

LABORATORIO DE PROGRAMACIÓN EN PYTHON

Sistema de Gestión de Inventario para Tienda de Tecnología

INTRODUCCIÓN

El manejo eficiente de inventarios es fundamental en cualquier negocio. Los estudiantes desarrollarán un sistema de gestión de inventario para una tienda de tecnología que permita administrar productos, realizar ventas, generar reportes y mantener control del stock disponible.

Este proyecto integrador combina los conceptos fundamentales de programación en Python: estructuras condicionales para la toma de decisiones, ciclos para procesar datos repetitivamente, funciones para modularizar el código y arreglos (listas y diccionarios) para el almacenamiento y manipulación de información estructurada.

A través de este proyecto, los estudiantes experimentarán un escenario real de desarrollo de software, aplicando buenas prácticas de programación y trabajo colaborativo.

OBJETIVO DEL PROYECTO

Objetivo General: Desarrollar un sistema de gestión de inventario funcional en Python que integre estructuras condicionales, ciclos, funciones y arreglos para administrar productos de una tienda de tecnología.

Objetivos Específicos:

- Implementar funciones modulares para cada operación del sistema (agregar, buscar, modificar, eliminar productos)
- Utilizar estructuras condicionales para validar datos y controlar el flujo del programa
- Aplicar ciclos para procesar colecciones de datos y menús interactivos
- Emplear listas y diccionarios para almacenar y gestionar información de productos
- Desarrollar habilidades de trabajo en equipo y documentación de código
- Crear un sistema con interfaz de usuario basada en consola intuitiva y funcional

HACER EQUIPO DE 5 INTEGRANTES

Distribución de Roles:

1. **Líder de Proyecto (Project Manager)**
 - Coordina las actividades del equipo
 - Supervisa el cumplimiento de plazos
 - Integra los módulos desarrollados por cada miembro
2. **Desarrollador de Funciones de Entrada (Input Developer)**
 - Implementa funciones para agregar productos
 - Desarrolla funciones de actualización de inventario
 - Valida datos de entrada del usuario
3. **Desarrollador de Funciones de Consulta (Query Developer)**
 - Crea funciones de búsqueda y filtrado
 - Implementa funciones de visualización de productos
 - Desarrolla reportes básicos
4. **Desarrollador de Funciones de Procesamiento (Processing Developer)**
 - Implementa funciones de ventas
 - Desarrolla cálculos de totales y estadísticas
 - Crea funciones de ordenamiento
5. **Responsable de Pruebas y Documentación (QA & Documentation)**
 - Prueba todas las funciones del sistema
 - Documenta el código y genera manual de usuario
 - Identifica y reporta errores

Nota: Todos los integrantes deben contribuir al código y comprender cada módulo del sistema.

MATERIALES A OCUPAR

Software:

- Python
- Editor de código (Visual Studio Code, PyCharm, GoogleColab, Sublime Text o similar)
- Git para control de versiones (opcional pero recomendado)

Hardware:

- Computadora

Documentación y Recursos:

- Documentación oficial de Python (<https://docs.python.org/es/3/>)
- Cuaderno o documento digital para planificación
- Repositorio compartido (GitHub, GitLab o Google Drive)

Conocimientos Previos Requeridos:

- Sintaxis básica de Python

- Tipos de datos (int, float, str, bool)
- Operadores aritméticos y lógicos
- Entrada y salida básica (input/print)

DESARROLLO DEL PROYECTO

Descripción del Sistema

El sistema debe gestionar un inventario de productos tecnológicos con las siguientes características:

Información de cada producto:

- ID del producto (único, numérico)
- Nombre del producto
- Categoría (Computadoras, Smartphones, Accesorios, Periféricos)
- Precio unitario
- Cantidad en stock
- Proveedor

Funcionalidades principales:

1. **Gestión de Productos**
 - Agregar nuevos productos
 - Modificar información de productos existentes
 - Eliminar productos
 - Buscar productos por diferentes criterios
2. **Operaciones de Venta**
 - Registrar ventas
 - Actualizar automáticamente el inventario
 - Calcular totales de venta
3. **Reportes y Consultas**
 - Listar todos los productos
 - Mostrar productos con stock bajo (menos de 5 unidades)
 - Calcular valor total del inventario
4. **Menú Interactivo**
 - Interfaz basada en consola
 - Validación de opciones del usuario
 - Mensajes de confirmación y error

