Practica 1

Manual técnico

Lenguajes formales y de programación

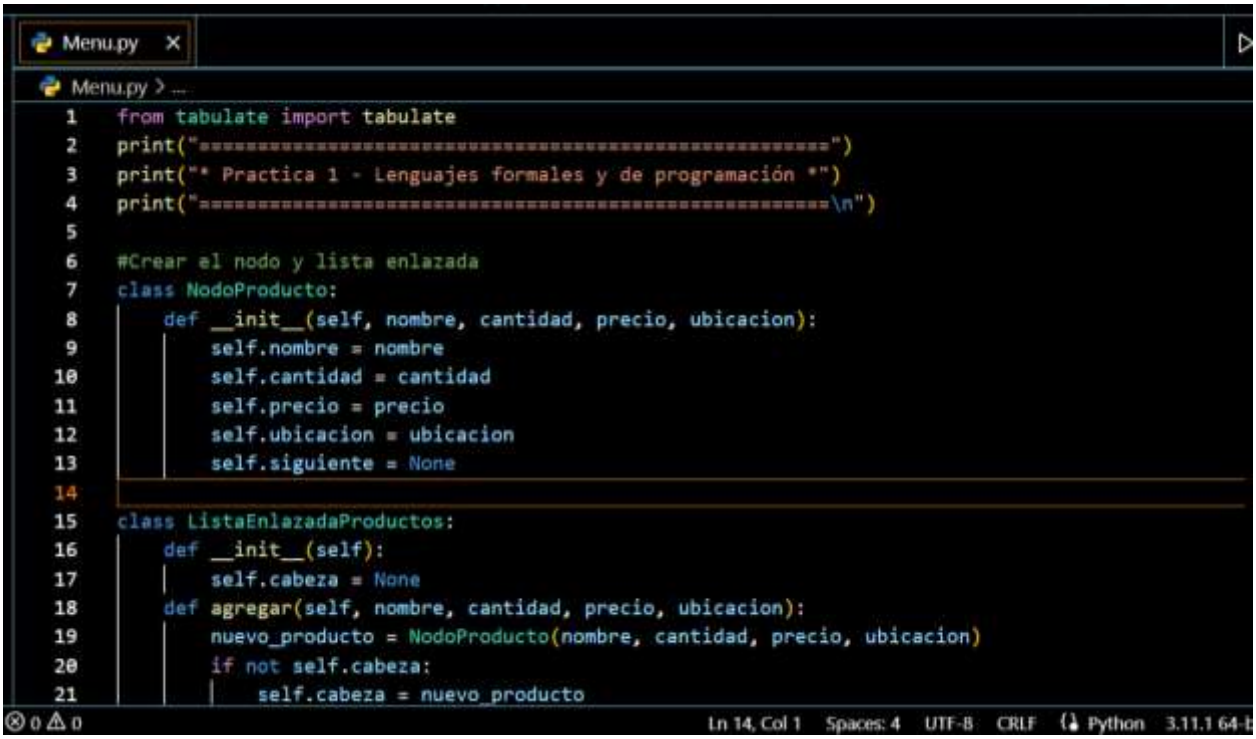
Isamir Alessandro Armas Cano
201901403

Manual tecnico

Este manual explica la logica del programa de python el cual tiene como fin una solución de software con base en los distintos paradigmas de programación vistos en clase y laboratorio. Y donde se utiliza el manejo de archivos, lógica de programación y manipulación de estructuras de datos en Python.

Para iniciar se establecio que se trabajaria con listas enlazadas para poder manejar los datos de los productos que estarian en un archivo diferente el cual se deberia de leer y sacar cada producto con su nombre, cantidad, precio y su ubicación.

