Universidad Nacional
Escuela de Informática
EIF 201-Programación I

Proyecto #2

Prof. Jennifer Fuentes Bustos
Prof . Maikol Guzmán Alán
Prof . Marianela Solano Oreas
Prof. Karol Leitón Arrieta

Fecha de entrega: Viernes 1 de noviembre de 2019
Modo de entrega: Aula Virtual Institucional
Modo de trabajo: Individual o en parejas

Objetivos

- Aplicar los conceptos teóricos, principios y técnicas estudiadas en clase.
- Investigar e incorporar conceptos y técnicas.
- Desarrollar código correctamente estructurado, debidamente encapsulado, eficiente y fácilmente reutilizable

Generalidades

- El código del proyecto debe estar estructurado adecuadamente, respetando los principios de diseño estudiados en clase.

- Las clases deben separar correctamente la declaración de la interfaz y su implementación, usando archivos de cabecera (archivos .h) y código fuente (archivos.cpp) por aparte. Los archivos de cabecera deberán siempre usar guardas (no se permite el uso de la directiva #pragma once cuando se utilice Visual Studio).

- Las clases de entidad no deberán contener código de entrada/salida (como salida a la consola usando cout, por ejemplo)

- El manejo de la interfaz con el usuario debe hacerse por medio de una o varias clases diseñadas para tal efecto.

- La interfaz debe ser diseñada en forma agradable, entendible y fácil de utilizar por **cualquier usuario.**, recuerde que si el usuario no entiende el sistema posiblemente se trate de un problema de diseño por parte del programador. Pruebe su programa con otros usuarios, esto para recibir retroalimentación y asegurarse que la interfaz sea clara y entendible.

- El programa debe tener una interfaz clara y un adecuado e intuitivo manejo de menús y submenús, que faciliten el uso de la interfaz, de los contrario el director del hospital no aprobará la compra del sistema.

- El sistema debe tener una adecuado y excelente manejo y control de errores, el cual no permita incluir datos diferentes al que se está solicitando, si el programa se "cae" el director del Hospital rechazará la compra del sistema en forma definitiva y buscará otro desarrollador de software.

- Debe presentar una funcionalidad correcta de los módulos mediamente una codificación eficiente y bien estructurada.

- Debe presentarse un establecimiento correcto de relaciones (por ejemplo: no se puede ingresar una cama si no existe un pabellón , no se puede ingresar un paciente si no existe una cama o no se puede crear un doctor si no existe la especialidad)

- Incluir UML de clases, el cual concuerde con el código (no tiene puntaje per debe entregarse)

Descripción

Usted ha sido contratado por el gimnasio "Vital Salud", el cual requiere un sistema que le permita administrar los socios, instructores, clases y rutinas.

El proceso de inscripción de cada socio consiste en registrar su número de cédula, nombre completo, correo electrónico, teléfono y la fecha de inscripción.

Una vez inscrito un nuevo socio se le asigna un instructor quien será el encargado de llevar el seguimiento de su evolución. El instructor debe registrar los datos de peso, porcentaje de masa muscular y porcentaje de grasa corporal, además será el encargado de crear y administrar la rutina del socio. La rutina depende de las necesidades del cliente, entre ellas aumentar porcentaje de masa muscular o disminuir porcentaje de grasa corporal. Para cada rutina se tiene la siguiente información: una codificación automática, fecha de creación, fecha estimada de finalización, objetivo, instrucciones de la rutina con sus respectivas series y repeticiones.

El instructor debe registrar cada rutina creada y al inicio de cada mes debe registrar los resultados de cada socios en cuanto a: peso, porcentaje de masa muscular y porcentaje de grasa corporal.

El gimnasio ofrece clases grupales, para cada clase se registra una codificación automática, nombre, instructor, salón y horario . Los socios tienen derecho a matricularse en las clases grupales que deseen.

El gimnasio cuenta con "n" **salones** para impartir las diferentes clases, cada salón tiene un nombre, código y capacidad. En cuanto a la asignación de salones estos deben asignarse evitando choques de horario entre clases.

El sistema requiere que todos los datos de la aplicación puedan ser recuperados desde un archivo de texto para evitar la pérdida de datos cuando se cierra la aplicación. También necesita que seleccione las estructuras de datos más apropiadas para la gestión de los datos.

Implemente las siguientes funcionalidades:

1. Contratación de instructores, debe visualizarse la lista de instructores (3pts)
2. Ingreso de socios y asignación de instructor (5pts)
3. Lista general de socios (datos básicos) (3pts)
4. Asignación de rutina a socio(3 pts)
5. Visualización de información de socio específico, detallando: datos básicos, instructor, clases grupales, historial general de rutinas. (14pts)
6. Visualización de rutinas vencidas por instructor, detallando: código de rutina, fecha de vencimiento y nombre del socio(6pts) // esto con el fin de que el instructor pueda tener una alerta para renovar rutinas vencidas
7. Búsqueda y visualización detallada de rutinas especificas, se muestra el detalle de la rutina deseada por socio (6pts)
8. Control de avance en cuanto a peso, IMC, porcentaje de grasa etc. implica datos histórico con sus respectivas fechas (6pts)
9. Listado de socios ordenados según resultados, esto con el fin de premiar los mejores (6pts)
10. Listado general de socios por instructor (7pts)
11. Instructor con mejores resultados con sus socios (4pts) // esto con el fin de poder premiar a los mejores instructores

12. Clases Grupales:
    - Creación de clases grupales (requiere instructor, salón y selección de horario) (4pts)
    - Visualización detallada de clases específicas (nombre, capacidad, campos disponibles, instructor y horario)(3pts)
    - Matricular socios en clases específicas(3pts)
    - Visualización de clases por salón(3pts)
    - Visualización socios por clase(4pts)
13. Se debe permitir guardar y recuperar de archivos toda la información anterior (20pts)

Nota:
En cada ítem se evalúa funcionalidad y eficiencia del código

Observaciones generales:

- Cada profesor define el IDE y la versión del mismo en que se deberá entregar el proyecto.
- El medio oficial de entrega será el aula virtual institucional.
- Se revisa el archivo subido al aula virtual, por lo cual verifique bien haber subido la versión correcta pues no se recibirá ningún otro archivo o versión después de la fecha de entrega.
- Queda a criterio del profesor solicitar defensa del proyecto.
- No se admitirá la entrega tardía del proyecto, en este caso, se calificará con nota 0 (cero).
- Los proyectos deben entregarse con código fuente completo y compilable así como su respectivo UML en pdf.
- Cualquier entrega que no sea de elaboración original del estudiante o haya sido copiado o adaptado de otro origen (plagio), de manera parcial o total, se calificará con nota 0 (cero) y se podrá proceder como lo indiquen los reglamentos vigentes de la universidad para elevar el caso.
- **El código del programa debe compilar correctamente, si el código no puede compilarse, el trabajo será calificado con una nota de 0 (cero).**
- Se evalúa la funcionalidad y eficiencia del programa, si algo NO funciona NO puede ser eficiente, por tanto lo que no funciona no se evalúa.