ENUNCIADOS EJERCICIOS FINALES PARADIGMAS DE PROGRAMACIÓN

Objetivo

Validar y evaluar los conocimientos en el diseño y la aplicación de los paradigmas de programación, vistos durante el semestre.

Aspectos para la entrega:

1. Máximo 2 personas por grupo
2. Fechas de entrega:
   1. El diagrama del diseño, Viernes 31 de Octubre .
   2. El proyecto completo. Grupo POO: Jueves 13 de Noviembre.
3. Forma de entrega: Ambos integrantes explican el diagrama, la implementación y demuestran el funcionamiento.
4. Tienen que asistir los integrantes del proyecto, el que no venga la evaluación es 0.0

Rubrica de Evaluación PROYECTO FINAL.

|  |  |
| --- | --- |
| Diseño  Diagrama de clases se presenta en modo Extendido:   * Correcto nombrado de clases incluyendo herencias y abstractas, Correcto nombrado de interfaces * Relaciones bien nombradas * Direccionalidad * Multiplicidad * Roles * Atributos con tipo y ocultamiento * Correcta implementación de la multiplicidad con los atributos involucrados * Constructor(es) con parámetros * Métodos con tipo, parámetros, ocultamiento, overrides, virtual o static * Herencia, polimorfismo * Interfaces * Eventos * Aspectos * Servicios * Inyección de Interfaces | 50 PUNTOS – Equivale al 20% del 3er parcial |

Implementación Biblioteca de clases con elementos de programación funcional y la interfaz en C#:

ES OBLIGATORIO TENER EL DISEÑO EVALUADO, PARA ACCEDER A LO SIGUIENTE:

|  |  |
| --- | --- |
| Métodos   * Manejo de Errores * Correcto retorno * No impresión desde los métodos | 2 puntos |
| Uso atributos estáticos | 2 puntos |
| Carga de datos para la instancia de objetos desde archivos | 2 puntos |
| Código completamente funcional | 4 puntos |
| Funcionamiento Correcto   * Lógica y funcionamiento de métodos * El programa no aborta o termina la ejecución abruptamente * El programa funciona como se solicitó | 6 puntos |
| Implementa 4 eventos, que notifican al usuario hechos importantes | 2 puntos por evento (8 puntos en total) |
| Implementa los siguientes aspectos:  Validación de información: acá se debe simular que se va a guardar en una base de datos, el aspecto debe simplemente validar la información y si es correcta presentar un mensaje correspondiente, de la misma forma si no es correcta.  Autenticación a la aplicación | 3 Puntos por aspecto (6 puntos en total) |

Hasta acá van 30 puntos, una biblioteca funcionando correctamente. Para demostrar esto deben presentar mínimo un ensamblado que incorpore la biblioteca y desde el program.cs se haga una demostración por medio de instrucciones, que la biblioteca funciona de acuerdo con el enunciado.

Implementación de la biblioteca en un proyecto MVC con interface Web. 20 puntos restantes. El proyecto debe ser similar al mostrado en clase.

SI HICIERON BIEN LO ANTERIOR Y SACARON MÍNIMO 40 PUNTOS, PUEDEN ACCEDER A LO SIGUIENTE:

La implementación adicional de los Eventos completos en el proyecto MVC, le permite subir adicional 0,3 a una sola nota que escoja entre estas: quices 1 y 2 o de los parciales 1 y 2.

La implementación adicional de los Aspectos completos en el proyecto MVC, le permite subir otro 0,3 adicional a una sola nota que escoja entre estas: quices 1 y 2 o de los parciales 1 y 2.

**Ejercicio 1**

Diseñe un diagrama UML e Implemente las siguientes clases en C# que se relacionan en un taller de automóviles: carros, reparaciones, repuestos.

Los carros tienen los siguientes atributos: placa, marca, modelo, año y el dueño del carro (para el caso del ejercicio, uno solo).

Las reparaciones tienen los siguientes atributos: el carro, fecha, la lista de los repuestos que utilizaron, el valor de la reparación y la lista de los mecánicos que atendieron la reparación (puede ser uno o varios mecánicos).

El taller lleva la lista de reparaciones hechas a los carros de los clientes. Cada que se hace una reparación, si el cliente tiene crédito, se suma al saldo que debe el cliente el valor de esa reparación.