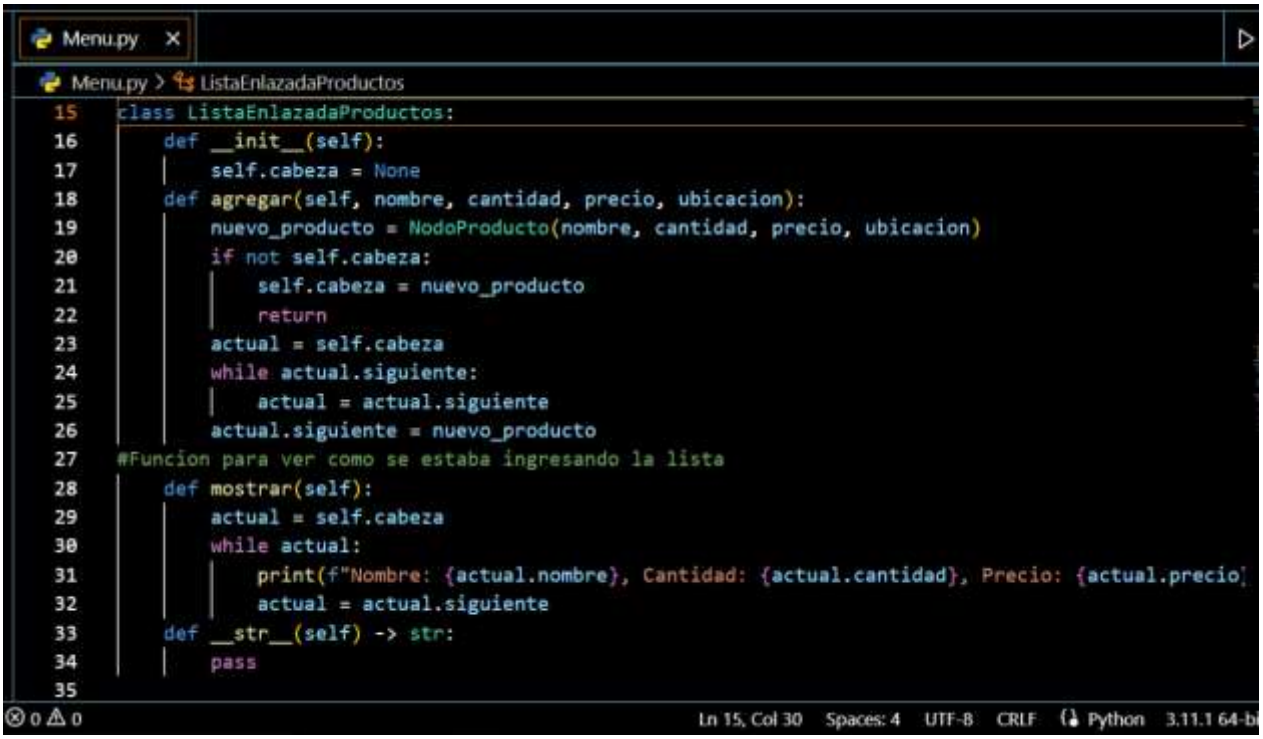
Para esto creamos un Nodo lo cual nos permite trabajar con listas enlazadas.
Se realizo de esta forma.



Se creo el NodoProducto con y se definio los datos que se utilizaran (nombre, cantidad, precio y ubicación)
Se define self.siguiente = none
Ya que es un argumento el cual se utiliza en las listas enlazadas.

