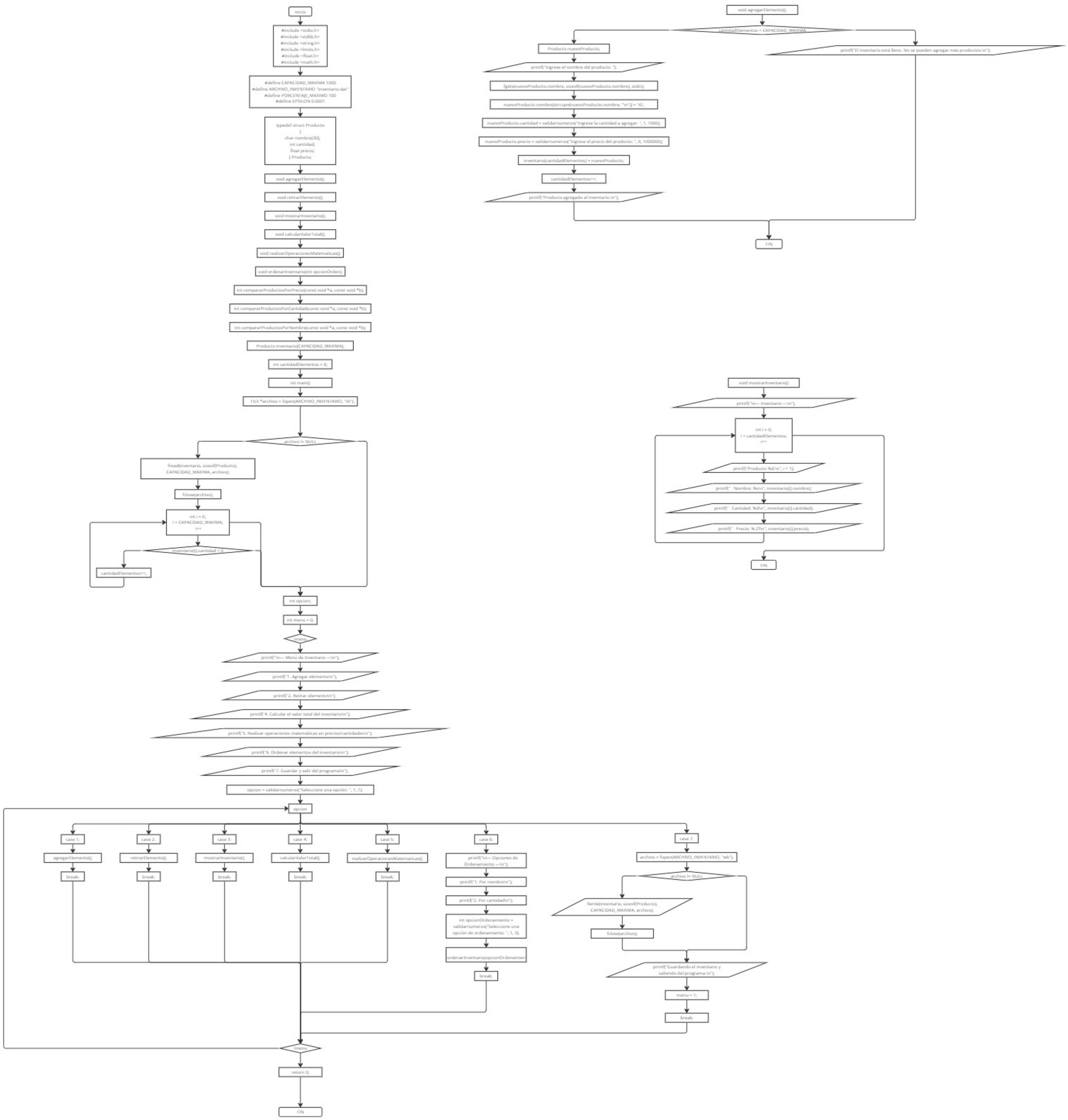
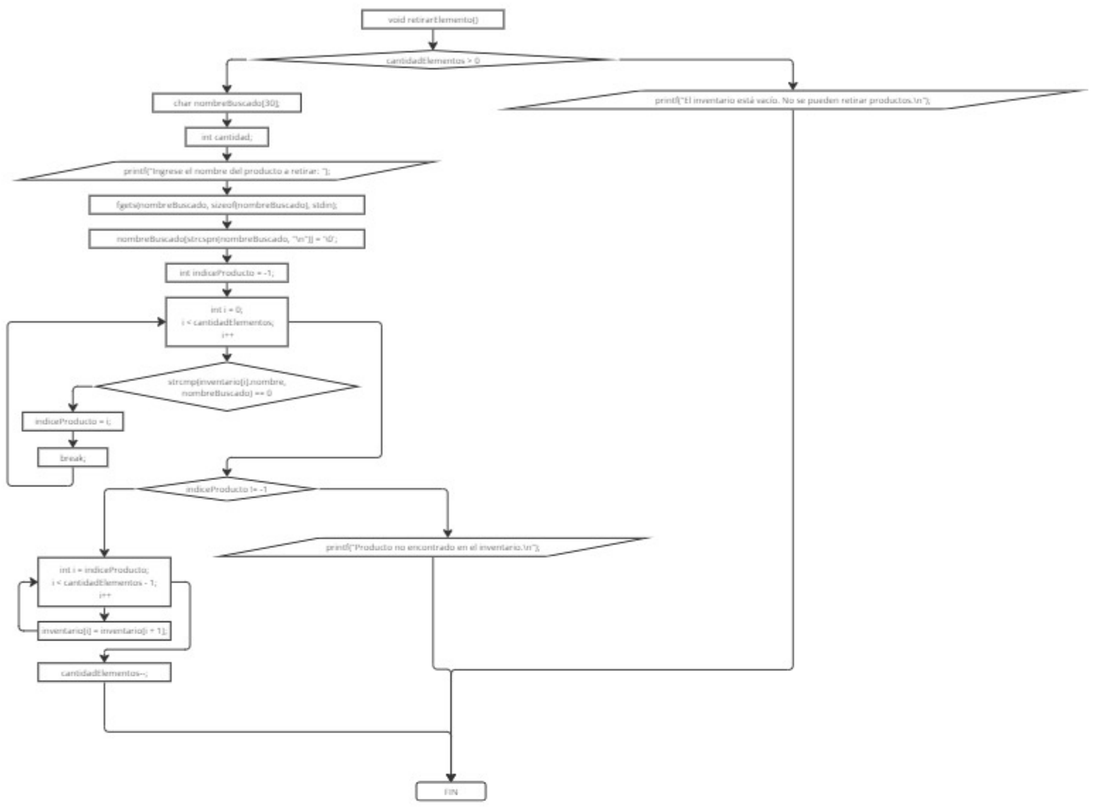
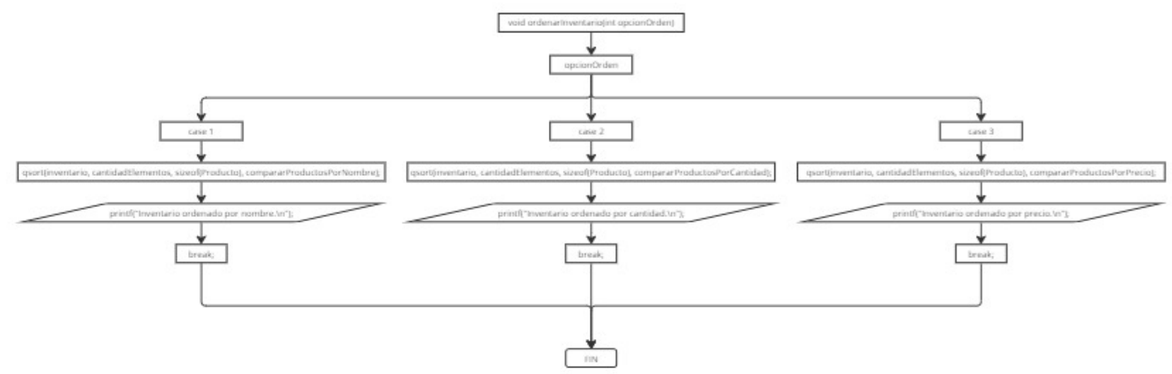
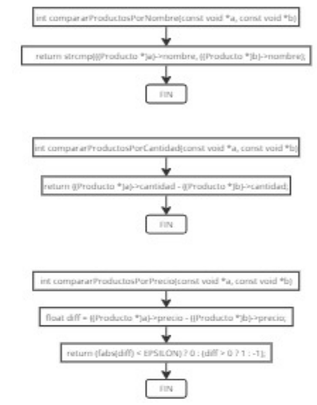
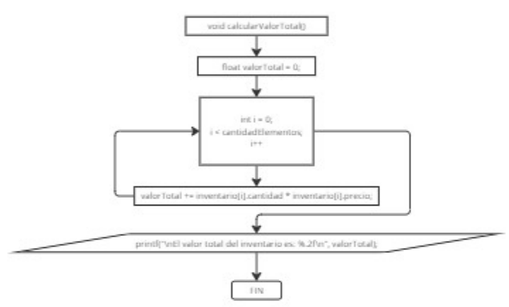
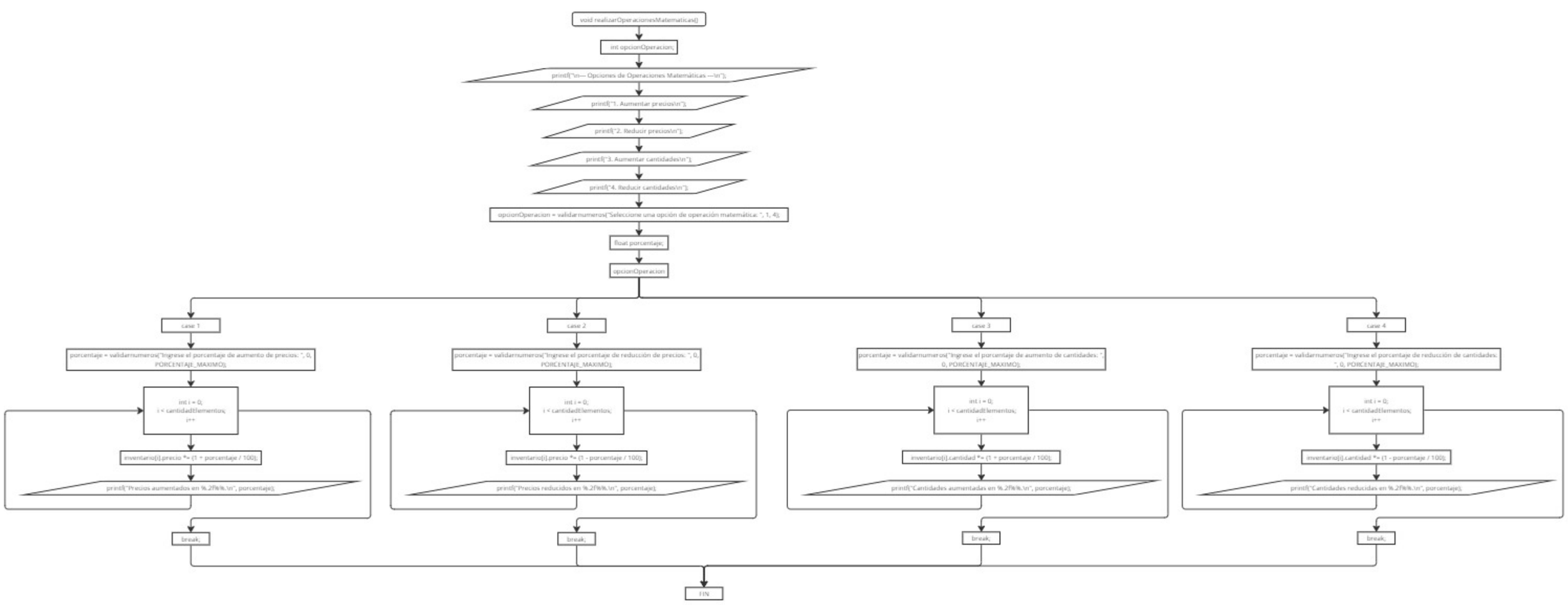

inicio
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <limits.h>
#include <float.h>
#include <math.h>
#define CAPACIDAD_MAXIMA 1000
#define ARCHIVO_INVENTARIO "inventario.dat"
#define PORCENTAJE_MAXIMO 100
#define EPSILON 0.0001
typedef struct Producto
{
char nombre[30];
int cantidad;
float precio;
} Producto;
void agregarElemento();
void retirarElemento();
void mostrarInventario();
void calcularValorTotal();
void realizarOperacionesMatematicas();
void ordenarInventario(int opcionOrden);
int compararProductosPorPrecio(const void *a, const void *b);
int compararProductosPorCantidad(const void *a, const void *b);
int compararProductosPorNombre(const void *a, const void *b);
Producto inventario[CAPACIDAD_MAXIMA];
int cantidadElementos = 0;
int main()
FILE *archivo = fopen(ARCHIVO_INVENTARIO, "rb");
archivo != NULL
fread(inventario, sizeof(Producto), CAPACIDAD_MAXIMA, archivo);
fclose(archivo);
int i = 0; i < CAPACIDAD_MAXIMA; i++
inventario[i].cantidad > 0
cantidadElementos++;
int opcion;
int menu = 0;
!menu