El taller debe tener una operación para que el cliente independiente si tiene o no crédito, cancele la reparación. Considere que la lógica es distinta en cada caso.

Los repuestos tienen nombre, proveedor que lo suministra, fecha de compra.

En el taller hay dos tipos de personas: los clientes y los mecánicos. Ambos comparten la siguiente información: id, nombre, teléfono. Únicamente los mecánicos tienen como atributo la especialidad y los clientes tienen un atributo para saber si tienen crédito o no, y el saldo que le deben al taller.

Los carros pueden ser eléctricos, en este caso se tiene como atributo la autonomía; híbridos, en este caso se tiene como atributo el número de baterías; y a gasolina, en este caso se tiene como atributo el número de cilindros.

El taller ofrece una reparación denominada puesta a punto la cual es diferente para cada tipo de vehículo:

Para los autos a gasolina se cambian las correas, bujías se drena el radiador y se le ajustan los inyectores.

Para los autos híbridos, se lubrica el motor generador eléctrico, se cambian los frenos y se calibran recargan las baterías, se examina desgaste de llantas, se examina amortiguación, se cambia el líquido de frenos.

Para los autos eléctricos, se comprueba el estado de la batería, se cambian lubricantes, se examina desgaste de llantas, se examina amortiguación, se cambia el líquido de frenos

Eventos: Cuando un auto se le dé ingreso, debe generar un evento para que la clase que va a hacer los trabajos sepa que debe actuar. Cuando se termine el trabajo en un carro, se debe generar un evento para informar a la clase que va a facturar que ya lo puede hacer. Cuando se cancele la factura, se debe enviar un evento a la clase encargada para permitir la salida del vehículo.

**Ejercicio 2**

Diseñe un diagrama UML e Implemente las siguientes clases en C# que se relacionan en una universidad: Estudiantes, Historia Académica, Asignaturas y docentes.

Los estudiantes y docentes tienen los atributos Nombre, Id, Fecha Nacimiento

Solamente los estudiantes tienen el atributo Nombre Acudiente.

Y Los docentes tienen como atributo una lista de los títulos que tiene y adicionalmente, el área donde dicta clase.

Las asignaturas tienen nombre, el grado en que se dicta y la lista de docentes que las puede dictar.

Cada evaluación tiene los atributos: estudiante, asignatura y nota final.

La historia académica tiene el año, el período y la lista de evaluaciones.

Las evaluaciones son reportadas por los docentes. Los estudiantes pueden consultar las evaluaciones que vaya haciendo los docentes. La universidad deber tener una operación para recorrer las evaluaciones y saber: cuántas estudiantes han ganado y perdido evaluaciones, cuál es la asignatura que más pierden y cuál en la que les va mejor.

Finalmente, represente estudiantes, asignaturas, docentes y evaluaciones.

Los estudiantes pueden ser de posgrado y pregrado. Los estudiantes de pregrado tienen como atributo una matriz hasta con 5 estudios previos y con la siguiente información: en la primera fila el pregrado, otros posgrados hechos. Las filas de esta matriz son 3: institución, título, año de graduación. Para los estudiantes de pregrado, se tiene un vector y en cada celda la siguiente información nombre del colegio donde egresó de bachillerato, año, título del diploma.

La universidad tiene unos requisitos de grado que son distintos para los estudiantes de pregrado y de posgrado. El método debe devolver un string, separado por | y la información será de acuerdo al tipo de estudiante.

Los estudiantes de pregrado pueden hacer práctica, pasantía investigativa o plan de negocios. Esta práctica tiene un monitor y una nota cuantitativa entre 0 y 5. El string debe devolver estos tres datos.

Los estudiantes de posgrado deben presentar una tesis y la nota es cualitativa: aprobó o no, los datos que se deben devolver en el string son: título de la tesis, director, evaluador 1, evaluador 2, nota cualitativa, fecha de sustentación.

Eventos: Cuando se recorre las evaluaciones, se debe generar un evento que informe a una clase permanencia de los estudiantes que no vayan bien en estas. Debe haber tres alertas: una para los estudiantes que matemáticamente tengan oportunidad; otra, para los estudiantes que matemáticamente no tengan oportunidad y otra para una clase directivos que los alerte de las dos situaciones anteriores. Otro evento, deberá informar a las dos clases anteriores la cancelación de una materia con los datos del estudiante que la canceló.

