Principios SOLID

principios SOLID

PONDERACIÓN: 1.5 (5% de 30 pts)

Horas Aproximadas: 6

Universidad San Carlos de Guatemala

Facultad de ingeniería.

Ingeniería en ciencias y sistemas

Índice

Competencias

Aplica los principios SOLID de diseño orientado a objetos para desarrollar soluciones de software modulares, mantenibles y escalables, favoreciendo la reutilización del código y la calidad del desarrollo mediante el uso adecuado de responsabilidades, abstracciones, interfaces y dependencias.

Objetivos

General

- Aplicar los conocimientos adquiridos a lo largo de la carrera de Ingeniería en Ciencias y Sistemas para generar software de alta calidad y escalable, a través de diferentes técnicas de desarrollo y utilizando las últimas tecnologías.

Específicos

- Que el estudiante haga uso y comprenda por completo los conceptos sobre código limpio.
- Que el estudiante se familiarice con la aplicación de buenas prácticas de programación y los principios SOLID.

Descripción / Enunciado

El estudiante debe crear un programa que permita gestionar un inventario de productos en una tienda. El programa debe permitir al usuario agregar, eliminar y consultar productos del inventario. Los productos tienen los siguientes atributos:

- **Nombre** (string)
- **Cantidad** (entero)
- **Precio** (decimal)

El programa debe permitir realizar las siguientes acciones:

1. Agregar un nuevo producto al inventario.
2. Eliminar un producto del inventario.
3. Mostrar la lista de productos con sus detalles.
4. Mostrar la lista de productos ordenados por precio y/o cantidad

5. Buscar un producto por su nombre y mostrar su información.

Como requisitos mínimos, el código deberá cumplir con los siguientes puntos de buenas prácticas en el código:

1. **Modularización**: El código debe estar organizado en funciones claras que realicen tareas específicas.
2. **Nombres descriptivos**: Las variables, funciones y clases deben tener nombres claros que reflejen su propósito.
3. **Comentarios**: El código debe estar comentado adecuadamente para explicar el propósito de cada parte del código.
4. **Manejo de errores**: El programa debe ser capaz de manejar situaciones como la entrada de datos incorrectos o la eliminación de un producto que no exista.
5. **Código limpio**: El código debe seguir buenas convenciones de formato (por ejemplo, sangrías consistentes, líneas en blanco donde sea necesario).

Tomar en cuenta que no es necesario la creación de una base de datos, todo debe de ser trabajado en memoria volátil, sin embargo, debe de ser con una estructura de datos, la cual el estudiante elegirá según crea conveniente (**para esto, el uso de librerías está permitido**).

Documentación

Como documentación, se debe de crear un README usando el lenguaje Markdown en el cual explique con sus palabras los principios SOLID vistos en clase e incluyan ejemplos con código FUNCIONAL

Entregables

- Subir a UEDI el enlace del repositorio.
- Debe de crear un repositorio privado con el siguiente formato de nombre:
    - Prácticas-SA-<<SECCIÓN>>-<<CARNE>>
    - Crear carpeta dentro del repositorio con el nombre **P1** e incluir los archivos a entregar
- Agregar al auxiliar al repositorio, con el rol Developer:
    - Sección A:
    - Sección B: **di3gini**

Consideraciones

- Se debe hacer uso de un repositorio en la nube para realizar la entrega de su proyecto (Gitlab, Github, Bitbucket, etc.)
- El lenguaje a utilizar es de elección libre según lo requiera el estudiante.
  Aun así, se recomienda el uso de javascript/typescript o python.
- Se revisará el historial de commits para corroborar que no fue hecho todo en un único commit
- No es necesario la implementación de un frontend como tal
- Se trabajará de manera individual.
- Las copias completas/parciales serán merecedoras de una nota de 0 puntos, los responsables serán reportados al catedrático de la sección y a la Escuela de Ciencias y Sistemas.

Cronograma

El cronograma permite planificar y organizar las diferentes etapas del desarrollo de la práctica

Ejemplo

Tarea	Fecha
Asignacion practica	03/08/2025
Fecha de entrega	07/08/2025
Fecha de calificación	08-09/08/2025

Rúbrica de Calificación

La evaluación de la práctica busca medir el cumplimiento de los objetivos planteados, así como la correcta aplicación de los conocimientos técnicos y habilidades.

Se debe agregar una tabla con los aspectos a calificar con su respectiva puntuación.

Descripcion	Punteo
Funcionalidades	20
Agregar al inventario	5
Eliminar	5
Mostrar la lista de productos	2
Ordenar la lista de productos por precio y/o cantidad	3
Buscar un producto por su nombre	5
Buenas prácticas de código	40

Modularización	10
Nombres descriptivos	15
Comentarios	5
Manejo de errores	10
Código limpio	10
Principios SOLID y sus ejemplos	30
Single Responsibility Principle	6
Open-Closed Principle	6
Liskov Substitution Principle	6
Interface Segregation Principle	6
Dependency Inversion Principle	6
Preguntas	10
Pregunta 1	5
Pregunta 2	5

Valores

En el desarrollo de la práctica, se espera que cada estudiante demuestre honestidad académica y profesionalismo. Por lo tanto, se establecen los siguientes principios:

1. **Originalidad del Trabajo**
    - Cada estudiante o equipo debe desarrollar su propio código y/o documentación, aplicando los conocimientos adquiridos en el curso.
2. **Prohibición de Copias y Plagio**
    - Si se detecta la copia total o parcial del código, documentación o cualquier otro entregable, la calificación será de **0 puntos**.
    - Esto incluye la reproducción de código entre compañeros, la reutilización de proyectos de semestres anteriores o el uso de código externo sin la debida referencia.
3. **Uso Responsable de Recursos Externos**
    - El uso de bibliotecas, frameworks y ejemplos de código externos está permitido, siempre y cuando se referencian correctamente y se comprendan plenamente. (Consultar con el catedrático su política)
4. **Revisión y Detección de Plagio**
    - Se podrán utilizar herramientas automatizadas y revisiones manuales para identificar similitudes en los proyectos.
    - En caso de sospecha, el estudiante deberá justificar su código y demostrar su desarrollo individual o en equipo. Si este extremo no es comprobable la calificación será de **0 puntos**.

Al detectarse estos aspectos se informará al catedrático del curso quien realizará las acciones que considere oportunas.

Referencias

Los principios SOLID de programación orientada a objetos explicados en Español sencillo