



UNIVERSIDAD PRIVADA ANTEÑOR ORREGO

FACULTAD DE INGENIERIA

ESCUELA DE INGENIERIA DE COMPUTACION Y SISTEMAS

SILABO DE PROGRAMACIÓN ORIENTADA A OBJETOS

I. DATOS INFORMATIVOS

1.1 Asignatura	:	Programación Orientada a Objetos
1.2 Código	:	ICSI-232
1.3 Tipo	:	Fundamental Especialidad Aplicado
1.4 Ciclo Académico	:	II
1.4 Semestre Académico	:	2010 - 20
1.5 Créditos	:	04
1.6 Extensión horaria	:	08 horas semanales. 128 horas totales.
Teoría	:	02 horas.
Laboratorio	:	04 horas.
Taller	:	02 horas.
1.7 Duración	:	17 Semanas.
		Fecha Inicio : 16/Agosto/2010
		Fecha Término: 11/Diciembre/2010
1.8 Pre-requisito	:	Ninguno.
1.9 Semestre	:	2009-10.
1.10 Docentes Responsables	:	Wilder Adán Namay Zevallos: wnamayz@upao.edu.pe Oscar Tincopa Urbina: otincopau@upao.edu.pe Henry Mendoza Puerta: hmendozap@upao.edu.pe Freddy Infantes Quiroz: finfantessq@upao.edu.pe Edwin Valencia Castillo: evalenciac@upao.edu.pe Wilfredo Valverde Quispe: wvalverdeq@upao.edu.pe

II. FUNDAMENTACION

El aprendizaje de las bases teóricas – prácticas del paradigma de programación orientada a objetos, constituye una necesidad para el desarrollo de las habilidades básicas y esenciales en la formación del ingeniero de computación y sistemas.

Este curso, por su naturaleza se orienta a desarrollar las habilidades de programación bajo el enfoque de objetos, el cual, permite que el estudiante se sumerja en el mundo de la programación a través del entendimiento de objetos, reales o imaginarios del mundo real e ideal.

La importancia de la materia se relaciona con el desarrollo de competencias y habilidades para la comunicación, modelación y programación aplicando el paradigma orientado a objetos mediante un trabajo individual en primera instancia y en equipo.

III. SUMILLA

La unidad de ejecución curricular desarrolla la asignatura Fundamentos de Programación Orientada a Objetos (POO), en el cual, el estudiante aprende el paradigma de orientación a objetos, como técnica de programación, mediante la internalización de los principios y técnicas esenciales de modelación y programación de sistemas de baja y media complejidad.

Los temas del paradigma de orientación a objetos que se desarrollaran en el presente curso están relacionados con la abstracción, encapsulamiento, el manejo de clases y objetos, relaciones y jerarquía entre clases, polimorfismo y patrones de diseño.

La asignatura tiene tres componentes: teórico, laboratorio y asesoría los que facilitan al estudiante desarrollar destrezas técnicas promoviendo durante el curso el desarrollo de actitudes, valores y principios éticos del futuro profesional en formación.

IV. COMPETENCIA GENERAL:

Implementa programas computacionales utilizando el paradigma de programación orientada a objeto y haciendo uso de las técnicas esenciales de construcción de objetos, jerarquía de clases para resolver problemas típicos.

ACTITUDES:

- Analiza situaciones problemáticas aplicando los principios del paradigma de orientación a objetos.
- Muestra interés en desarrollar destrezas técnicas relacionadas con la programación orientada a objetos.
- Reconoce el aporte del paradigma de orientación a objetos en la programación de sistemas y desarrollo de productos de software.
- Asume roles y responsabilidades en la solución de problemas y ejercicios asignados al equipo del cual forma parte.

V. PROGRAMACION POR UNIDADES DE APRENDIZAJE

UNIDAD I : **PARADIGMA DE LA PROGRAMACION ORIENTADA A OBJETOS.**
OBJETIVO : Utilizar los principios de la programación orientada a objetos
COMPETENCIA : Utiliza un lenguaje OO para implementar los conceptos y principios de la POO.
POO.

CONCEPTUALES	PROCEDIMENTAL	ACTITUDINALES
TEORIA: - Presentación del curso. - Introducción a la POO: Abstracciones, Clases, Objetos, Métodos - Estructuras de Control LABORATORIO: - Estructura de un Programa Java. - Familiarización con JCreator. - Algoritmos Secuenciales.	Internaliza los conceptos básicos de la POO. Se familiariza con el IDE JCreator	Demuestra interés en la clase, es puntual y responsable. Demuestra responsabilidad en el desarrollo de los ejercicios.