**Ejercicio 4**

Diseñe un diagrama UML e Implemente las siguientes clases en C# que se relacionan en una farmacia: Medicamentos, Laboratorios y movimientos.

Todos los medicamentos tienen los siguientes atributos: Nombre del medicamento, laboratorio que lo fabrica. Los medicamentos pueden ser en pasta, tienen la cantidad de pastillas y los mg; en capsulas, tienen si el tipo de relleno es gel o en polvo y medicamentos líquidos, tienen los mililitros y el material del envase: vidrio o plástico.

Para los laboratorios los atributos son: Nombre, teléfono de contacto.

Para el movimiento tiene los atributos son: Fecha, Medicamento, cantidad (si es una venta es negativo y si es un abastecimiento, es positivo). La farmacia tiene una lista de movimientos.

La farmacia tiene usuarios y clientes, ambos tienen los siguientes datos: nombre, cédula y teléfono. Los clientes adicionalmente, tienen la cantidad de puntos que van acumulando. Ambos tienen un método aplicar descuento que tiene el siguiente comportamiento: la tienda puede en un momento activar un descuento general a todos los productos que aplica tanto para el cliente como para el usuario. Pero al cliente siempre sobre el descuento general se le suma un 3% para cápsulas, 5% gel y 8% en polvo.

Eventos:

* Un evento disparará un mensaje cuando un producto llegue al punto de reorden. Es decir, si un producto tiene un punto de reorden en 10 unidades, cuando la existencia del producto llegue a este punto, deberá lanzar el mensaje del evento.
* Otro evento estará relacionado con mostrar a los clientes los puntos que va acumulando
* Un último evento, indicará el vencimiento de un producto un mes antes de que este llegue a su fecha de vencimiento

**Ejercicio 5**

Diseñe un diagrama UML e Implemente las siguientes clases en C# que se relacionan en una hacienda ganadera: Reses, Vacunas, Potreros.

Las reses son de tres tipos: ternero, cuando tienen una edad hasta los 12 meses; novillo, tienen edad hasta los 48 meses y cebón, cuando tienen una edad mayor a los 48 meses.

Las reses tienen los siguientes atributos: Edad, vacunas.

Todas las vacunas tienen los siguientes atributos: Nombre del medicamento, Lote, Fecha de aplicación.

Las vacunas son de dos tipos: vivas y bacterianas. Las vacunas bacterianas, tienen un atributo para período de aplicación que puede ser entre 2 y 4 semanas.

Las vacunas vivas, tienen un atributo para manejar el grado de atenuación que puede ser 10, 20 o 30.

La aplicación de las vacunas depende del tipo de res al que se le aplica. Para los novillos se aplica 2 vivas y dos bacterianas, para los terneros 3 bacterianas por una viva, y par a los cebones 4 vivas, 1 bacteriana. Se debe contener un atributo para verificar cuántas vacunas van de cada una.

El potrero tiene los siguientes atributos: Identificación, lista de Reses que están en el potrero y la edad para estar en el potrero. Quiere decir que un potrero solo tiene un tipo de res, no se mezclan. A cada potrero le caben 150 reses de ganado.

Cuando una res se vende, se registra una venta que está compuesta por la fecha, la res que se vende, el monto por el que se vendió y el potrero de dónde se saca.

Finalmente, la hacienda tiene una lista de las reses vendidas, y la información de los potreros.

Eventos:

* Un evento informará cuando a una res le sea completado su esquema de vacunación
* Un evento, alertará el vencimiento de una vacuna un mes antes de que este llegue a su fecha de vencimiento real
* Finalmente, un evento informará cuando un potrero llega al 50% de ocupación.

**Ejercicio 6**

Diseñe un diagrama UML e Implemente las siguientes clases en C# que se relacionan en un proveedor de vídeo en streaming: Contenido, Películas, Series, Cuentas y Usuarios. Este proveedor además posee un servicio de streaming de juegos. Los juegos tienen los siguientes atributos: nombre, género.

El contenido se compone de películas y series. Las películas tienen los atributos nombre, duración, calificación. Las series tienen los atributos nombre, temporada y capítulos por temporada.

