

Python Inicial

Jose Miguel Paz Portilla

Proyecto Final

Inventario de Productos

12 de julio de 2025

Índice

1	Introducción	3
2	Implementación	3
3	Pruebas	5
3.1	Opción fuera del rango	5
3.2	Campos del producto	5
3.3	Busqueda de producto por nombre	5
3.4	Eliminación de producto por posición	6
4	Código	6
4.1	makefile	6
4.2	main.py	7
4.3	menu.py	8
4.4	opciones.py	9
4.5	metodos_productos.py	9
5	Conclusión	13

1 Introducción

Con el objetivo de realizar un inventario de productos con persistencia de información, se desarrollo un programa escrito en python que cumple los siguientes requisitos:

1. Usar listas para almacenar y gestionar los datos.
2. Incorporar bucles while y for según corresponda.
3. Validar entradas del usuario o usuaria, asegurándote de que no se ingresen datos vacíos o incorrectos.
4. Utilizar condicionales para gestionar las opciones del menú y las validaciones necesarias.
5. Presentar un menú que permita elegir entre las funcionalidades disponibles: agregar productos, visualizar productos, buscar productos y eliminar productos.
6. El programa debe continuar funcionando hasta que se elija una opción para salir.
7. El programa persiste la información en un archivo del disco rígido.

2 Implementación

Utilizo la función main como función principal del programa, el cual muestra un menu con opciones al usuario, y según la opcion elegida realiza alguno de los requisitos solicitados.

Para correr el programa desde una terminal ubicada donde se encuentran los archivos de python y el archivo makefile ingresar *make*, como muestra la Figura 1

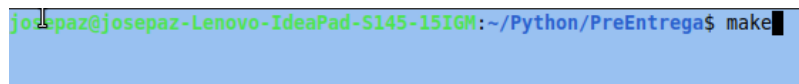


Figura 1: Como correr el programa desde terminal

El programa refresca la terminal y muestra un menu con opciones, se le solicita al usuario que ingrese un número del menu entre las disponibles, como se muestra en la Figura 2.

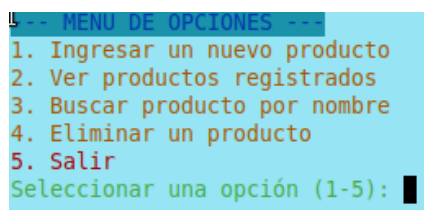


Figura 2: Mostrando el menu

Al ingresar la opción 1, de solicita el nombre, categoria y precio del nuevo producto para almacenarlo en la lista productos, como se muestra en la Figura 3.

```
--- NUEVO PRODUCTO ---  
Ingrese el nombre del producto: zapato  
Ingrese la categoría del producto: A  
Ingrese el precio del producto (sin centavos): 1500  
Presione ENTER para continuar
```

Figura 3: Ingreso nuevo producto

Posteriormente al ingresar la opción 2, se lista los productos en el inventario, como se muestra en la Figura 4.

```
--- LISTA DE PRODUCTOS REGISTRADOS ---  
1. Nombre: zapato | Categoría: A | Precio: $1500  
Presione ENTER para continuar
```

Figura 4: Mostrar productos

Para buscar un producto por su nombre se ingresa la opción 3, el programa busca en la lista del inventario y va almacenando en una lista auxiliar todos los productos que coinciden con el nombre del producto, y lo muestra por la terminal, como se muestra en la Figura 5.

```
--- BUSQUEDA DE PRODUCTO ---  
Ingresar el nombre del producto a buscar: zapato  
1. Nombre: zapato | Categoría: A | Precio: $1500  
Presione ENTER para continuar
```

Figura 5: Buscar producto por nombre

Con la opción 4, se puede eliminar un producto con la posición en la lista mostrada por terminal, como se muestra en Figura 6.

```
--- ELIMINACION DE PRODUCTO ---  
--- LISTA DE PRODUCTOS REGISTRADOS ---  
1. Nombre: zapato | Categoría: A | Precio: $1500  
Ingrese el número del producto a eliminar: 1  
Presione ENTER para continuar
```

