Proyecto Semestral

Clave/Asignatura: ICI3241 - Programación Avanzada
INF2241 – Programación Orientada a Objetos
Profesor: Claudio Cubillos Figueroa - claudio.cubillos@pucv.cl
Periodo: Semestre II —2021
Fecha: Valparaíso, 09 de agosto de 2021

Objetivo

El presente proyecto tiene por objeto que los alumnos implementen una aplicación informática integrando los contenidos vistos en el curso y en los talleres de ayudantía. El proyecto debe cumplir con los requisitos definidos en este enunciado.

Requisitos generales del sistema

- El proyecto considera el desarrollo de un producto de software que dé respuesta a una necesidad de la sociedad.
- Este software deberá ser implementado en su totalidad por el grupo de trabajo.
- El problema a abordar debe ser escogido y modelado por los propios alumnos de 3 personas.
- Excepcionalmente podrán haber grupos de 2 o 4 alumnos, previa revisión y autorización del profesor. Para ello los alumnos deberán enviar la solicitud y antecedentes antes de la fecha límite de registro de grupos.
- Toda entrega se realizará por el aula virtual.

Elección de temas:

- Los alumnos deben cuidar de no seleccionar temas repetidos de otros semestres.
- Si usted está al menos por segunda vez cursando la asignatura podrá reutilizar el proyecto del semestre pasado. En caso de que exista más de un integrante interesado en reutilizar el proyecto, pero se encuentren en grupos separados, ninguno podrá reutilizar el proyecto. Esta reutilización queda sujeto a previa evaluación del profesor.

Entregas de Proyecto

El proyecto consta de distintos entregables. Existen 4 entregas parciales (EPxx) y 3 entregas acumulativas (Parte A, B y Final):

- Desarrollar el proyecto sobre la plataforma Netbeans 12
- El Proyecto deberá ser desarrollado bajo el lenguaje de JAVA.

Entrega Parcial 1 (EP1)

EP1.1 Realizar un análisis de los datos a utilizar y principales funcionalidades a implementar que dan sentido a la realización del proyecto.
EP1.2 Diseño conceptual de clases del Dominio y su código en Java
EP1.3 Todos los atributos de todas las clases deben ser privados y poseer sus respectivos métodos de lectura y escritura (getter y setter).
EP1.4 Se deben incluir datos iniciales dentro del código.

- **(opcional)** Leer datos desde archivo/base de datos.

Entrega Parcial 2 (EP2)

EP2.1. Diseño conceptual y codificación de 2 (dos) niveles de anidación de colecciones de objetos.
EP2.2. Diseño conceptual y codificación de 2 (dos) clases que utilicen sobrecarga de métodos.
EP2.3. Diseño conceptual y codificación de al menos 1 clase mapa del JCF.

Entrega Parcial 3 (EP3)

EP3.1. Diseño de diagrama de clases UML de lo considerado hasta la EP3

EP3.2. Se debe hacer un menú para el Sistema donde ofrezca las funcionalidades de: 1) Inserción Manual / agregar elemento y 2) Mostrar por pantalla listado de elementos. Esto para cada una de las 2 colecciones anidadas.

Parte A (entrega acumulada)

PA.1. La integración de todo lo entregado anteriormente (entregas Parciales) más lo detallado a continuación.

PA.2. Se debe hacer un menú para el Sistema donde ofrezca las funcionalidades de: 1) Edición o modificación del elemento y 2) Eliminación del elemento. Esto para cada una de las 2 colecciones anidadas.

- **(opcional)** Implementar la funcionalidad de buscar elemento en 1 o más niveles.

PA.3. Se debe generar un reporte en archivo txt que considere mostrar datos de las 2 colecciones anidadas (ej: csv).

- **(opcional)** Se puede generar un archivo de salida de planilla de cálculo (.xls o .xlsx)
- **(opcional)** Se puede utilizar un componente gráfico estadístico (JFreeChart u otro) en ventana.