<p>TEORIA:</p> <ul style="list-style-type: none"> - Parámetros, Argumentos. - Métodos de Instancia. - Métodos de Clase. - Devolución de Objetos. - Objetos y Mensajes <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Estructuras Condicionales - Estructuras Repetitiva - Implementación de Clases, Instanciación de Objetos, 	<p>Comprende las nociones de parámetros y argumentos en los métodos invocadores e invocados.</p> <p>Comprenden las Características de métodos que reciben y devuelven objetos.</p>	<p>Participa activamente en su grupo de trabajo para resolver las cuestiones planteadas en clase.</p> <p>Demuestra interés en clase, es puntual y responsable.</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Encapsulamiento de variables. - Declaración e inicialización de las propiedades de un objeto. - Clases envolventes. - Clase String <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Uso de métodos de instancia, métodos de clase. - Manejo de parámetros y argumentos. 	<p>Detalla las formas de declaración e inicialización de objetos, clase String y clases envolventes.</p>	<p>Valora la importancia de la declaración e inicialización de los objetos.</p> <p>Reconoce correctamente las declaraciones de las clases envolventes.</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Relaciones entre clases - Herencia e Interfaces - Polimorfismo <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Ejemplos de Clases Envolventes. - Ejemplos de la clase <i>String</i>. - Ejemplos de Métodos y Métodos de Instancia. 	<p>Explican los tipos de relaciones entre clases.</p> <p>Explica el mecanismo de la herencia.</p> <p>Explica las formas de invocar a los métodos de la superclase</p> <p>Explica la sobrecarga y sobre escritura y las formas de limitar su acceso</p>	<p>Asume una posición reflexiva.</p> <p>Demuestra interés y curiosidad por las formas en cómo se establece la relaciones entre clases.</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Tipos de Enlaces: Enlace Temprano y Enlace Dinámico. - Organización de Clases. <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Herencia entre Clases - Agregación y Composición - Polimorfismo 	<p>Explica el concepto de asignación de valor en los diferentes estadios de un programa.</p> <p>Explica las formas y técnicas de agrupación de clases y jerarquía de clases..</p>	<p>Demuestra interés en comprender los mecanismos de enlace y su relación con la herencia.</p> <p>Acepta la importancia de organizar las clases y sus jerarquías.</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Clases Internas / Anónimas. - Arquitectura MVC. <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Enlaces temprano y tardío. - Casting (<i>upcasting</i> / <i>downcasting</i>). 	<p>Explica los conceptos de clases internas y anónimas.</p> <p>Explica los conceptos de la arquitectura Modelo-Vista-Controlador.</p>	<p>Demuestra interés por comprender la naturaleza de los tipos de clase.</p> <p>Participa activamente en su grupo para responder las cuestiones planteadas en clase.</p>

<p>TEORIA:</p> <ul style="list-style-type: none"> - Modelo de eventos. - Modelo Callback. <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Clases Internas y Anónimas - Modelo MVC 	<p>Explica los conceptos de la gestión de eventos.</p> <p>Explica el modelo <i>callback</i> y su relación con las clases internas.</p>	<p>Valora la importancia del modelo de eventos</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Manejo de Excepciones <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Modelo de eventos. - Modelo callback. - Modelo de delegación. 	<p>Explica los mecanismos para manejo y gestión de errores de un programa.</p>	<p>Asume una posición reflexiva y participativa en su grupo de trabajo para determinar errores potenciales en un programa.</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Colecciones Genéricas. - Clases Persistentes. <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Manejo de excepciones. - Uso de colecciones 	<p>Explica las colecciones genéricas como forma de almacenamiento temporal de información en la memoria.</p> <p>Explica el mecanismo de la persistencia.</p>	<p>Asume una posición reflexiva y acepta la importancia de usar colecciones para almacenar información estructurada en la memoria.</p> <p>Demuestra interés en aprender a reconocer las clases persistentes</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Flujos de Entrada y Salida. - Serialización de Objetos. <p>LABORATORIO</p> <ul style="list-style-type: none"> - Implementación de clases persistentes. - Manejo de excepciones. 	<p>Explica la naturaleza de almacenamiento de la información persistente</p>	<p>Participa activamente para determinar las clases persistente de un conjunto de clases.</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Interfaces Graficas de Usuario (Widgets, Gui's). - Gestores de Diseño. <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Manejo de Flujos - Manejo de archivos - Serialización de objetos 	<p>Explica que son y cómo se clasifican los widget.</p> <p>Explica los componentes de un kit de herramientas abstractas.</p> <p>Determina los gestores de diseño mas apropiados para una interfaz gráfica.</p>	<p>Valora la construcción de programas que utilizan interfaces gráficas de usuario.</p> <p>Demuestra interés por comprender el uso adecuado de las interfaces gráficas de usuario.</p>
<p>TEORIA:</p> <ul style="list-style-type: none"> - Gestores de Diseño <p>LABORATORIO:</p> <ul style="list-style-type: none"> - Implementación de <i>Frames</i> - Manejo de Gestores de diseño y <i>widgets</i> 	<p>Explica que son y para que sirven los gestores de diseño.</p>	<p>Demuestra interés en la programación de interfaces graficas organizadas mediante gestores de diseño.</p>