Figura 6: Eliminar producto por índice

Finalmente para finalizar el programa se ingresa la opción 5, y se muestra por terminal un mensaje de finalización, como en la Figura 7.

```
I  
¡PROGRAMA FINALIZADO!  
josepaz@josepaz-Lenovo-IdeaPad-S145-15IGM:~/Python/PreEntrega$
```

Figura 7: Finalización del programa

3 Pruebas

3.1 Opción fuera del rango

Como se observa en la Figura 8, al ingresar opciones fuera de las opciones del menu el programa evita que se ejecuten.

```
-- MENÚ DE OPCIONES ---
1. Ingresar un nuevo producto
2. Ver productos registrados
3. Buscar producto por nombre
4. Eliminar un producto
5. Salir
Seleccionar una opción (1-5): 0
ERROR: OPCIÓN FUERA DEL RANGO
Presione ENTER para continuar
```

(a) Error por opción por debajo del rango

```
--- MENÚ DE OPCIONES ---
1. Ingresar un nuevo producto
2. Ver productos registrados
3. Buscar producto por nombre
4. Eliminar un producto
5. Salir
Seleccionar una opción (1-5): 6
ERROR: OPCIÓN FUERA DEL RANGO
Presione ENTER para continuar
```

(b) Error por opción por encima del rango

Figura 8: Errores por elegir opciones fuera del rango

3.2 Campos del producto

Como se observa en la Figura 9, al ingresar campos vacios del nombre o categoria del producto o bien si se ingresa valores negativos o reales al precio del producto, el sistema no permite continuar con la ejecución del programa.

```
--- NUEVO PRODUCTO ---
Ingrese el nombre del producto:
ERROR: NOMBRE DE PRODUCTO ESTA VACIA
Ingrese el nombre del producto: pera
Ingrese la categoría del producto:
ERROR: NOMBRE DE LA CATEGORIA ESTA VACIA
Ingrese la categoría del producto: fruta
Ingrese el precio del producto (sin centavos): -500
ERROR: EL PRECIO INGRESADO ES NEGATIVO
Ingrese el precio del producto (sin centavos): 2500.2
ERROR: EL PRECIO NO ES UN VALOR ENTERO
Ingrese el precio del producto (sin centavos): 300
Presione ENTER para continuar
```

Figura 9: Pruebas al ingresar los campos del producto

3.3 Búsqueda de producto por nombre

Al ingresar varios productos, se guardan en la lista en forma ordenada alfabeticamente por nombre. Cuando se busca un nombre de un producto se genera una sublista con las coincidencias y se muestra por la terminal en caso en encontrar como se muestra en la Figura 10, sino muestra un mensaje que el producto no esta en el inventario.

```

1-- LISTA DE PRODUCTOS REGISTRADOS ---
1. NOMBRE: anana | CATEGORÍA: fruta | PRECIO: $300
2. NOMBRE: manzana | CATEGORÍA: fruta | PRECIO: $5000
3. NOMBRE: Naranja | CATEGORÍA: fruta | PRECIO: $2000
4. NOMBRE: PaLta | CATEGORÍA: VeRdura | PRECIO: $3000
5. NOMBRE: palta | CATEGORÍA: fruta | PRECIO: $200
6. NOMBRE: Poroto | CATEGORÍA: Legumbre | PRECIO: $2000
Presione ENTER para continuar

```

(a) Productos en el inventario

```

1-- BUSQUEDA DE PRODUCTO ---
Ingresar el nombre del producto a buscar: palta
1. Nombre: PaLta | Categoría: VeRdura | Precio: $3000
2. Nombre: palta | Categoría: fruta | Precio: $200
Presione ENTER para continuar

```

(b) Búsqueda de un producto por nombre encontrado

Figura 10: Búsqueda de producto por nombre

3.4 Eliminación de producto por posición

Como se muestra en la Figura 11, en la terminal se observan los productos que hay en el inventario, junto a un numero, para eliminar por ejemplo naranja, se ingresa su posición en la lista mostrada que es 3, luego se vuelve a mostrar los productos y se observa que fue eliminado del inventario.

```

1-- ELIMINACION DE PRODUCTO ---
--- LISTA DE PRODUCTOS REGISTRADOS ---
1. NOMBRE: anana | CATEGORÍA: fruta | PRECIO: $300
2. NOMBRE: manzana | CATEGORÍA: fruta | PRECIO: $5000
3. NOMBRE: Naranja | CATEGORÍA: fruta | PRECIO: $2000
4. NOMBRE: PaLta | CATEGORÍA: VeRdura | PRECIO: $3000
5. NOMBRE: palta | CATEGORÍA: fruta | PRECIO: $200
6. NOMBRE: Poroto | CATEGORÍA: Legumbre | PRECIO: $2000
Ingrese el número del producto a eliminar: 3

