

# Python Inicial

Jose Miguel Paz Portilla  
Pre-Entrega de Proyecto

21 de mayo de 2025

# Índice

<b>1</b>	<b>Introducción</b>	<b>3</b>
<b>2</b>	<b>Implementación</b>	<b>3</b>
<b>3</b>	<b>makefile</b>	<b>5</b>
<b>4</b>	<b>main.py</b>	<b>5</b>
<b>5</b>	<b>menu.py</b>	<b>6</b>
<b>6</b>	<b>opciones.py</b>	<b>7</b>
<b>7</b>	<b>metodos_productos.py</b>	<b>7</b>
<b>8</b>	<b>Conclusión</b>	<b>10</b>

# 1 Introducción

Con el objetivo de realizar un inventario de productos realizados en python, se realizo una version preliminar con 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.

# 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

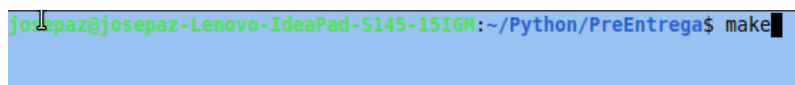
A terminal window with a blue background. The prompt is 'josepaz@josepaz-Lenovo-IdeaPad-S145-15IGM:~/Python/PreEntrega\$'. The command 'make' has been entered and is followed by a cursor.

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.

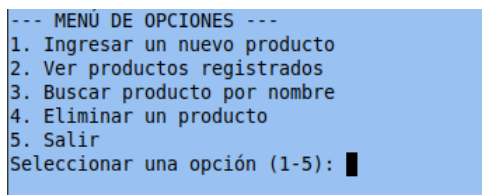
A terminal window with a blue background. It displays a menu titled '--- MENÚ DE OPCIONES ---'. The menu items are: '1. Ingresar un nuevo producto', '2. Ver productos registrados', '3. Buscar producto por nombre', '4. Eliminar un producto', and '5. Salir'. Below the menu, it says 'Seleccionar una opción (1-5):' followed by a cursor.

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.

```
--- BÚSQUEDA 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: Finalizacion del programa

### 3 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 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 )

#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 = []
    volver_al_menu = True

    while volver_al_menu == True :
        limpiar_pantalla ()
        mostrar_menu ()
        opcion = ingresar_opcion ()
        volver_al_menu = ejecutar_opcion ( opcion , productos )

    limpiar_pantalla ()
    print("\n\n\t\t;PROGRAMA FINALIZADO!\n\n")

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

## 5 menu.py

```
#Importa el módulo estándar os, esto proporciona
#funciones para interactuar con el sistema operativo.
import os
import opciones

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 ("--- MENÚ DE OPCIONES ---")
    print ("1. Ingresar un nuevo producto")
    print ("2. Ver productos registrados")
    print ("3. Buscar producto por nombre")
    print ("4. Eliminar un producto")
    print ("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 ( "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("ERROR: OPCIÓN FUERA DEL RANGO")

    input("Presione ENTER para continuar")
    return True # continuar el bucle

```

## 6 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'

```

## 7 metodos\_\_productos.py

```

# Se definen las funciones que sirven para manipular los productos

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("Ingrese el nombre del producto: ").strip()
    if nombre != "":
        break
    print("El nombre no puede estar vacío.")

while True:
    categoria = input("Ingrese la categoría del producto: ").strip()
    if categoria != "":
        break
    print("La categoría no puede estar vacía.")

while True:
    #Bloque para atrape el error de conversion de cadena a entero
    try:
        precio = int(input("Ingrese el precio del producto (sin
        ↪ centavos): "))
        if precio < 0:
            print("El precio no puede ser negativo.")
            continue
        break
    except ValueError:
        print("Entrada inválida. Ingrese un número entero.")

producto = [nombre, categoria, precio]
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 ("No hay productos registrados.")
        return

    print ("--- 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(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 ("--- BUSQUEDA DE PRODUCTO ---")
    # Borra espacios de los bordes y lo paso a minusculas
    nombre = input("Ingresar el nombre del producto a buscar:
    ↪ ").strip().lower()
    if nombre == "":
        print("El término de búsqueda no puede estar vacío.")
        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(f"{indice}. Nombre: {nombre} | Categoría:
            ↪ {categoria} | Precio: ${precio}")
    else:
        print("No se encontraron productos con ese nombre.")

"""
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 ("No hay productos para eliminar.")
        return

    mostrar_productos(productos)
    while True:
        try:
            indice = int(input("Ingresa 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("Número inválido. Intente nuevamente.")

```

```
except ValueError:  
    print("Entrada inválida. Ingrese un número entero.")
```

## 8 Conclusión

En este informe realizado al combinar L<sup>A</sup>T<sub>E</sub>X 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 un forma más práctica el código.