Ingeniería del Software (ISW)

Evaluación Teoría. Acto 2.

09-01-2017 ETSInf-UPV

Nombre:

Cuestiones *(3'5 puntos)* **Tiempo: 2 horas 30 min**

1. (*1 punto)* Indica si las siguientes afirmaciones son verdaderas o falsas y razona tu respuesta.

a) Un diagrama de secuencia permite representar la parte estática de un sistema, incluyendo las relaciones entre objetos.

b) Para implementar los constructores tenemos que observar la multiplicidad de las relaciones y si existen restricciones de navegación entre las clases.

c) Si hay Polimorfismo de inclusión o dirigido por la herencia y en el lenguaje C# tenemos una variable "`Persona p;`" y `p` referencia en tiempo de ejecución a un objeto de tipo `Estudiante` (siendo `Estudiante` subclase de `Persona`), al invocar el método `p.un_metodo()` que está definido tanto en `Persona` como en `Estudiante`, de la siguiente forma `public void un_metodo(){…}`, se ejecutará el método de la clase `Estudiante`.

d) La complejidad ciclomática indica el número máximo de pruebas de unidad (casos de prueba) que se pueden hacer a un método o función.

2. *(1 punto)* ¿Qué es Entity Framework? ¿Qué ventajas e inconvenientes implica su uso?

3. *(1 punto)* Define el término “Caso de Prueba”, cuál es su objetivo y cita qué técnicas para diseñar casos de prueba existen.

4. *(0’5 puntos)* ¿Qué estrategia de reutilización de código alternativa a la herencia se puede utilizar? Pon un ejemplo.

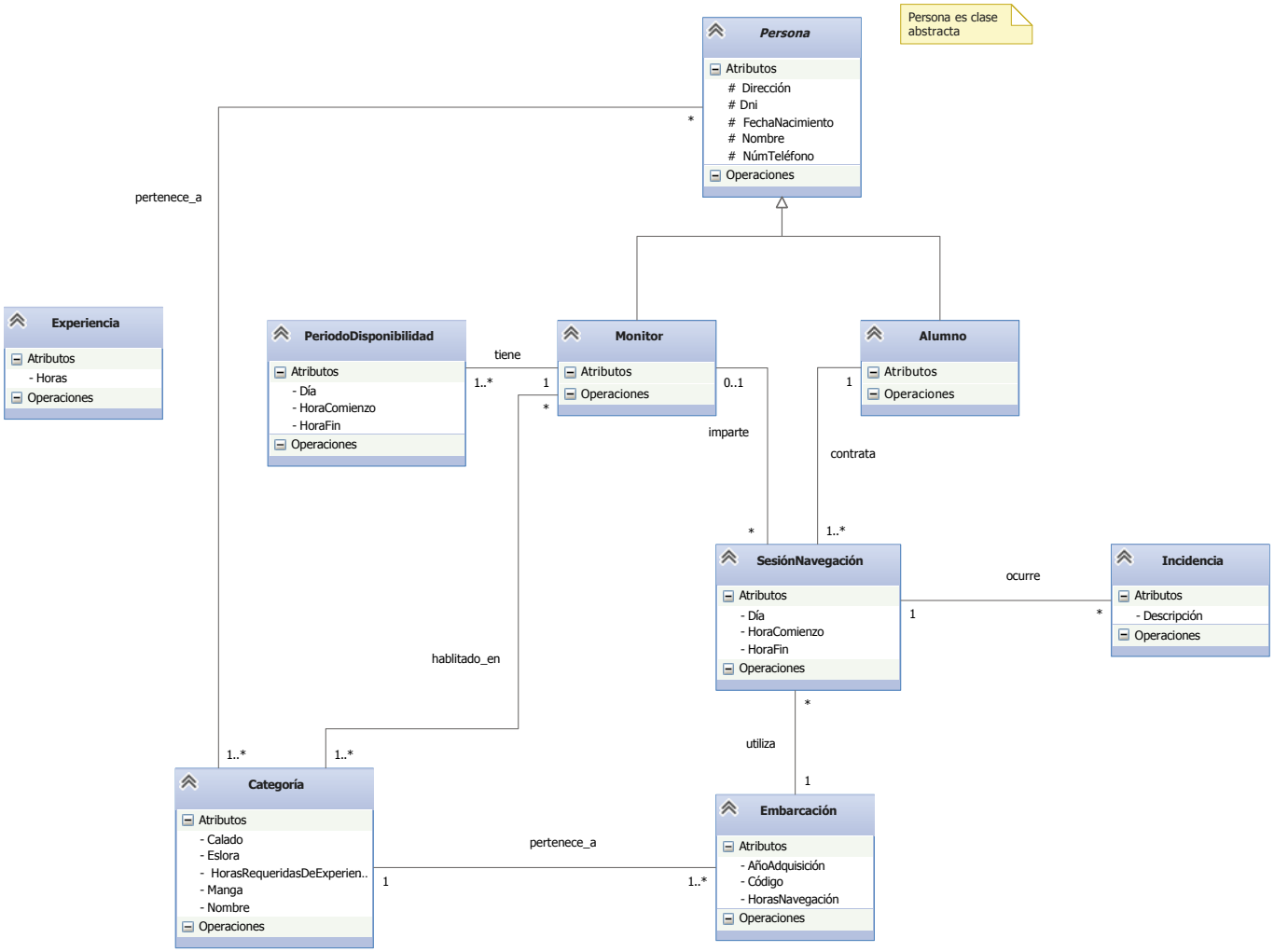
Ingeniería del Software (ISW)