```

(a) Posición del producto a eliminar

```

1-- LISTA DE PRODUCTOS REGISTRADOS ---
1. NOMBRE: anana | CATEGORÍA: fruta | PRECIO: $300
2. NOMBRE: manzana | CATEGORÍA: fruta | PRECIO: $5000
3. NOMBRE: PaLta | CATEGORÍA: VeRdura | PRECIO: $3000
4. NOMBRE: palta | CATEGORÍA: fruta | PRECIO: $200
5. NOMBRE: Poroto | CATEGORÍA: Legumbre | PRECIO: $2000
Presione ENTER para continuar

```

(b) Inventario despues de eliminar el producto

Figura 11: Eliminación de producto por posición en el menú

4 Código

4.1 makefile

```

#Para ejecutar el archivo utilizando este makefile escribir en terminal:
# make
# o sino:

```

```
# make run
run: main.py metodos_productos.py menu.py opciones.py
python3 main.py
```

4.2 main.py

```
# Pre_Entrega
# Alumno: Paz Portilla, Jose Miguel
# Comision: 25010

#De metodos_productos.py importa los metodos:
#ingresar_producto, mostrar_productos, buscar_producto, eliminar_producto.
from metodos_productos import (
    ingresar_producto,
    mostrar_productos,
    buscar_producto,
    eliminar_producto,
    cargar_productos,
    guardar_productos
)

#De menu.py importa los metodos:
#limpiar_pantalla, mostrar_menu, ejecutar_opcion, ingresar_opcion.
from menu import (
    limpiar_pantalla,
    mostrar_menu,
    ejecutar_opcion,
    ingresar_opcion
)

"""
Es el programa principal, repetitivamente limpia la terminal(hasta ingresar
↪ terminar program),
muestra un menu de opciones, toma la opcion y realiza alguna accion sobre
↪ productos a partir de ella.
Parámetros:-
Retorna:-
"""
def main () :

    productos = cargar_productos()
    volver_al_menu = True

    while volver_al_menu == True :
        limpiar_pantalla ()
        mostrar_menu ()
        opcion = ingresar_opcion ()
        volver_al_menu = ejecutar_opcion ( opcion , productos )
    guardar_productos(productos)
    limpiar_pantalla ()
    print("\n\n\t\t;PROGRAMA FINALIZADO!\n\n")
```

```
if __name__ == "__main__":  
    main()
```