printf("\n--- Menú de Inventario ---\n");
printf("1. Agregar elemento\n");
printf("2. Retirar elemento\n");
printf("4. Calcular el valor total del inventario\n");
printf("5. Realizar operaciones matemáticas en precios/cantidades\n");
printf("6. Ordenar elementos del inventario\n");
printf("7. Guardar y salir del programa\n");
opcion = validarNumeros("Seleccione una opción: ", 1, 7);
opcion
case 1:
agregarElemento();
break;
case 2:
retirarElemento();
break;
case 3:
mostrarInventario();
break;
case 4:
calcularValorTotal();
break;
case 5:
realizarOperacionesMatematicas();
break;
case 6:
printf("\n--- Opciones de Ordenamiento ---\n");
printf("1. Por nombre\n");
printf("2. Por cantidad\n");
int opcionOrdenamiento = validarNumeros("Seleccione una opción de ordenamiento: ", 1, 3);
ordenarInventario(opcionOrdenamien
break;
case 7:
archivo = fopen(ARCHIVO_INVENTARIO, "wb");
archivo != NULL
fwrite(inventario, sizeof(Producto), CAPACIDAD_MAXIMA, archivo);
fclose(archivo);
printf("Guardando el inventario y saliendo del programa.\n");
menu = 1;
break;
!menu
return 0;
FIN
void agregarElemento();
cantidadElementos < CAPACIDAD_MAXIMA
Producto nuevoProducto;
printf("Ingrese el nombre del producto: ");
fgets(nuevoProducto.nombre, sizeof(nuevoProducto.nombre), stdin);