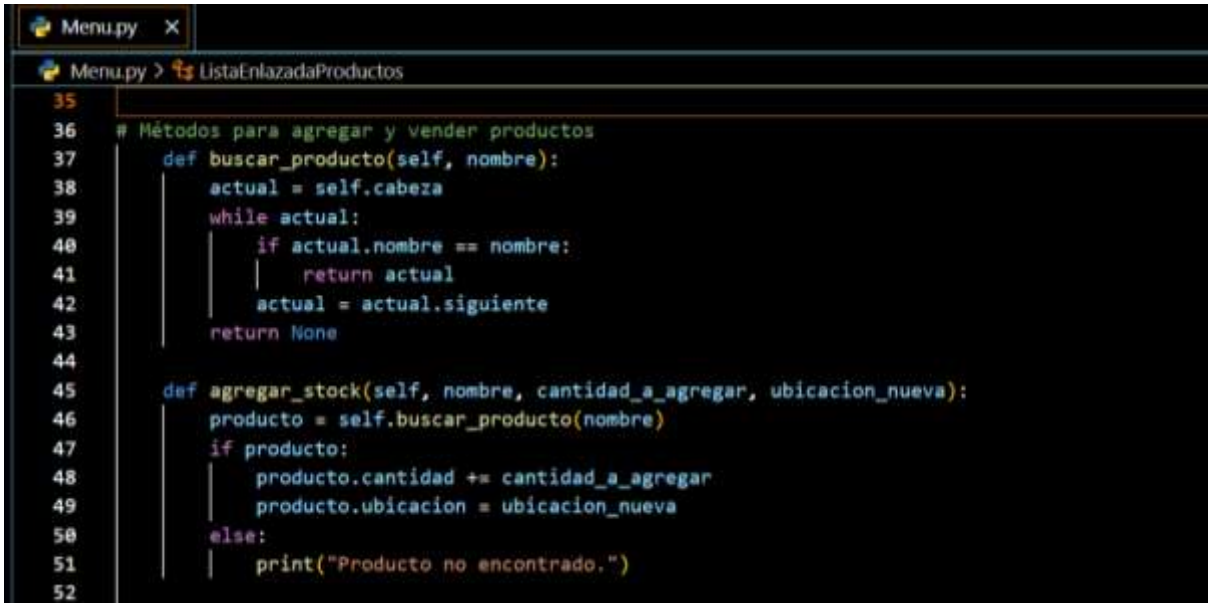
Luego definimos las funciones a utilizar para manipular nuestra lista los cuales son:

- ListaEnLazadaProductos: Con esta definimos la cabeza de el nodo
- Agregar: Esta funcion nos sirve para agregar datos en la lista.
- Mostrar: Esta funcion se utilizo para visualizar como se estaba colocando los productos en la lista.



Con esto obtenemos la lista enlazada a la cual ingresaremos los productos, a estos productos se le agregaran productos y se venderan productos de la misma. Entonces definimos las funciones para agregar productos y vender productos de la siguiente forma.

- Buscar_producto: Con esta definicion nos ubicamos en el lugar que ocupa el producto que se quiere modificar.
- Agregar_stock: Con esta agregamos productos validando que la cantidad que se agregara se sume a la que ya se encuentra en el inventario (lista enlazada).



- Vender_producto: Con esta funcion validamos que la cantidad a vender sea menor o igual a la cantidad en inventario y que la ubicación sea la misma.

```
            producto.cantidad += cantidad_a_agregar
            producto.ubicacion = ubicacion_nueva
        else:
            print("Producto no encontrado.")

    def vender_producto(self, nombre, cantidad_a_vender, verificar_ubi):
        producto = self.buscar_producto(nombre)
        if producto:
            if producto.cantidad >= cantidad_a_vender and producto.ubicacion == verificar_ubi:
                producto.cantidad = producto.cantidad -cantidad_a_vender
            else:
                print("No hay suficiente stock para la venta.")
        else:
            print("Producto no encontrado.")

