# Integrantes: Matthew Kummerfield, Sebastian Arriola, David Del Aguila

# http://vectorblog.org/wp-content/uploads/2013/06/Telehealth-cartoon-e1370345782337.jpgHoja de trabajo No. 1: Almacenamiento íntegro

Objetivos de la hoja de trabajo

* Recordar mejores prácticas para escribir programas.
* Evaluar la manera de almacenar información.
* Introducir al estudiante al concepto de *encapsulación de información*.

Recomendación: trabajar en grupos de dos integrantes.

## Parte 1: Recordando conceptos y mejores prácticas de programación

1. En la disciplina de Desarrollo de software, ¿qué es modelar? Proporcione un ejemplo

- modelar es planear el programa utilizando diagramas de flujo, también algo como UML. Es planear la solución al problema, antes de programar. Un ejemplo podría ser el ejercicio del Hospital, en el cual de primero definimos las clases que íbamos a utilizar y sus propiedades y métodos.

1. ¿Qué es el *Unified Modelling Language* (UML) y para qué se utiliza? Mencione una ventaja de utilizar UML en sus documentos de diseño de Software por ejemplo en la tarea de diseñar el software Administrador de pacientes.

- Una forma de visualizar el diseño de un sistema. Ventajas podrían ser que agiliza el proceso de diseño de un programa y nos da orden.

1. ¿Qué es un diagrama UML de *Secuencia*? Dibuje un diagrama UML de Secuencia del ingreso de un registro de un paciente en su aplicación.

- es un diagrama para definir y mostrar la interacción entre objetos.

**

*Interacción con otros grupos. Haga una pausa y exponga y discuta con otro grupo su diagrama de secuencia y el diagrama de secuencia del otro grupo.*

1. ¿Qué buenas prácticas de programación conoce? ¿Cómo se deben nombrar las variables, parámetros, atributos y métodos?

- seguir un estándar para nombrarlas, como Camel Notation, para variables, métodos y clases.

1. Coloque 5 líneas de su programa (no necesariamente líneas juntas) y explique si son buen ejemplo o mal ejemplo sobre cómo nombrar variables, parámetros, atributos o métodos.

- public string razon { get; set; }

public ETipoDeSangre sangre { get; set; }

public string sangreS { get; set; }

public string nombre { get; set; }

public string apellido { get; set; }

pensamos que es buen ejemplo porque se sigue el mismo estándar para nombrar las variables, esto da orden al programa.

*Interacción con otros grupos. Haga una pausa y exponga y discuta con otro grupo sus ejemplos sobre el buen o el mal nombramiento en su programa.*

1. Lectura: Notación Húngara. Lea sobre la Notación Húngara. Resuma qué es la Notación Húngara.

- Consiste en prefijos indican el tipo de la variable.

1. Coloque uno de los métodos de su programa de Administración de pacientes y escríbalo en notación húngara. Coloque por lo menos 10 líneas de código.

- public void vGuardar()

{

File.Delete(CUtilities.SAVE\_FILE\_NAME);

TextWriter twTextWriter = File.AppendText(CUtilities.SAVE\_FILE\_NAME);

foreach(CPaciente cpPaciente in lstPacientes)

{

twTextWriter.WriteLine(cpPaciente.ToSaveString());

}

twTextWriter.Close();

}

1. ¿Qué ventajas y desventajas identifica sobre el uso de Notación Húngara. ¿Recomienda utilizarla? ¿Por qué sí o por qué no?

- desventaja: puede ser confuso si no sabemos bien que significan los prefijos. También extiende nombre de las variables y por lo tanto hace el programa mas lento y que use mas memoria.

Ventajas: hace que el programa sea más organizado. Ayuda a saber que tipo de dato es una variable si estamos programando en un ambiente sin IDE. También facilita el mantenimiento del programa.

*Interacción con otros grupos. Comente sobre la Notación Húngara aplicada a sus 10 líneas de código.*

## Parte 2: Almacenamiento íntegro

1. ¿Cuál es la utilidad de colocar atributos privados en una clase?

- No dejar que ciertas partes del programa modifiquen o utilicen datos que no deberían. También da seguridad.

1. ¿Cuál es la utilidad de colocar métodos privados en una clase?

- Igual que arriba, para que no se utilicen fuera de la clase y que no gasten memoria por gusto.

1. Lectura: Encapsulación de información.
2. Identifique un fragmento de código (de su aplicación de Administración de pacientes) ya sea donde se haya encapsulado el manejo de datos o donde no esté encapsulado el manejo de datos.

- public class CPaciente : CPersona

{

#region Public Variables

public string razon { get; set; }

public ETipoDeSangre sangre { get; set; }

public string sangreS { get; set; }

#endregion

#region Constructors

public CPaciente(string nombre, string apellido, string razon, ETipoDeSangre sangre)

{

this.nombre = nombre;

this.apellido = apellido;

this.razon = razon;

this.sangreS = CUtilities.StringValueOf(sangre);

}

#endregion

#region Public Methods

public string ToSaveString()

{

return nombre + "$" + apellido + "$" + razon + "$" + (int)sangre + "$";

}

#endregion

}

*Interacción con otros grupos. Comente sobre los fragmentos de código identificados.*

1. Comente: ¿Para qué encapsular?

- Para poder llevar un mejor orden cuando se trabaja en equipo. Para asegurar datos.