nuevoProducto.nombre[strcspn(nuevoProducto.nombre, "\n")] = '\0';
nuevoProducto.cantidad = validarNumeros("Ingrese la cantidad a agregar: ", 1, 1000);
nuevoProducto.precio = validarNumeros("Ingrese el precio del producto: ", 0, 1000000);
inventario[cantidadElementos] = nuevoProducto;
cantidadElementos++;
printf("Producto agregado al inventario.\n");
printf("El inventario está lleno. No se pueden agregar más productos.\n");
FIN
void mostrarInventario()
printf("\n--- Inventario ---\n");
int i = 0; i < cantidadElementos; i++
printf("Producto %d:\n", i + 1);
printf(" Nombre: %s\n", inventario[i].nombre);
printf(" Cantidad: %d\n", inventario[i].cantidad);
printf(" Precio: %.2f\n", inventario[i].precio);
FIN


void retirarElemento()
cantidadElementos > 0
char nombreBuscado[30];
int cantidad;
printf("Ingrese el nombre del producto a retirar: ");
fgets(nombreBuscado, sizeof(nombreBuscado), stdin);
nombreBuscado[strcspn(nombreBuscado, "\n")] = '\0';
int indiceProducto = -1;
int i = 0;
i < cantidadElementos;
i++
strcmp(inventario[i].nombre,
nombreBuscado) == 0
indiceProducto = i;
break;
indiceProducto != -1
int i = indiceProducto;
i < cantidadElementos - 1;
i++
inventario[i] = inventario[i + 1];
cantidadElementos--;
printf("Producto no encontrado en el inventario.\n");