4.3 menu.py

```
#Importa el módulo estándar os, esto proporciona  
#funciones para interactuar con el sistema operativo.  
import os  
import opciones  
  
from colorama import Fore, Style, Back, init  
init(autoreset=True)  
  
from metodos_productos import (  
    ingresar_producto,  
    mostrar_productos,  
    buscar_producto,  
    eliminar_producto  
)  
  
"""  
Refresca o limpia la terminal  
Parámetros:-  
Retorna:-  
"""  
  
def limpiar_pantalla():  
    # os.system: Ejecuta un comando del sistema operativo  
    # como si se escribiera en la terminal.  
    # Si el nombre del sistema operativo es windows utiliza cls,  
    # sino utiliza clear para refrescar la pantalla  
    os.system('cls' if os.name == 'nt' else 'clear')  
  
"""  
Muestra las opciones del menu por la terminal  
Parámetros:-  
Retorna:-  
"""  
  
def mostrar_menu():  
    print(Fore.BLUE + Back.CYAN + "--- MENÚ DE OPCIONES ---" + Style.RESET_ALL)  
    print(Fore.YELLOW + "1. Ingresar un nuevo producto")  
    print(Fore.YELLOW + "2. Ver productos registrados")  
    print(Fore.YELLOW + "3. Buscar producto por nombre")  
    print(Fore.YELLOW + "4. Eliminar un producto")  
    print(Fore.RED + "5. Salir")  
  
"""  
Muestra las opciones del menu por la terminal  
Parámetros:-
```



```

Retorna: la opcion que se ingreso por teclado,
eliminando los espacios en blanco al inicio y al final
"""
def ingresar_opcion ():
    opcion = input ( Fore.GREEN + "Seleccionar una opción (1-5): " ).strip()
    return opcion

"""
Realiza la opcion sobre la lista de productos
Parámetros:- opcion a realizar y la lista de productos
Retorna: Bool para indicar si debe seguir pidiendo opciones o no
"""
def ejecutar_opcion ( opcion , productos ) :
    match opcion:
        case opciones.INGRESAR_NUEVO_PRODUCTO:
            limpiar_pantalla()
            ingresar_producto(productos)
        case opciones.MOSTRAR_PRODUCTOS_POR_TERMINAL:
            limpiar_pantalla()
            mostrar_productos(productos)
        case opciones.BUSCAR_PRODUCTO_POR_NOMBRE:
            limpiar_pantalla()
            buscar_producto(productos)
        case opciones.ELIMINAR_PRODUCTO_POR_INDICE:
            limpiar_pantalla()
            eliminar_producto(productos)
        case opciones.FINALIZAR_PROGRAMA:
            return False # salir del programa
        case _:
            print(Fore.RED + "ERROR: OPCIÓN FUERA DEL RANGO")

    input(Fore.MAGENTA + "Presione ENTER para continuar")
    return True # continuar el bucle

```

4.4 opciones.py

```

#Constantes que representan las opciones del menu
INGRESAR_NUEVO_PRODUCTO = '1'
MOSTRAR_PRODUCTOS_POR_TERMINAL = '2'
BUSCAR_PRODUCTO_POR_NOMBRE = '3'
ELIMINAR_PRODUCTO_POR_INDICE = '4'
FINALIZAR_PROGRAMA = '5'

```

4.5 metodos_productos.py

```

# Se definen las funciones que sirven para manipular los productos

import os

```

```

import csv # CSV = Comma-Separated Values: utiliza los metodos read() y write()
↳ para manejar la base_datos.csv con separacion de comas, writerows convierte
↳ la lista productos como sublistas producto con comas y los almacena por
↳ filas.
from colorama import Fore, Style, Back, init
init(autoreset=True)
NOMBRE = 0 # producto[0]: NOMBRE, producto[1]: CATEGORIA, producto[2]: PRECIO

"""
Genera un producto a partir del nombre, categoria y precio ingresados por
↳ teclado
Ingresa el producto a la lista de productos
Parámetros:
productos(list): lista de productos
Retorna:-
"""

def ingresar_producto ( productos ):
    print ("--- NUEVO PRODUCTO ---")
    while True:
        nombre = input(Fore.YELLOW + "Ingrese el nombre del producto:
↳ ").strip()
        if nombre != "":
            break
        print(Fore.RED + "ERROR: NOMBRE DE PRODUCTO ESTA VACIA")

    while True:
        categoria = input(Fore.YELLOW + "Ingrese la categoría del
↳ producto: ").strip()
        if categoria != "":
            break
        print(Fore.RED + "ERROR: NOMBRE DE LA CATEGORIA ESTA VACIA")

    while True:
        #Bloque para atrape el error de conversion de cadena a entero
        try:
            precio = int(input(Fore.YELLOW + "Ingrese el precio del
↳ producto (sin centavos): "))
            if precio < 0:
                print(Fore.RED + "ERROR: EL PRECIO INGRESADO ES
↳ NEGATIVO")
                continue
            break
        except ValueError:
            print(Fore.RED + "ERROR: EL PRECIO NO ES UN VALOR
↳ ENTERO")

    producto = [nombre, categoria, precio]

    # Insertar producto en la lista de productos ordenado por nombre
    producto_fue_insertado = False
    for indice in range(len(productos)): #i va de 0 hasta len(productos)-1