METODOLOGÍA

Fase 1: Planificación y Diseño

Actividades:

- Reunión de equipo para distribución de roles
- Análisis de requerimientos del sistema
- Diseño de la estructura de datos (diccionarios para productos)
- Definición de funciones principales
- Creación de diagrama de flujo del menú principal

Entregable: Documento de diseño con estructura de datos y lista de funciones

Fase 2: Implementación de Funciones Básicas

Actividades:

- Crear estructura base del programa (menú principal)
- Implementar funciones de agregar y listar productos
- Desarrollar funciones de búsqueda básica
- Implementar validaciones de entrada
- Uso de ciclos while para menús
- Uso de ciclos for para recorrer inventario

Entregable: Código funcional con operaciones básicas CRUD

Fase 3: Implementación de Funcionalidades Avanzadas

Actividades:

- Implementar sistema de ventas
- Crear funciones de reportes
- Desarrollar búsquedas avanzadas con filtros
- Implementar ordenamiento de productos
- Agregar función de respaldo de datos (opcional)

Entregable: Sistema completo con todas las funcionalidades

Fase 4: Pruebas y Documentación

Actividades:

- Pruebas unitarias de cada función
- Pruebas de integración del sistema completo
- Corrección de errores identificados
- Documentación del código (comentarios y docstrings)
- Elaboración de manual de usuario
- Preparación de presentación

Entregable: Sistema probado, documentado y presentación final

CUESTIONARIO SOBRE EL PROYECTO

Responder las siguientes preguntas de manera individual (cada integrante debe presentar sus respuestas):

Sección 1: Estructuras Condicionales

1. ¿Qué estructura condicional utilizaste para validar que el ID del producto sea único? Explica el proceso de validación.
2. En el sistema de ventas, ¿cómo implementaste la condición para verificar si hay suficiente stock antes de realizar una venta?
3. Proporciona un ejemplo de uso de `elif` en tu código y explica por qué fue necesario.

Sección 2: Ciclos

4. ¿Qué tipo de ciclo utilizaste para el menú principal y por qué elegiste ese tipo?
5. Explica la diferencia entre usar `for` y `while` en el contexto de tu proyecto. Da un ejemplo de cada uno.
6. ¿Cómo implementaste la búsqueda de productos usando ciclos? ¿Qué pasaría si tuvieras 10,000 productos?

Sección 3: Funciones

7. Lista las cinco funciones más importantes de tu sistema y describe brevemente qué hace cada una.
8. ¿Por qué es importante usar funciones en lugar de escribir todo el código en el programa principal?
9. Describe una función que reciba parámetros y retorne un valor. ¿Qué ventaja tiene usar `return`?

Sección 4: Arreglos (Listas y Diccionarios)

10. ¿Qué estructura de datos elegiste para almacenar los productos (lista de diccionarios, diccionario de diccionarios, etc.)? Justifica tu elección.
11. Explica cómo agregaste un nuevo producto al inventario usando operaciones con arreglos.
12. ¿Cómo implementaste la eliminación de un producto? ¿Qué método de listas o diccionarios utilizaste?

Sección 5: Integración y Análisis

13. ¿Cuál fue el mayor desafío técnico que enfrentaste durante el desarrollo y cómo lo resolviste?

14. Si tuvieras que agregar una nueva funcionalidad (por ejemplo, sistema de descuentos), ¿cómo la integrarías al código existente?
15. ¿Qué aprendiste sobre trabajo en equipo durante este proyecto? ¿Cómo se organizaron para evitar conflictos en el código?