printf("El inventario está vacío. No se pueden retirar productos.\n");
FIN


void ordenarInventario(int opcionOrden)
opcionOrden
case 1
case 2
case 3
qsort(inventario, cantidadElementos, sizeof(Producto), compararProductosPorNombre);
qsort(inventario, cantidadElementos, sizeof(Producto), compararProductosPorCantidad);
qsort(inventario, cantidadElementos, sizeof(Producto), compararProductosPorPrecio);
printf("Inventario ordenado por nombre.\n");
printf("Inventario ordenado por cantidad.\n");
printf("Inventario ordenado por precio.\n");
break;
break;
break;
FIN


int compararProductosPorNombre(const void *a, const void *b)
return strcmp((Producto *)a)->nombre, ((Producto *)b)->nombre);
FIN
int compararProductosPorCantidad(const void *a, const void *b)
return ((Producto *)a)->cantidad - ((Producto *)b)->cantidad;
FIN
int compararProductosPorPrecio(const void *a, const void *b)
float diff = ((Producto *)a)->precio - ((Producto *)b)->precio;
return (fabs(diff) < EPSILON) ? 0 : (diff > 0 ? 1 : -1);
FIN


void calcularValorTotal()
float valorTotal = 0;
int i = 0;
i < cantidadElementos;
i++
valorTotal += inventario[i].cantidad * inventario[i].precio;
printf("\nEl valor total del inventario es: %.2f\n", valorTotal);
FIN


void realizarOperacionesMatematicas()
