**Proyecto Curso: Programación Orientada a Objetos**

**Administración Hotelera**

**Integrantes:**

Ángeles Trujillo, Dennys

Amaya Suarez, Juan Manuel

Herrera Goñi, Danny

Hurtado Arguedas, Giovanni

Vera Villanueva, Carlos Alberto

**2012**

# Presentación

La Programación Orientada a Objetos es una forma de programación que toma practicas de paradigmas de programación previos como la programación estructurada y modular.

La buena POO permite la reutilización de código que finalmente se traduce en menor esfuerzo y cantidad de código por proyecto.

# Índice

Contenido

[Presentación 1](#_Toc326069868)

[Índice 2](#_Toc326069869)

[Introducción 3](#_Toc326069870)

[Caso Propuesto 3](#_Toc326069871)

[Historia de Usuario 3](#_Toc326069872)

[Pruebas Unitarias 4](#_Toc326069873)

# Introducción

En el siguiente caso práctico debemos ser capaces de aplicar los conceptos de POO para la definición y diseño de un proyecto software, basándonos en el lenguaje de Java.

Para el avance parcial lo que se solicita es desarrollar una historia de usuario con sus respectivos @Test.

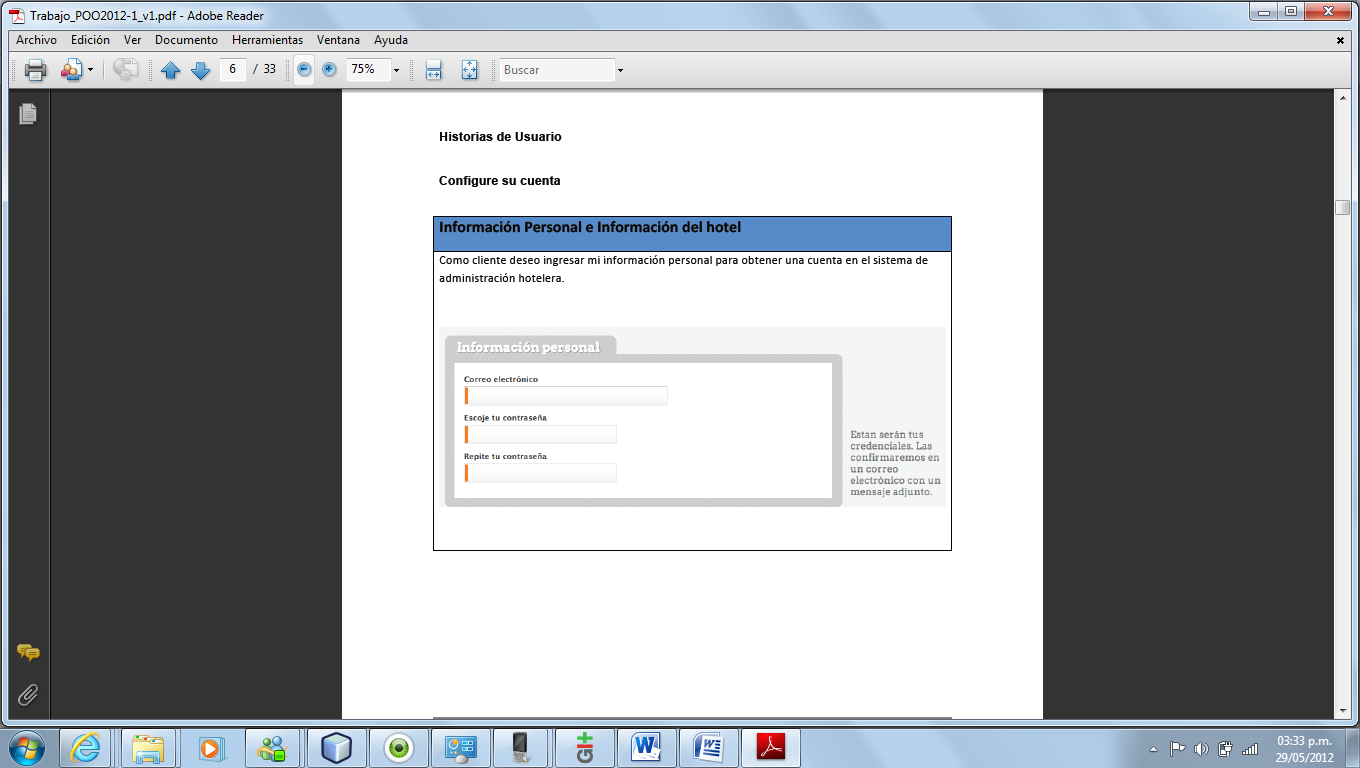
El avance debe estar en el repositorio GitHub, con su respectiva documentación.