ACTIVIDAD A REALIZAR DEL PROYECTO INTEGRADOR

Entrega Final del Proyecto

Fecha de Entrega: [Especificar fecha según calendario del curso]

Componentes de la Entrega:

1. Código Fuente (40%)

Entregar un archivo `.py` o un conjunto de archivos `.py` que contenga:

- Código completo y funcional
- Comentarios explicativos en cada función
- Docstrings en funciones principales
- Código formateado y legible

Estructura mínima requerida:

```
# Sistema de Gestión de Inventario
# Equipo: [Nombres de integrantes]
# Fecha: [Fecha]

# Lista o diccionario para almacenar productos
inventario = []

# Funciones requeridas:
# - agregar_producto()
# - buscar_producto()
# - modificar_producto()
# - eliminar_producto()
# - realizar_venta()
# - mostrar_inventario()
# - generar_reporte()
# - menu_principal()
```

3. Documentación Técnica (20%)

Documento que contenga:

- Diagrama de flujo del programa
- Descripción de cada función (parámetros, retorno, propósito)

- Estructura de datos utilizada
- Decisiones de diseño y justificación
- Limitaciones conocidas del sistema

4. Demostración (15%)

Demostración de 5-10 minutos:

- Ejecución del programa
- Demostración de todas las funcionalidades
- Explicación breve de partes clave del código
- Cada integrante debe participar en la explicación

5. Respuestas al Cuestionario (15%)

- Archivo PDF con respuestas individuales de cada integrante
- Respuestas completas y fundamentadas
- Ejemplos de código cuando sea necesario

Criterios de Evaluación Adicionales

Funcionalidad (40 puntos):

- El programa ejecuta sin errores (10 pts)
- Todas las funcionalidades requeridas están implementadas (15 pts)
- Validaciones de entrada apropiadas (10 pts)
- Interfaz de usuario clara e intuitiva (5 pts)

Código (30 puntos):

- Uso correcto de estructuras condicionales (7 pts)
- Implementación apropiada de ciclos (7 pts)
- Funciones bien diseñadas y modulares (8 pts)
- Uso eficiente de arreglos (8 pts)

Documentación (15 puntos):

- Comentarios claros en el código (5 pts)
- Documentación técnica detallada (10 pts)

Presentación y Trabajo en Equipo (15 puntos)

- Demostración profesional (10 pts)
- Evidencia de colaboración equitativa (5 pts)

Rúbrica Específica por Funcionalidad

Cada funcionalidad debe cumplir:

Agregar Producto:

- Solicita todos los datos requeridos
- Valida que el ID sea único
- Valida tipos de datos (precio numérico, cantidad entera)
- Confirma la operación exitosa

Buscar Producto:

- Permite búsqueda por ID, nombre, proveedor o categoría
- Maneja casos donde no se encuentra el producto
- Muestra resultados de forma clara

Modificar Producto:

- Permite seleccionar qué campo modificar
- Valida nuevos valores ingresados
- Actualiza correctamente el inventario

Eliminar Producto:

- Solicita confirmación antes de eliminar
- Elimina correctamente del inventario
- Maneja casos de IDs inexistentes

Realizar Venta:

- Verifica disponibilidad de stock
- Actualiza inventario automáticamente
- Calcula y muestra total de venta
- Maneja ventas de múltiples productos

Reportes:

- Muestra inventario completo ordenado
- Identifica productos con stock bajo
- Calcula valor total del inventario
- Formato de salida legible

REFERENCIAS

Matthes, E. (2019). *Python crash course: A hands-on, project-based introduction to programming* (2nd ed.). No Starch Press.

McKinney, W. (2022). *Python for data analysis: Data wrangling with pandas, NumPy, and Jupyter* (3rd ed.). O'Reilly Media.

Python Software Foundation. (2024). *The Python tutorial*.
<https://docs.python.org/3/tutorial/>

Ramalho, L. (2022). *Fluent Python: Clear, concise, and effective programming* (2nd ed.). O'Reilly Media.

Real Python. (2024). *Python functions*. <https://realpython.com/defining-your-own-python-function/>

Sweigart, A. (2019). *Automate the boring stuff with Python: Practical programming for total beginners* (2nd ed.). No Starch Press.

Van Rossum, G., Warsaw, B., & Coghlan, N. (2013). *PEP 8 – Style guide for Python code*. Python.org. <https://peps.python.org/pep-0008/>

Puntos Extra (5 puntos extras por cada opción)

- a) Uso de interfaz gráfica (no consola)
- b) Uso de bases de datos
- c) Informe estadístico de productos más vendidos
- d) Implementación de archivos