```

```

        # compara ignorando mayúsculas/minúsculas
        if nombre.lower() < productos[indice][NOMBRE].lower():
            productos.insert(indice, producto)
            producto_fue_insertado = True
            break
    # Si no fue ingresado producto en la lista de productos,
    # debido a que al compararlo no hubo otro nombre menor alfabeticamente,
    # Simplemente se agrega el producto al final de la lista productos
    if not producto_fue_insertado:
        productos.append(producto)

"""
Muestra por terminal la lista de productos
con el formato: nombre/categoria/precio
Parámetros:
productos(list): lista de productos
Retorna:-
"""
def mostrar_productos ( productos ) :
    if len (productos) == 0:
        print (Fore.RED + "No hay productos registrados.")
        return

    print (Fore.BLUE + "--- LISTA DE PRODUCTOS REGISTRADOS ---")
    # Empieza a enumerar desde 1 el i, y va deserializando producto de la
    ↪ lista_productos
    for indice, producto in enumerate( productos , start = 1 ):
        nombre, categoria, precio = producto
        print(Fore.YELLOW + f"{indice}. NOMBRE: {nombre} | CATEGORÍA:
        ↪ {categoria} | PRECIO: ${precio}")

"""
Busca en la lista de productos los productos que coinciden con el nombre de
↪ productos
ingresado por teclado. Luego los muestra por terminal en caso de encontrarlo
Parámetros:
productos(list): lista de productos
Retorna:-
"""
def buscar_producto ( productos ) :
    print (Fore.BLUE + "--- BUSQUEDA DE PRODUCTO ---")
    # Borra espacios de los bordes y lo paso a minusculas
    nombre = input(Fore.GREEN + "Ingresar el nombre del producto a buscar:
    ↪ ").strip().lower()
    if nombre == "":
        print(Fore.RED + "ERROR: NOMBRE DE PRODUCTO A BUSCAR ESTA
        ↪ VACIA")
        return

    encontrados = []
    for producto in productos:

```

```

        if nombre in producto[NOMBRE].lower():
            encontrados.append(producto)

    if len(encontrados) != 0:
        for indice, encontrado in enumerate ( encontrados , start = 1 )
        ↪ :
            nombre, categoria, precio = encontrado
            print(Fore.CYAN + f"{indice}. Nombre: {nombre} |
            ↪ Categoría: {categoria} | Precio: ${precio}")
    else:
        print(Fore.RED + "NOMBRE DE PRODUCTO NO ESTA EN EL INVENTARIO")

"""
Elimina un producto de la lista de productos a partir de la posicion mostrada
↪ por terminal
Parámetros:
productos(list): lista de productos
Retorna:-
"""
def eliminar_producto ( productos ):
    print ("--- ELIMINACION DE PRODUCTO ---")
    if len(productos) == 0:
        print (Fore.RED + "INVENTARIO ESTÁ VACIO")
        return

    mostrar_productos(productos)
    while True:
        try:
            indice = int(input(Fore.GREEN + "Ingrese el número del
            ↪ producto a eliminar: "))
            if 1 <= indice and indice <= len(productos):
                #Se resta 1 porque la posicion en la lista
                ↪ empieza en 0
                eliminado = productos.pop ( indice - 1 )
                break
            else:
                print(Fore.RED + "ERROR: OPCION DE LA LISTA DE
                ↪ PRODUCTOS FUERA DE RANGO")
        except ValueError:
            print(Fore.RED + "ERROR: INGRESO DE UN NUMERO NO
            ↪ ENTERO")

def cargar_productos (archivo='base_datos.csv'):
    productos = []
    if os.path.exists (archivo):
        with open (archivo, 'r', newline = '') as base_datos:
            productos_csv = csv.reader(base_datos)
            for producto in productos_csv:
                nombre, categoria, precio = producto[0],
                ↪ producto[1], int(producto[2])
            productos.append ([nombre, categoria, precio])

```

```
        return productos

def guardar_productos (productos, archivo='base_datos.csv'):
    with open(archivo, 'w', newline='') as base_datos:
        productos_csv = csv.writer (base_datos)
        productos_csv.writerows (productos)
```

5 Conclusión

En este informe realizado al combinar L^AT_EX y Python se presenta una simplificación del programa inventario, donde se pudo aplicar lo aprendido en clases, como listas, condicionales, bucles, además se usó modulación y funciones, además de el uso de la función main y un makefile para ejecutar de una forma más práctica el código.