PROGRAMACION DE UNIDADES TEMATICAS

UNIDAD II : PROGRAMACIÓN ORIENTADA A OBJETOS AVANZADA.

OBJETIVO : Utilizar e Internalizar técnicas avanzadas de la programación Orientada a Objetos (POO).

COMPETENCIA : Reconoce y aplica técnicas avanzadas de POO en la solución de problemas.

CONCEPTUALES	PROCEDIMIENTOS	ACTITUDINALES
TEORIA: - Programación con multihilos LABORATORIO: - Ejemplos de Clases	Elabora algoritmos utilizando los conceptos de programación con hilos. Altera o modifica programas para soportar programación	Desarrolla ejercicios grupales mostrando interés, responsabilidad. Manifiesta interés por la programación paralela.
TEORIA: - Patrones de Diseño: Patrones de <i>Creación</i> LABORATORIO: - Ejemplos de la clase Thread / Runnable - Implementacion de algoritmos concurrentes.	Explica la naturaleza de los patrones de diseño de creación.	Participa activamente para encontrar problemas típicos cuya solución requiere de un patrón de diseño.
TEORIA: - Patrones de diseño: Patrones de <i>Comportamiento</i> LABORATORIO: Implementación de Singletons y Prototipos.	Explica la naturaleza de los patrones de diseño de comportamiento	Participa activamente para encontrar problemas típicos cuya solución requiere de un patrón de diseño. Valora la importancia de los patrones de diseño de comportamiento.
TEORIA: - Proceso de Desarrollo de Software LABORATORIO: - Implementación de Comandos, Mementos y Observadores.	Explica las fases de desarrollo de un proyecto de software. Reconoce los roles de los miembros de un equipo de desarrollo de software Aprende a utilizar las tarjetas CRC para el diseño basado en responsabilidades.	Asume con responsabilidad el rol designado en el equipo. Comprende e investiga los diferentes procesos de desarrollo de software.
TEORIA: - Revisión final del Proyecto LABORATORIO: - Evaluación final de laboratorio.	Explica las fases de desarrollo de un proyecto de software.	Sustenta en forma oral el desarrollo de su proyecto manifestando responsabilidad e interés.

VI. ESTRATEGIAS METODOLOGICAS:

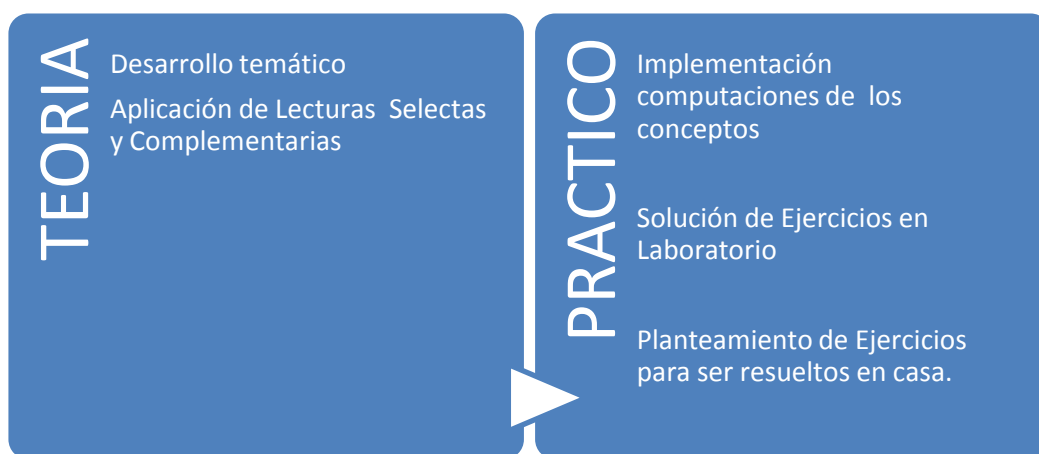
Esta asignatura se desarrollará en base a dos aspectos:

TEÓRICO:

- a) Exposición interactiva en clases de los contenidos temáticos de cada unidad y la importancia de estos en la resolución de problemas computacionales.
- b) Lecturas complementarias para ampliar los temas dados en las clases teóricas.

