

TRABAJO FINAL PROGRAMACION ORIENTADA A OBJETOS

Ciberseguridad 2025

Ing.Carlos García

Alexandra Arisneidy Sanchez Santana



Indicaciones:

Programación Orienta a objeto

Proyecto final

- Desarrollar una aplicación visual con la librería tkinter que permita implementar los algoritmos de carga de artículos, consulta por código y listado completo, esto debe estar integrado con Mysql para almacenar los datos.

Código del trabajo

```
import sqlite3
import tkinter as tk
from tkinter import ttk, messagebox

# -----
# CLASE CONEXIÓN (POO)
# -----
class ConexionDB:
    def __init__(self):
        self.conexion = sqlite3.connect("articulos.db")
        self.cursor = self.conexion.cursor()
        self.crear_tabla()

    def crear_tabla(self):
        self.cursor.execute("""
            CREATE TABLE IF NOT EXISTS articulos (
                codigo INTEGER PRIMARY KEY,
                descripcion TEXT,
                precio REAL
            )
        """)
```

```

        )
        """
    self.conexion.commit()

# -----
# CLASE DAO (ALGORITMOS)
# -----
class ArticuloDAO:
    def __init__(self):
        self.db = ConexionDB()

    def insertar(self, codigo, descripcion, precio):
        self.db.cursor.execute(
            "INSERT INTO articulos VALUES (?, ?, ?)",
            (codigo, descripcion, precio)
        )
        self.db.conexion.commit()

    def consultar(self, codigo):
        self.db.cursor.execute(
            "SELECT * FROM articulos WHERE codigo=?",
            (codigo,)
        )
        return self.db.cursor.fetchone()

    def listar(self):
        self.db.cursor.execute("SELECT * FROM articulos")
        return self.db.cursor.fetchall()

# -----
# CLASE INTERFAZ (TKINTER)
# -----
class App:
    def __init__(self, root):
        self.root = root
        self.root.title("Mantenimiento de artículos")
        self.root.geometry("400x300")

        self.dao = ArticuloDAO()

        self.notebook = ttk.Notebook(root)
        self.notebook.pack(fill="both", expand=True)

        self.tab_carga()
        self.tab_consulta()

```

```

        self.tab_listado()

# ----- TAB 1 -----
def tab_carga(self):
    tab = ttk.Frame(self.notebook)
    self.notebook.add(tab, text="Carga de artículos")

    tk.Label(tab, text="Código").pack()
    self.ent_cod = tk.Entry(tab)
    self.ent_cod.pack()

    tk.Label(tab, text="Descripción").pack()
    self.ent_desc = tk.Entry(tab)
    self.ent_desc.pack()

    tk.Label(tab, text="Precio").pack()
    self.ent_precio = tk.Entry(tab)
    self.ent_precio.pack()

    tk.Button(tab, text="Confirmar", command=self.guardar).pack(pady=10)

def guardar(self):
    try:
        self.dao.insertar(
            int(self.ent_cod.get()),
            self.ent_desc.get(),
            float(self.ent_precio.get())
        )
        messagebox.showinfo("OK", "Artículo guardado")
    except:
        messagebox.showerror("Error", "Datos incorrectos")

# ----- TAB 2 -----
def tab_consulta(self):
    tab = ttk.Frame(self.notebook)
    self.notebook.add(tab, text="Consulta por código")

    tk.Label(tab, text="Código").pack()
    self.ent_buscar = tk.Entry(tab)
    self.ent_buscar.pack()

    tk.Button(tab, text="Consultar", command=self.consultar).pack(pady=5)

    self.lbl_res = tk.Label(tab, text="")
    self.lbl_res.pack()

```

```

def consultar(self):
    art = self.dao.consultar(int(self.ent_buscar.get()))
    if art:
        self.lbl_res.config(
            text=f"Descripción: {art[1]}\nPrecio: {art[2]}"
        )
    else:
        messagebox.showinfo("Info", "Artículo no encontrado")

# ----- TAB 3 -----
def tab_listado(self):
    tab = ttk.Frame(self.notebook)
    self.notebook.add(tab, text="Listado completo")

    self.text = tk.Text(tab)
    self.text.pack(fill="both", expand=True)

    tk.Button(tab, text="Listado completo", command=self.listar).pack()

def listar(self):
    self.text.delete("1.0", tk.END)
    for art in self.dao.listar():
        self.text.insert(tk.END, f"{art}\n")

# -----
# EJECUCIÓN
# -----
if __name__ == "__main__":
    root = tk.Tk()
    app = App(root)
    root.mainloop()

```

Imágenes del trabajo funcionado

Modelo

Mantenimiento de artículos

Carga de artículos Consulta por código Listado completo

Código

Descripción

Precio

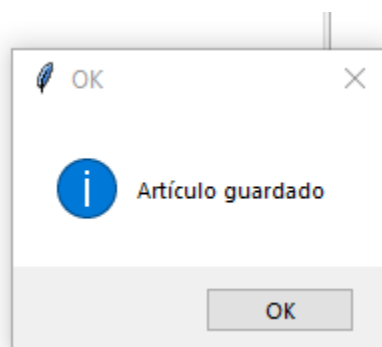
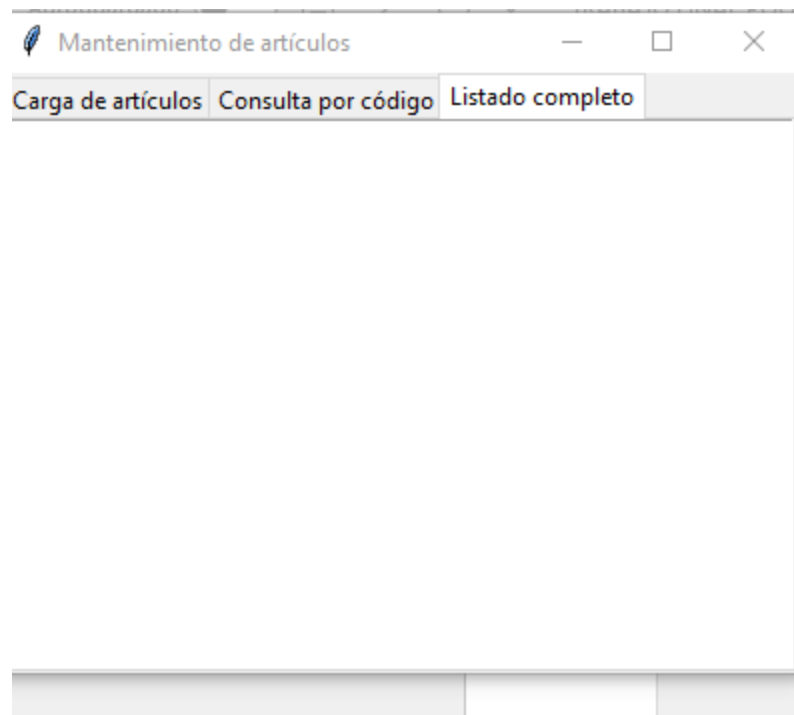
Confirmar

Mantenimiento de artículos

Carga de artículos Consulta por código Listado completo

Código

Consultar



Mantenimiento de artículos

Carga de artículos Consulta por código Listado completo

Código

1

Consultar

Descripción: MANZANA
Precio: 50.0

Mantenimiento de artículos

Carga de artículos Consulta por código Listado completo

Código

2

Consultar

Descripción: FILO
Precio: 10.0