```

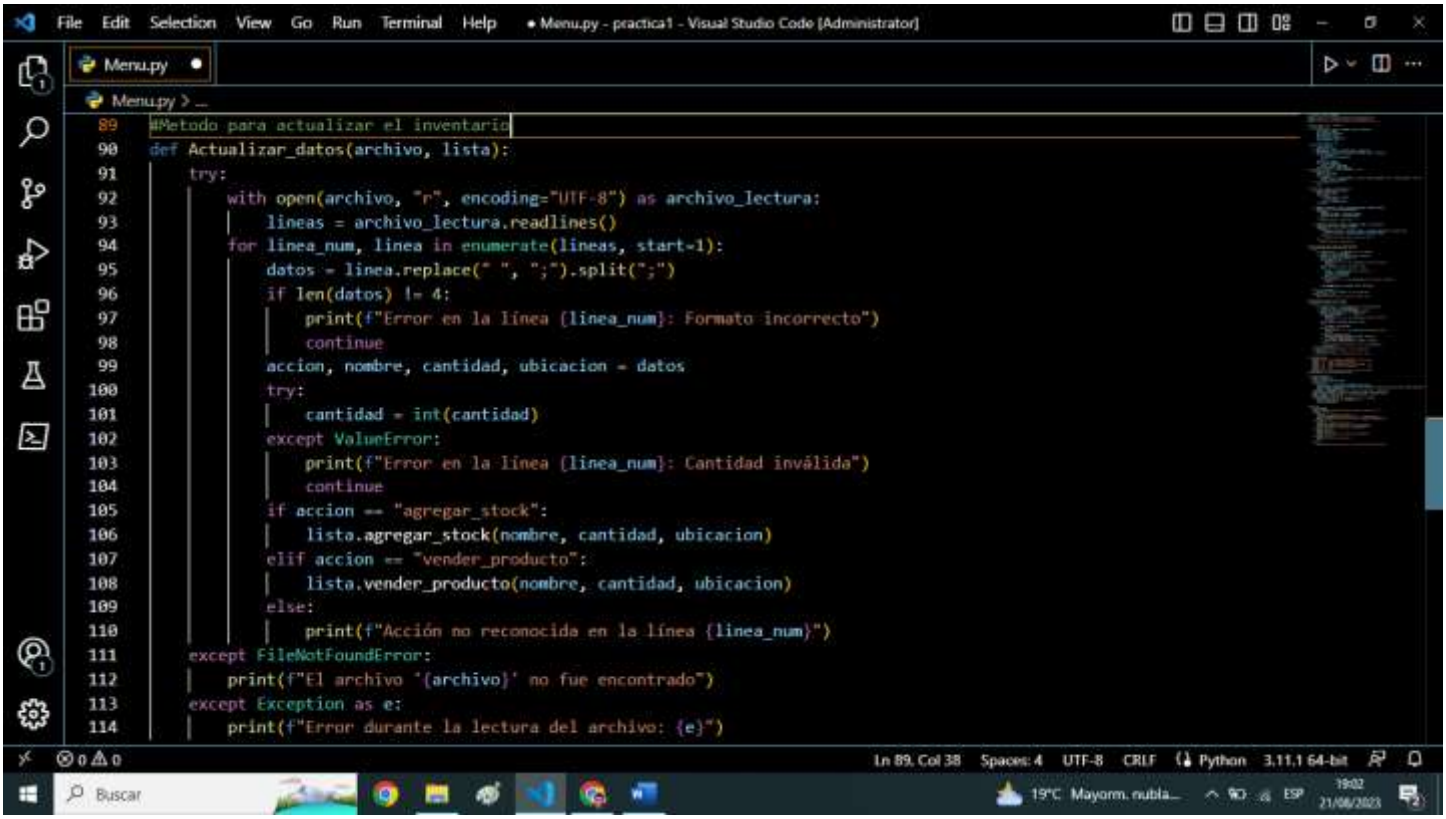
Ahora teniendo definidas las funciones que utilizaremos para agregar y vender se paso a leer el arcivo **.inv** para definir los productos que encontraremos en la lista enlazada.

Leer_datos_desde_archivo:

```
Menu.py
Menu.py > ListaEnlazadaProductos > agregar_stock

#Leer el archivo.inv y crear los primeros datos.
def leer_datos_desde_archivo(archivo, lista):
    try:
        with open(archivo, "r", encoding="UTF-8") as archivo:
            lineas = archivo.readlines()
        for linea_num, linea in enumerate(lineas, start=1):
            datos = linea.strip().split(";")
            if len(datos) != 4:
                print(f"Error en la línea {linea_num}: Formato incorrecto")
                continue
            nombre, cantidad, precio, ubicacion = datos
            try:
                nombre = str(nombre).strip("crear_producto ")
                cantidad = int(cantidad)
                precio = float(precio)
            except ValueError:
                print(f"Error en la línea {linea_num}: Datos numéricos inválidos")
                continue

            lista.agregar(nombre, cantidad, precio, ubicacion)

    except FileNotFoundError:
        print(f"El archivo '{archivo}' no fue encontrado")
    except Exception as e:
        print(f"Error durante la lectura del archivo: {e}")
```

Primero leemos el archivo, este lo leemos linea por linea y separamos cada palabra de esa linea con **.split(";").** Con esto partimos la linea en 4 partes de las cuales se obtiene nombre, cantidad, precio, ubicación.
La primera parte de cada producto nos queda como "crear_producto nombre".
Para obtener el nombre utilizamos .strip("crear_producto").
Con el **try** nos aseguramos que si hay errores en el archivo se pueda seguir corriendo el programa

Actualizar_datos:



Con esta funcion leemos el archivo **.mov** primero utilisamos el try nuevamente para asegurarnos de que el programa siga corriendo aunque el archivo tenga errores.

