

**Inteligencia Artificial**

**Curso: Programacion I**

**Tema: Programacion en C**

**REALIZADO POR:**

**Dilan Alejandro Salas Hidalgo**

**DOCENTE**

**Ing. Rina Maribel Guerra Chiriboga**

**FECHA**

22 de mayo, 2025

Contenido

[1. Introducción 3](#_Toc198316665)

[2. Análisis del Problema y Diseño de la Solución 3](#_Toc198316666)

[3. Implementación del Código 3](#_Toc198316667)

[4. Validación y Pruebas 3](#_Toc198316668)

[5. Conclusiones y Mejoras Futuras 3](#_Toc198316669)

# 1. Introducción

# Propósito del programa: Este sistema de gestión de inventario permite a una tienda realizar operaciones clave con sus productos:

# Calcular el valor total del inventario

# Identificar productos más caros y más baratos

# Determinar el precio promedio

# Buscar productos por nombre

# Importancia: Automatiza procesos manuales propensos a errores, proporcionando información valiosa para la toma de decisiones en la gestión del inventario.

# Principales retos:

# Manejar la relación entre nombres y precios sin usar estructuras

# Validar correctamente las entradas del usuario

# Garantizar que los cálculos sean precisos

# Organizar el código en múltiples archivos sin usar características avanzadas

# 2. Análisis del Problema y Diseño de la Solución

# Elementos clave:

# Variables: Arreglos paralelos para nombres (char[10][50]) y precios (float[10])

# Cálculos: Sumatorias, promedios, comparaciones

# Estructuras de control: Bucles for, condicionales if, while para validación

# Funciones: Modularización por operaciones específicas

# Alternativas de solución:

# Alternativa A:

# Arreglos paralelos sincronizados

# Funciones específicas para cada operación

# Validación de entrada en cada paso

# Alternativa B:

# Arreglo único de strings con formato "nombre:precio"

# Uso de funciones de string para extraer información

# Menos arreglos, pero más procesamiento de strings

# Análisis comparativo:

# La Alternativa A fue seleccionada por:

# Mayor eficiencia en operaciones matemáticas

# Código más legible y mantenible

# Validación más sencilla de datos numéricos

# 3. Implementación del Código

# Implementé el sistema usando arreglos paralelos y funciones especializadas, organizado en 3 archivos:

# inventario.h - Declaraciones

# inventario.c - Implementaciones

# ProgramaDeTienda.c - Programa principal

# Fucionalidades clave implementadas 1. Ingreso de Productos con Validación

# 

# **Explicación**:

# Usa un bucle for para capturar numProductos elementos.

# %49s evita desbordamiento de buffer en nombres.

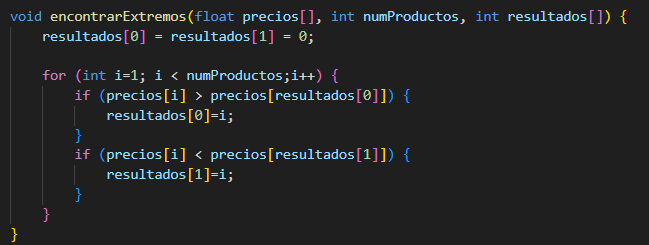
# El while valida que el precio sea un número positivo.

# 2.calculo del precio total

# 

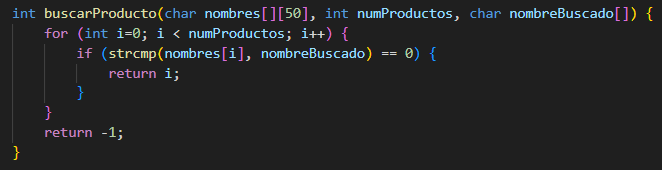
# **Lógica**: Recorre el arreglo precios[] sumando todos los valores.

# 3.Detección de Productos Extremos



* Usa un arreglo resultados[] para devolver dos valores (índices del producto más caro y más barato).
* Compara cada precio con los valores registrados.

4.Búsqueda por nombre



* Compara strings con strcmp().
* Retorna el índice del producto o -1 para "no encontrado".

# 4. Validación y Pruebas

# Casos de prueba:

# Entrada normal:

# 3 productos con precios variados

# Verificación de cálculos correctos

# Entrada inválida:

# Letras donde se esperan números

# El programa rechaza y pide nuevo ingreso

# Búsquedas:

# Producto existente: muestra precio correcto

# Producto inexistente: mensaje adecuado

# Errores encontrados:

# Falta de llave de cierre:

# Solución: Revisión cuidadosa de cada función

# Desbordamiento de buffer:

# Solución: Uso de %49s en scanf

# 5. Conclusiones y Mejoras Futuras

**Aprendizajes**:

* Importancia de la validación de entrada
* Ventajas de la modularización
* Manejo de arreglos

**Mejoras futuras**:

1. **Persistencia de datos**:
   * Guardar/leer inventario de/a archivo
2. **Interfaz mejorada**:
   * Menú interactivo con más opciones
3. **Optimizaciones**:
   * Ordenamiento para búsquedas más rápidas
   * Validación de nombres duplicados