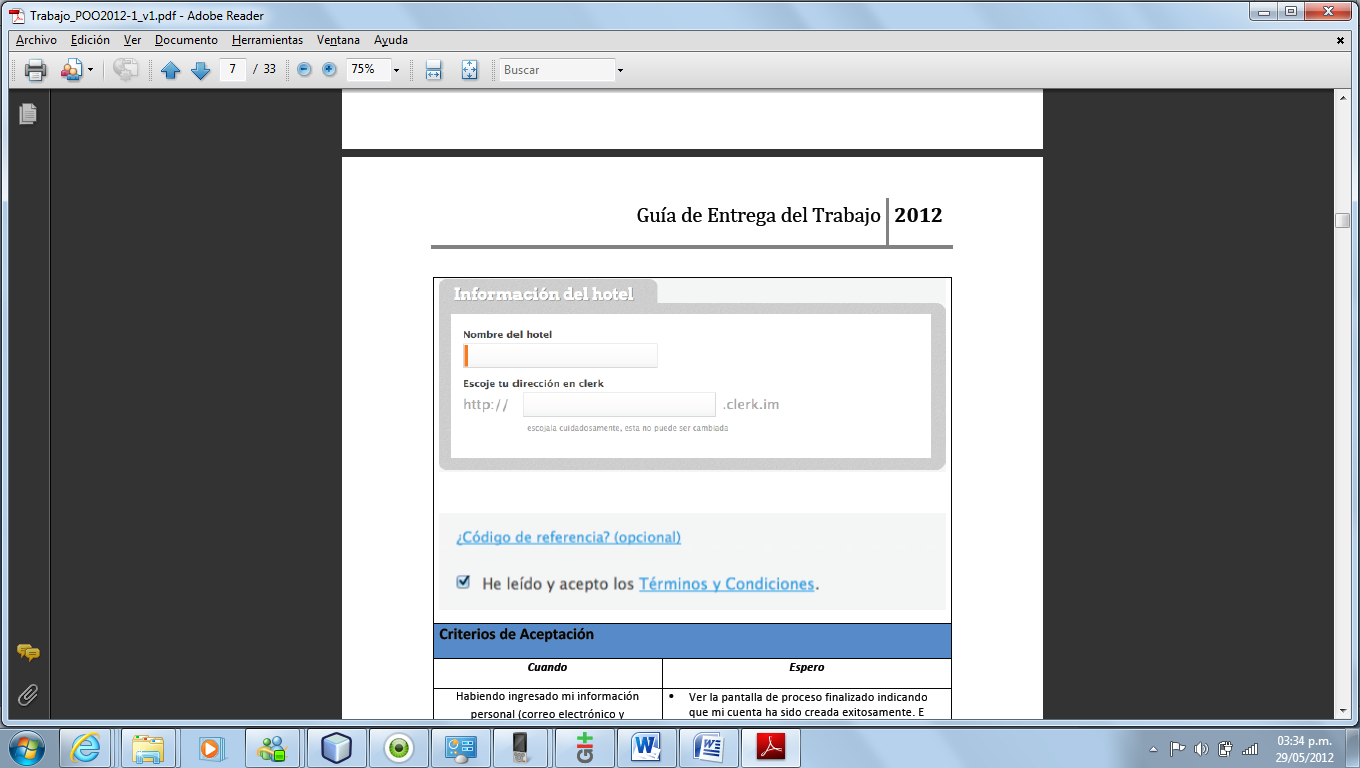
# Caso Propuesto

En el presente ciclo nos toca desarrollar el caso práctico de Administración Hotelera, el cual deberá ser desarrollado durante todo el ciclo académico, dividido en dos entregables.

# Historia de Usuario

La historia de usuario seleccionada es : **Configurar Cuenta en el sistema.**





|  |  |
| --- | --- |
| **Criterios de Aceptación** | |
| *Cuando* | *Espero* |
| * Habiendo ingresado i información personal (correo electrónico y contraseña) y la información del Hotel (nombre y dirección que tendré en el sistema hotelero), además de dar conformidad de haber leído y aceptado los Términos y Condiciones. | * Ver la pantalla de proceso finalizado indicando que mi cuenta ha sido creada exitosamente. E indicando que tengo 24 horas para confirmar el email recibido del sistema hotelero. Se indica además que se tiene el plan gratuito hasta que se cambie a un plan con costo. |
| * No ingreso los campos solicitado o aceptado los Términos y condiciones. | * Un mensaje de error por cada campo no ingresado |
| * Cuando ingreso el nombre del hotel | * Espero hasta 5 recomendaciones sobre el nombre. Una vez seleccionado o ingresado el nombre la dirección web se generara anteponiéndole http:// <<nombre>>.clerk.im |