PRÁCTICO:

- a) Resolución de ejercicios tipos por el profesor de laboratorio.
- b) Resolución de ejercicios parcialmente resueltos.
- c) Resolución de ejercicios propuestos.
- d) Proyecto final.



VII. MATERIALES EDUCATIVOS Y OTROS RECURSOS DIDACTICOS:

Los materiales que se emplearán serán los siguientes:

a. Materiales educativos interactivos:

Texto guía, material de clase, hojas de prácticas de salón y laboratorio, direcciones electrónicas para recabar información sobre contenidos planteados.

b. Materiales educativos para la exposición:

Pizarra, plumones, mota, proyector multimedia.

VIII. INDICADORES, TÉCNICAS E INSTRUMENTOS DE EVALUACIÓN:

- ❖ La evaluación del curso comprende dos exámenes: Parcial y Final (EP y EF, respectivamente).
- ❖ Las actividades prácticas de laboratorio serán desarrolladas en forma conjunta por el profesor y el estudiante. Se llevarán a cabo dos prácticas calificadas de laboratorio. La solución de los ejercicios propuestos en todos los laboratorios acumula puntaje para el Promedio de Laboratorio (PL).
- ❖ Se desarrollará además un proyecto (PF) a lo largo de todo el semestre, el cual será sujeto a evaluación (avances y presentación final). Además durante el taller se realizarán ejercicios, los que acumulan puntaje para el Promedio de Taller (PT).
- ❖ La nota promocional se define del siguiente modo:
⇒ Promocional = 26%EP + 26%EF + 15% PL + 16%PP + 17%PT

- ❖ En la nota promocional se aplica redondeo, donde se aproxima al entero superior a las notas con parte fraccionaria mayor o igual a 0.5.
- ❖ La escala de calificación es de 0 a 20.
- ❖ El alumno aprueba el curso si Promocional ≥ 11 .
- ❖ Habrá un examen de aplazados para los alumnos que estén desaprobados, pero con promocional mayor o igual a 07.
- ❖ El alumno que falte a alguna evaluación, laboratorio o taller, tendrá nota 0.
- ❖ El sistema de evaluación no permite la existencia de exámenes de rezagados por ningún motivo, salvo justificación solicitada por escrito ante las autoridades respectivas.
- ❖ La asistencia a clases de teoría, taller y laboratorio es obligatoria, más del 30% de inasistencia en cada parte del curso inhabilita al alumno a tener promedio promocional.

IX. PROGRAMA DE TUTORÍA:

Consiste en orientar y ayudar a los alumnos a resolver problemas académicos y personales con el aporte del docente debido a su experiencia; por lo tanto se tendrá en cuenta lo siguiente:

- a) Todas las dificultades que se suscriben en el desarrollo de la asignatura en forma individual o grupal serán consultadas con el docente.
- b) El docente tendrá la responsabilidad de dar una orientación permanente y continua a los estudiantes para motivarlos y estimularlos a través de herramientas pedagógicas a fin de lograr los aprendizajes deseados.

X. BIBLIOGRAFIA

BIBLIOGRAFÍA BÁSICA:

- ❖ Lemay L, Cadenhead R. *Aprendiendo Java en 21 Días*. USA: Pearson.
- ❖ Deitel, Harvey, M.; Deitel, Paul J. *Cómo programar en JAVA*. Editorial: Pearson Educación. 2004.
- ❖ Bruce Eckel - *Thinking in Java*. 2006
- ❖ Britton C, Doake J. *A student guide to Object Oriented Development*. UK: Elsevier Ltd. 2005
- ❖ Garrido J. *Object Oriented Programming: From Problem to Solving to Java*. USA: Charles River Media; 2003.

BIBLIOGRAFÍA COMPLEMENTARIA:

- ❖ Morelli R, Walde R. *Java, Java, Java: Object Oriented Problem Solving*. New York: Prentice Hall; 2005
- ❖ Bruce E. *Thinking in Java – 2006*
- ❖ Barry H. *Object-Oriented Programming With Java Computers*. USA 2000
- ❖ Goran Svenk. *Object-Oriented Programming - Computers – 2003*.
- ❖ Arnow D, Weiss G. *Introduction to programming using Java. An object oriented approach*. New York: Addison Wesley; 2000.
- ❖ Thomas C. *An introduction to object-oriented programming with Java*. 2da ed. New York: McGraw Hill; 1999.