Leemos el archivo linea por linea y al igual que el anterior utilizamos el **replace(“ ”, “;”).strip(“;”)**
Para quitar el espacio y poner un “;” y poder dividir cada linea en 4 partes : accion, nombre, cantidad, ubicación para guardarlos en la variable datos

Ya que aquí tenemos 2 acciones diferentes las cuales son agregar_stock y vender_producto
Tenemos la parte accion la cual comparamos y utiliamos para actualizar nuestro inventario.
Despues de validar cual de las acciones se realizara utilizamos las funciones previamente definidas para trabajar con estos datos del archivo de movimientos.

Ahora definimos el menu que mostraremos al usuario:
Con este menu damos 4 opciones al usuario para que el pueda realizar lo que desee.

Luego creamos una lista con la cual trabajaremos nuestra lista enlazada
Esto es bastante importante para el funcionamiento de nuestro programa

Lista_productos = ListaEnlazadaProductos()

```
Menu.py > Generar_reporte
113         except Exception as e:
114             print(f"Error durante la lectura del archivo: {e}")
115
116     def mostrar_menu():
117         print("========== SISTEMA DE INVENTARIO ==========")
118         print("| |1|   Cargar inventario inicial           |")
119         print("| |2|   Cargar instrucciones de movimientos  |")
120         print("| |3|   Crear informe de inventario          |")
121         print("| |4|   Salir                                |")
122         print("==========================================")
123     lista_productos = ListaEnlazadaProductos()
124
```

Definimos la funcion para Generar_reporte
Creamos la lista la cual utilizaremos para sacar los datos
Recorremos la lista enlazada para llenar nuestra lista datos_tabulados
Vemos que realizamos una consrante precio_total, esta nos sirve para obtener el valor total multiplicando la cantidad y el precio de cada producto

Para crear la tabla utilizamos la librería **Tabulate** la cual importamos al inicio del codigo
Esta nos ayuda para crear una tabla brindando el nombre de las columnas en la constante **encabezados**
Luego definimos el nombre del archivo que generaremos
Ahora escribimos la tabla en el archivo creado.

```
123 lista_productos = ListaEnlazadaProductos()
124
125 def Generar_reporte():
126     datos_tabulados = []
127     actual = lista_productos.cabeza
128     while actual:
129         precio_total = float(actual.cantidad*actual.precio)
130         datos_tabulados.append([actual.nombre, actual.cantidad, actual.precio,precio_total, actu
131         actual = actual.siguiente
132     encabezados = ["NOMBRE", "CANTIDAD", "PRECIO","VALOR TOTAL", "UBICACIÓN"]
133     tabla = tabulate(datos_tabulados, headers=encabezados, tablefmt="grid")
134     nombre_archivo = "Resultados_201901403.txt"
135     with open(nombre_archivo, "w", encoding="UTF-8") as archivo:
136         archivo.write("Informe del Inventario: \n")
137         archivo.write(tabla)
138
139 while True:
140     mostrar_menu()
141     opcion = input("Seleccione una opción (1/2/3) o 4 para salir: ")
```

Por ultimo creamos un while el cual nos sirve para brindarle al usuario la opcion que escoja y si esta es una diferente a la que le ofrecemos volver a preguntarle
Si elige la opcion 1 : Esta lee los datos del archivo . inv y los ingresa en nuestra lista_productos para poder trabajar con estos datos

Si elige la opcion 2 : Esta opcion lee los movimientos del archivo .mov y actualiza la lista_productos.

Si elige la opcion 3 : Esta opcion genera un archivo Reporte_201901403 con una tabla la cual tiene registrado los ultimos movimientos realizados en la lista_productos.

Si elige la opcion 4 : Esta opcion cierra el programa.