Evaluación Teoría. Acto 2.

09-01-2017 ETSInf-UPV

Problemas *(6'5 puntos)*

Problema 1. *(4 puntos)* ISWSoft ha recibido el encargo de desarrollar una aplicación para una Escuela de Vela. Partiendo del diagrama de clases obtenido por el equipo de desarrollo, se pide:



a. *(2 puntos)* Obtener el diagrama de secuencia UML 2.0 para especificar el comportamiento del caso de uso "Contratar Sesión de Navegación" descrito como:

Para contratar una nueva sesión de navegación, el empleado introduce el dni del alumno que quiere contratar la sesión y la categoría de embarcación que quiere usar. El sistema comprueba si el alumno tiene suficiente experiencia (número mínimo de horas para navegar solo en esa categoría). Si el alumno no tiene la experiencia requerida, el sistema notifica dicha situación con un mensaje, y la sesión de navegación no es contratada. Si el alumno tiene la experiencia requerida, el empleado introducirá el rango de fechas que el alumno tiene disponible y la franja horaria en la que desea realizar su sesión de navegación. El sistema busca embarcaciones de esa categoría que estén disponibles en el rango de fechas y franja horaria introducida. Si encuentra embarcaciones disponibles el sistema las muestra junto a los días de disponibilidad. El empleado selecciona la embarcación y el día escogidos por el alumno, registrándose en ese momento la sesión de navegación. Si no hay embarcaciones disponibles el sistema muestra un mensaje de error.

b. *(1,0 punto)* Para todas las clases del modelo anterior, declara todos los constructores con sus argumentos de entrada y tipo de los mismos, que garanticen el comportamiento que se especifica en el diagrama de clases, e implementa únicamente los constructores de las clases `SesiónNavegación`, `Persona` y `Monitor`.

c. *(1,0 punto)* Utilizando los constructores declarados en el apartado anterior, y los métodos que se consideren necesarios, escribe el código C# para inicializar un sistema de forma correcta, garantizando que haya al menos un objeto de cada clase (*se pueden utilizar los valores que se deseen*).

Ingeniería del Software (ISW)

Evaluación Teoría. Acto 2.

09-01-2017 ETSInf-UPV

Problema 2. *(2'5 puntos)* Dado el siguiente método implementado en C#

```
public void busqueda(int num, int[] vector)
{
    int l = 0, h = 9;
    int m = 0;
    bool found = false;

    while (l <= h && found == false)
    {
        m = (l + h) / 2;
        if (vector[m] == num)
            found = true;
        if (vector[m] > num)
            h = m - 1;
        else
            l = m + 1;
    }
    if (found == false)
    {
        Console.Write("\nEl elemento {0} no esta en el vector", num);
    }
    else
    {
        Console.Write("\nEl elemento {0} esta en la posicion: {1}", num, m + 1);
    }
}
```

Se pide:

a) Diseñar los casos de prueba siguiendo la técnica del camino básico (dibuje el grafo de flujo correctamente etiquetado, calcule la complejidad ciclomática, especifique los caminos independientes y los casos de prueba asociados a cada camino).

b) Suponiendo que en la línea que marca la flecha se añade una sentencia "`return`" (que finaliza la ejecución del método), obtener el diagrama de flujo y calcular la complejidad ciclomática (no especificar caminos independientes ni casos de prueba).