando el delimitador “punto y coma” y esto crea una lista con todas las partes de la línea que estaban separadas por dicho delimitador. Por lo cual la lista palabras tendrá los valores que fueron separados en esta parte.

  Luego de esto, se manda a llamar a la función agregar producto por medio del manejador lista. A la función se le pasan los parámetros de nombre, cantidad, precio unitario y ubicación. Esto porque la lista llamada palabra contiene dichos valores en ese orden.

  Por último, se imprime un menú final para preguntarle al usuario si desea realizar otra operación y se guarda la respuesta del usuario en la variable opción. Con la estructura if-else se validan las opciones según sea el caso.



- ***Función Cargar Instrucciones de Movimiento***

  Esta función primero guarda la ruta del archivo a cargar en una variable llamada ruta. Luego por medio de un try-catch accede al archivo.

  En el try, se utiliza el with open, el cual abre y cierra automáticamente el archivo. Se utiliza del modo de apertura *r* el cual abre el archivo en modo de lectura, además genera un error si el archivo no existe.

  Luego por medio de un for se recorre todas las líneas del archivo en modo lectura. Se crea la lista llamada palabras que será igual a la línea del archivo.

  El método strip() se utiliza para eliminar los espacios en blanco y caracteres de nueva línea al principio y al final de la línea del archivo.

  El método split() se utiliza para dividir una cadena en una lista de subcadenas utilizando el delimitador “espacio en blanco” y esto crea una lista con todas las partes de la línea que estaban separadas por dicho delimitador. Por lo cual la lista palabras tendrá los valores que fueron separados en esta parte.

  Después se vuelve utilizar de nuevo el método split(), pero en este caso con el delimitador “punto y coma”, esto será para el segundo valor de la lista palabras. Y esta nueva lista que se generará se guarda en la lista datos producto.

  Se valida si el primer valor de la lista palabras es igual a “agregar stock” o “vender producto”. Según sea la coincidencia se manda a llamar a la función con el mismo nombre y se pasan los valores de nombre, ubicación y cantidad que están almacenados en la lista datos productos.

  Por último, se imprime un menú final para preguntarle al usuario si desea realizar otra operación y se guarda la respuesta del usuario en la variable opción. Con la estructura if-else se validan las opciones según sea el caso.

main.py

main.py > crear_informe_inventario

```
59  def cargar_instrucciones_movimiento():
60      print("-------------------------------------------------")
61      print("CARGAR INSTRUCCIONES DE MOVIMIENTO")
62      print("-------------------------------------------------")
63      print("Ingrese la ruta de las instrucciones a cargar:")
64      ruta = input()
65      print(" ")
66      try:
67          with open(ruta, "r") as archivo:
68              for linea in archivo:
69                  palabras = linea.strip().split(' ')
70                  instruccion = palabras[0]
71                  datos_producto = palabras[1].split(';')
72                  if instruccion == "agregar_stock":
73                      manejador_lista.agregar_stock(datos_producto [0], datos_producto[2], int(datos_producto[1]))
74                  elif instruccion == "vender_producto":
75                      manejador_lista.vender_producto(datos_producto [0], datos_producto[2], int(datos_producto[1]))
76      except FileNotFoundError:
77          print("El archivo no existe.")
78
79      print("-------------------------------------")
80      print("¿Desea realizar otra operación?")
81      print("1. Sí")
82      print("2. No")
83      print("-------------------------------------")
84      opcion = input("Ingrese Una Opción: ")
85      if opcion=="1":
86          menu_principal()
87      elif opcion=="2":
88          salir()
89      else:
90          print("-------------------------------------------------")
91          print("OPCIÓN NO VALIDA")
92          menu_principal()
```

- ***Función Crear Informe de Inventario***
  Esta función solicita el nombre con el cual se guardará el archivo y lo guarda en una variable llamada nombre. Luego por medio del manejador lista, se manda a llamar a la función crear informe a la cual se le manda la variable nombre concatenado con “.txt”, esto generará un archivo txt.

  Por último, se imprime un menú final para preguntarle al usuario si desea realizar otra operación y se guarda la respuesta del usuario en la variable opción. Con la estructura if-else se validan las opciones según sea el caso.