# Pruebas Unitarias

### Administracion de Cuenta Test : AdmCuentaTest

@Test

public void clienteNoTieneCorreo(){

assertEquals(0, admcuenta.getCuentas().size());

}

@Test

public void clienteYaTieneUncorreoRegistrada(){

admcuenta.registrarCuenta("cvera@hotmail.com","car123","car123","Sheraton","Sheraton-hotel",true,101100);

assertEquals(1, admcuenta.getCuentas().size());

}

@Test

public void clienteYaTieneCorreosRegistradas(){

admcuenta.registrarCuenta("lsanchez@hotmail.com","lui123","lui123","Inkamar","Inkamar-hotel",true,101100);

admcuenta.registrarCuenta("mas2711@hotmail.com","jua123","jua23","Marriot","Marriot-hoterl",true,101100);

admcuenta.registrarCuenta("cvera@hotmail.com","car123","car123","Andes","Andes-hotel",true,101100);

assertEquals(3, admcuenta.getCuentas().size());

}

@Test

public void siNoIngresoCorreoElectronicoMeDebeDarError(){

assertFalse(admcuenta.verificarCorreo(null));

System.out.println("Debe ingresar correo electronico");

}

@Test

public void siIngresoCorreoDebeAceptarlo(){

assertTrue(admcuenta.verificarCorreo("cvera@hotmail.com"));

System.out.println("Correo Ingresado");

}

@Test

public void siNoIngresoPasswordMeDebeDarError(){

assertFalse(admcuenta.verificarPassword(null));

System.out.println("Debe ingresar Password");

}

@Test

public void siIngresoPasswordDebeAceptarlo(){

assertTrue(admcuenta.verificarPassword("lui123"));

System.out.println("Password Ingresado");

}

@Test

public void siNoIngresoRepetirPasswordMeDebeDarError(){

assertFalse(admcuenta.verificarRepetirPassword(null));

System.out.println("Debe Ingresar repetir Password");

}

@Test

public void siIngresoRepetirPasswordDebeAceptarlo(){

assertTrue(admcuenta.verificarRepetirPassword("lui123"));

System.out.println("Repetir Password Ingresado");

}

@Test

public void siNoIngresoNombreDeHotelMeDebeDarError(){

assertFalse(admcuenta.verificarNombreHotel(null));

System.out.println("Debe Ingresar nombre de hotel");

}

@Test

public void siIngresoNombreDeHotelDebeAceptarlo(){

assertTrue(admcuenta.verificarNombreHotel("Andes"));

System.out.println("Nombre de Hotel Ingresado");

}

@Test

public void siNoIngresoConformidadMeDebeDarError(){

assertFalse(admcuenta.verificarConformidad(false));

System.out.println("Debe Activar Conformidad");

}

@Test

public void siIngresoConformidadDebeAceptarlo(){

assertTrue(admcuenta.verificarConformidad(true));

System.out.println("Conformidad Activado");

}

@Test

public void registrarCuentas(){

assertEquals(0, admcuenta.getCuentas().size());

admcuenta.registrarCuenta("cvera@hotmail.com","car123","car123","Sheraton","Sheraton-hotel",true,023100);

assertEquals(1, admcuenta.getCuentas().size());

admcuenta.registrarCuenta("lsanchez@hotmail.com","lui123","lui123","Inkamar","Inkamar-hotel",true,023100);

admcuenta.registrarCuenta("jrodriguez@hotmail.com","jua123","jua23","Marriot","Marrot-hotel",true,023100);

admcuenta.registrarCuenta("pmendoza@hotmail.com","pau123","pau123","Andes","Andes-hotel",true,023100);

assertEquals(4, admcuenta.getCuentas().size());

try{

admcuenta.registrarCuenta("jrodriguez@hotmail.com","jua123","jua23","Marriot","Marriot-hotel",true,023100);

fail();

}catch(Exception ex){

assertEquals("CORREO ELECTRONICO YA REGISTRADO", ex.getMessage());

}

}

}

### Administrador Cuenta de Usuario : Adm Usuario Test

@Test

public void clienteSinCuenta(){

assertEquals(0, cliente.getCuentas().size());

}