El usuario tiene un id interno, el nombre y la fecha de afiliación.

Finalmente, las cuentas tienen los atributos: usuario y el contenido que ha visto.

El proveedor de streaming tiene la lista de todas las cuentas.

Los usuarios ganan puntos que se acumulan en sus cuentas, pero estos puntos los van acumulando de la siguiente manera: Para el contenido de películas, se van acumulando las horas y minutos vistos; para el contenido de series, se van acumulando los episodios o capítulos vistos y para los juegos se van acumulando los score que vaya sacando. Usted debe definir las políticas de puntos que acumula el usuario en la cuenta, esto quiere decir que usted dice cuántos minutos, o capítulos o score debe sacar para acumular 1 punto. Los usuarios tienen 3 categorías: Master, Pro, General. Usted define también las condiciones para pasar de una categoría a la otra.

Eventos:

* Un evento informará cuando lleguen nuevos títulos al sistema de streaming.
* Un evento, informará cuando se rompa el mayor puntaje de un juego.
* Finalmente, un evento informará el cambio de categoría de un usuario.

**Ejercicio 8**

Se necesita diseñar y desarrollar una solución orientada a objetos, para el manejo de un hotel. En el hotel hay tres tipos de habitaciones: Sencilla(30), Ejecutiva(10), Suite(5), con la siguiente configuración:

Sencilla: Tiene 1 cama doble o dos sencillas. En los pisos del 2 al 4, 10 por piso.

Ejecutiva: Tiene 1 cama queen o dos semidobles, además tiene minibar compuesto por 4 botellas de licor, dos botellas de agua, 1 kit de aseo personal, 2 gaseosas. En el piso 5.

Suite: Tiene 1 cama King, o una cama queen y una semidoble. Además, tiene un minibar compuesto por una botella de vino, 4 botellas de licor, 3 kit de aseo personal, 4 gaseosas. Tiene también un juego de 2 batas de baño. Están en el piso 6.

Todas las habitaciones tienen un costo: la habitación sencilla cuesta $200.000 la noche, la ejecutiva cuesta $350.000 la noche y la suite cuesta $500.000.

Las botellas de licor valen 25.000 cada una, la botella de vino vale 50.000, cada kit de aseo vale 9.000, el agua vale 3.500 y las gaseosas 3.000 cada una. El cliente o huésped pueden comprar las batas y cada una cuestan 70.000.

Cada habitación que tiene un minibar debe tener un método para llenarlo, que debe tener en cuenta cuántos productos quedaron en éste y complementar hasta que nuevamente queden full según lo que debe tener la habitación. Use para ello interfaces.

El hotel dispone un servicio de restaurante con una carta básica: El desayuno vale 15.000, el almuerzo 25.000 y la cena 20.000.  Si pide servicio a la habitación se recarga $5.000, independiente del número de órdenes. El servicio de lavandería cuesta $12.000 por prenda y la planchada $9.000. Ninguno de estos servicios tiene IVA.

La recepción del hotel se encarga de la facturación de cada habitación. El seguro hotelero para cada habitación es el 2.5% del precio por noche y el IVA el 19% que pagan solamente los huéspedes colombianos, los extranjeros están exentos de pago IVA. Se necesita implementar para la recepción el check-in check-out y la facturación al momento del checkout, con todos los consumibles.

Hay una oficina que maneja las reservas que se hacen, cuántas y cuáles habitaciones tienen reservas, cuáles están ocupadas y durante cuánto tiempo, cuándo desocupan una habitación para poder montar una nueva reserva.

El hotel tiene clientes y huéspedes. Tanto el cliente como el huésped tienen nombre, tipo id (CC, TI, PA, CE), número de documento y teléfono celular. Además, el cliente es aquel que es miembro del programa de fidelidad de la cadena de hoteles y tiene un código especial y a ellos se les aplica un descuento especial que cada semana cambia. Únicamente los clientes pueden llegar al hotel y solicitar una habitación sin hacer una reserva.

El programa debe tener una clase para cargar desde archivos planos la información de clientes y huéspedes, teniendo en cuenta que ambos objetos tienen información diferente.

Eventos:

* Un evento informará cada que se reporte un consumo del minibar
* Un evento, alertará la cancelación de una reserva y liberación de una habitación.