PA.4. El código fuente debe estar bien modularizado de acuerdo a lo descrito en el informe además de seguir las buenas prácticas de documentación interna y legibilidad.

PA.5. Todas las funcionalidades pueden ser implementadas mediante consola.

- **(opcional)** La implementación de una o más interfaces gráficas de usuario, con la inclusión de ventanas con componentes AWT, SWING, JAVAFXu otro.

PA.6 Utilizacion de GitHub (Realización de al menos 3 Commit)

Entrega Parcial 4 (EP4)

EP4.1. Se deben incluir al menos 2 funcionalidades propias que sean de utilidad para el negocio (distintas de la inserción, edición, eliminación y reportes). Específicamente:

- Seleccionar un objeto por criterio, considera la selección de un objeto basado en un criterio específico, involucrando dos o más colecciones anidadas. Por ejemplo, selección del alumno con la nota final más baja de todos los cursos, o seleccionar el pasajero más joven de todos los buses de una compañía.
- Subconjunto filtrado por criterio: considera la selección de un subconjunto de objetos basado en un criterio específico, involucrando dos o más colecciones anidadas. Por ejemplo, selección de los alumnos con nota final entre 4,0 y 7,0 de entre todos los cursos; o seleccionar a todos los pasajeros que tengan asiento impar de entre todos los buses de la compañía.

EP4.2. Diseño y codificación de 2 (dos) clases que utilicen sobreescritura de métodos.

EP4.3. Diseño y codificación de 1 (una) clase abstracta que sea padre de al menos 2 (dos) clases. La clase abstracta debe ser utilizada por alguna otra clase (contexto)

EP4.4. Diseño y codificación de 1 (una) interfaz que sea implementada por al menos 2 (dos) clases. La interfaz debe ser utilizada por alguna otra clase (contexto)

- **(opcional)** generar documentación con Javadoc.

Parte B (entrega acumulada)

PB.1. La integración de todo lo entregado anteriormente (entregas parciales y Parte A) más lo detallado a continuación.

PB.2. Se deben implementar al menos 3 ventanas gráficas (GUIs en AWTo SWING): 1 ventana de menú, 1 ventana de agregar elemento y 1 ventana de listar elementos.

PB.3. Se debe aplicar encapsulamiento y principios OO

PB.4 Continuidad en la utilización de GitHub (Realización de al menos 3 Commit adicionales a los ya hechos en la parte A)

Entrega Final (entrega acumulada)

EF.1. La integración de todo lo entregado anteriormente (entregas parciales, Parte Ay B) más lo detallado a continuación.

EF.2. Implementar el manejo de excepciones capturando errores potenciales específicos mediante Try-catch

EF.3. Crear 2 clases que extiendan de una Excepción y que se utilicen en el programa.

EF.4. Aplicación del patrón de diseño Strategy (Estrategia)

- **(opcional)** Reemplazar los datos iniciales en el código por una conexión con base de datos local (MySQL), archivo de texto CSVo Excel, para la persistencia de datos.
- **(opcional)** Considerar la implementación del patrón Modelo-Vista-Controlador (MVC) en la arquitectura del sistema.

Entregas:

- El Proyecto debe ser subido al aula virtual hasta las 23:55 hrs. de las fechas de entrega planteadas:
- Instrucciones de instalación/ejecución del proyecto.
- Informe Escrito
- Código fuente (los .java o el proyecto Netbeans).
- Los demás archivos necesarios (de texto, esquemas sql u otros pertinentes).

Informe escrito:

- Corresponde a un reporte ejecutivo.
- Va respondiendo punto por punto (EP1.1 a EF.3) a cada uno de los requisitos solicitados, indicando:
  - cómo ese requerimiento fue aplicado al proyecto
  - dónde se encuentra evidencia de lo implementado (en un diagrama, archivo .java, clase, No de línea de código)

Defensa del proyecto

No se considerará defensa del proyecto, debido a que se realiza en línea. Sin embargo, los ayudantes podrán solicitar información adicional sobre el proyecto para su correcta evaluación.