Ejercicio 1: Sistema de Gestión de Estudiantes
Descripción: Crea un programa que gestione la información de estudiantes en una clase. Cada estudiante tiene un nombre, un número de identificación (ID), y una nota promedio. El programa debe permitir:

1. **Agregar estudiantes.**
2. **Mostrar la información de todos los estudiantes.**
3. **Mostrar la información del estudiante con la mejor nota promedio.**

Instrucciones:

1. **Crea una clase Estudiante con los atributos nombre, id, y notaPromedio.**
2. **Implementa métodos en la clase para:**
    - **Establecer el nombre, ID, y nota promedio del estudiante.**
    - **Obtener y mostrar la información del estudiante.**
3. **En la función main, crea un arreglo de objetos Estudiante.**
4. **Agrega varios estudiantes al arreglo.**
5. **Implementa una función que recorra el arreglo para encontrar al estudiante con la mejor nota promedio.**
6. **Muestra la información de todos los estudiantes y del estudiante con la mejor nota promedio.**

Ejercicio 1 V2: Sistema Dinámico de Gestión de Estudiantes
Descripción: Crea un programa que gestione la información de estudiantes en una clase. Permite al usuario ingresar la cantidad de estudiantes y sus datos, y luego muestra la información de todos los estudiantes y del estudiante con la mejor nota promedio.
Instrucciones:

1. **Crea una clase Estudiante con los atributos nombre, id, y notaPromedio.**
2. **Implementa métodos en la clase para:**
    - **Establecer el nombre, ID, y nota promedio del estudiante.**
    - **Obtener y mostrar la información del estudiante.**
3. **En la función main, utiliza un arreglo dinámico de objetos Estudiante para gestionar la cantidad de estudiantes ingresados por el usuario.**
4. **Permite al usuario ingresar los datos de los estudiantes.**
5. **Implementa una función que recorra el arreglo para encontrar al estudiante con la mejor nota promedio.**
6. **Muestra la información de todos los estudiantes y del estudiante con la mejor nota promedio.**

Ejercicio 2: Sistema Básico de Gestión de Inventario
Descripción: Crea un programa que gestione un inventario de productos. Cada producto tiene un nombre, un precio, y una cantidad en stock. El programa debe permitir agregar productos al inventario y luego mostrar los detalles de todos los productos.
Requisitos:

- Crea una clase Producto con los atributos nombre, precio, y cantidad.
- Usa punteros para manejar un arreglo dinámico de objetos Producto.
- Implementa una función que agregue un nuevo producto al inventario.
- Implementa una función que imprima los detalles de todos los productos.

NO HACER

Ejercicio 3: Sistema de Gestión de Proyectos

Descripción: Crea un sistema de gestión de proyectos donde cada proyecto tiene varias tareas y cada tarea puede ser asignada a diferentes empleados. Deberás crear las siguientes clases:

1. **Clase Empleado:**
    - **Atributos:**
        - nombre (cadena)
        - id (entero)
    - **Métodos:**
        - asignarTarea(Tarea* tarea): Añade una tarea a la lista de tareas asignadas al empleado.
        - mostrarTareas(): Muestra todas las tareas asignadas al empleado.
2. **Clase Tarea:**
    - **Atributos:**
        - descripcion (cadena)
        - duracion (entero, en horas)
        - empleadosAsignados (arreglo dinámico de punteros a Empleado)
    - **Métodos:**
        - asignarEmpleado(Empleado* empleado): Añade un empleado al arreglo de empleados asignados.
        - mostrarEmpleados(): Muestra todos los empleados asignados a la tarea.
3. **Clase Proyecto:**
    - **Atributos:**
        - nombre (cadena)
        - tareas (arreglo dinámico de punteros a Tarea)
    - **Métodos:**
        - agregarTarea(Tarea* tarea): Añade una tarea al arreglo de tareas del proyecto.
        - mostrarTareas(): Muestra todas las tareas del proyecto con los empleados asignados a cada una.

Ejercicio Extra: Sistema de Gestión de Productos en una Tienda

Descripción: Crea un programa que gestione un inventario de productos en una tienda. Cada producto tiene un nombre, un código de producto, y un precio. El programa debe permitir:

1. Agregar productos al inventario.
2. Mostrar la información de todos los productos.
3. Actualizar el precio de un producto dado su código.
4. Mostrar el producto más caro del inventario.

Requisitos:

1. **Clase Producto:**
    - **Atributos:** nombre (string), codigo (int), precio (float).
    - **Métodos:**
        - establecerDatos: Establece los atributos del producto.
        - mostrarDatos: Muestra la información del producto.
        - actualizarPrecio: Actualiza el precio del producto.
        - obtenerPrecio: Devuelve el precio del producto.
        - obtenerCodigo: Devuelve el código del producto.
2. **Funciones:**
    - agregarProducto: Agrega un nuevo producto al arreglo dinámico.
    - mostrarInventario: Muestra la información de todos los productos.
    - actualizarPrecioPorCodigo: Actualiza el precio de un producto dado su código.
    - encontrarProductoMasCaro: Encuentra y muestra el producto más caro del inventario.
3. **En la función main:**
    - Permite al usuario ingresar la cantidad de productos y luego sus datos.
    - Permite actualizar el precio de un producto dado su código.
    - Muestra la información de todos los productos y el producto más caro.