int opcionOperacion;
printf("\n--- Opciones de Operaciones Matemáticas ---\n");
printf("1. Aumentar precios\n");
printf("2. Reducir precios\n");
printf("3. Aumentar cantidades\n");
printf("4. Reducir cantidades\n");
opcionOperacion = validarNumeros("Seleccione una opción de operación matemática: ", 1, 4);
float porcentaje;
opcionOperacion
case 1
case 2
case 3
case 4
porcentaje = validarNumeros("Ingrese el porcentaje de aumento de precios: ", 0, PORCENTAJE_MAXIMO);
porcentaje = validarNumeros("Ingrese el porcentaje de reducción de precios: ", 0, PORCENTAJE_MAXIMO);
porcentaje = validarNumeros("Ingrese el porcentaje de aumento de cantidades: ", 0, PORCENTAJE_MAXIMO);
porcentaje = validarNumeros("Ingrese el porcentaje de reducción de cantidades: ", 0, PORCENTAJE_MAXIMO);
int i = 0;
i < cantidadElementos;
i++
inventario[i].precio *= (1 + porcentaje / 100);
inventario[i].precio *= (1 - porcentaje / 100);
inventario[i].cantidad *= (1 + porcentaje / 100);
inventario[i].cantidad *= (1 - porcentaje / 100);
printf("Precios aumentados en %.2f%%.\n", porcentaje);
printf("Precios reducidos en %.2f%%.\n", porcentaje);
printf("Cantidades aumentadas en %.2f%%.\n", porcentaje);
printf("Cantidades reducidas en %.2f%%.\n", porcentaje);
break;
break;
break;
break;
FIN

miro