Programa de la Asignatura:

Lenguajes de Programación III

Código: 758

Carrera: ***Ingeniería en Computación*** Plan: ***2008*** Carácter: ***Obligatoria***
Unidad Académica: ***Secretaría Académica*** Curso: ***Tercer Año – Primer cuatrimestre***
Departamento: ***Ingeniería*** Carga horaria total: ***60 hs.*** Carga horaria semanal: ***4 hs.***
Formación Experimental: ***00 %*** Formación teórica: ***00 %*** Formación práctica: ***00 %***

Materias Correlativas Obligatorias

- ***Lenguajes de Programación II***
- ----------
- ----------

Cuerpo Docente

Fontela, Carlos
Sanchez, Diego

Índice

AÑO ACADÉMICO 2012 ÚLTIMA REVISIÓN 00/00/2012

Firma Docente Firma Coordinador

1. **FUNDAMENTACION**

2. **ENCUADRE Y ARTICULACIÓN DE LA ASIGNATURA**

3. **OBJETIVOS**

4. **CONTENIDOS MÍNIMOS**

Java. Mejoras e innovaciones sobre C++. Programación basada en objetos. Programación orientada a objetos. Creación de aplicaciones Java. Estructuras de selección y de control. Introducción a las clases, objetos y métodos. Arreglos, cadenas y caracteres. Lista, cola, pila y árbol en Java. Archivos y flujos. Introducción a las applets de Java. Uso de gráficos. Componentes de una interfaz gráfica con el usuario. Multihilos. Uso de multimedia: Imágenes, animación y audio.

5. **PROGRAMA ANALÍTICO**

Unidad 1:

OBJETOS

Objetos y mensajes. Objetos y clases. Estado, comportamiento, identidad. Paquetes. Historia hacia la POO. POO vs. Procedural. Atributos de clase. Métodos de clase. Uso en Java, C# y Smalltalk.

Unidad 2:

CLASES

Implementación de clases: atributos, métodos y propiedades, constructores, excepciones. Diseño contractual. TDD o diseño guiado por las pruebas. Objetivos de la orientación a objetos. Abstracción y ocultamiento de información: encapsulamiento. Implementación en Java, C# y Smalltalk.

Unidad 3:

REUTILIZACIÓN

Delegación. Herencia. UML: clases, paquetes, secuencias. Cuándo usar herencia y cuándo delegación. Redefinición. Clases abstractas. Herencia múltiple como concepto. Constructores, destructores, herencia y composición. Atributos y métodos protegidos. Implementación en Java, C# y Smalltalk.

Unidad 4:

Métodos virtuales. Métodos abstractos. Polimorfismo como concepto. Polimorfismo y vinculación tardía. Métodos abstractos. Interfaces. Clases internas. Implementación en Java, C# y Smalltalk.

Unidad 5:

PRUEBAS DE SOFTWARE EN EL PARADIGMA DE OBJETOS

Objetivos de las pruebas. Técnicas de prueba: caja blanca y caja negra. Tipos de pruebas: revisiones de código, unitarias, de integración, de sistema, de aceptación. Pruebas en POO y en entornos interactivos: su diferencias frente a las pruebas tradicionales. Axiomas sobre datos adecuados de pruebas. Automatización de pruebas.

Unidad 6:

EXCEPCIONES

Dependencia de estados: enfoques conservadores y optimistas. Errores y excepciones. Los modelos primitivos de excepciones. Los modelos de excepciones de los lenguajes orientados a objetos: lanzamiento y captura. Excepciones en Java, C# y Smalltalk. Ventajas y desventajas cada modelo de excepciones.

Unidad 7:

PROGRAMACIÓN ORIENTADA A OBJETOS: OTROS CONCEPTOS

Colecciones e iteradores. Genericidad. Programación guiada por eventos. Estados, transiciones y diagramas de estados. Persistencia: objetivos, principios, limitaciones de los lenguajes de POO; tipos de persistencia en función de la forma de manejar objetos complejos; serialización y XML. Concurrencia: escenarios concurrentes, objetos activos, exclusión mutua y sincronización, objetos inmutables, llaves de lectura y escritura, llaves de exclusión mutua (mutex), semáforos, actualizaciones optimistas; bloqueos transitorios y bloqueos patológicos; dependencia de estados, excepciones y constructores. Implementación en Java, C# y Smalltalk.

Unidad 8:

DISEÑO DE INTERFACES GRÁFICAS DE USUARIO

Interacción humano-computadora (HCI). Por qué centrarse en el usuario. Importancia de la interfaz de usuario. Aceptación práctica de un sistema. El caso particular de los sitios web. Concepto de usabilidad. Consecuencias de la falta de usabilidad. Estilos de interfaces. Metodologías de diseño de interfaces. Guías de diseño. Uso de texto y caracteres. Uso de colores e imágenes. Errores habituales de texto y de interacción. Metáforas y símiles. Colores. Legibilidad. Elementos de control. La atención del usuario. Estándares externos e internos. Definición de objetivos de usabilidad. Evaluación heurística de la usabilidad..

6. **BIBLIOGRAFÍA BÁSICA**

- Carlos Fontela, “Orientación a objetos con Java y UML”.
- Kent Beck, “Test Driven Development: By Example”.
- Steve McConnell, “Code Complete”.
- Kent Beck, “Implementation Patterns”.
- Bertrand Meyer, “Desarrollo de software orientado a objetos”.
- Eric Evans, “Domain Driven Design”.
- Grady Booch, “Análisis y diseño orientado a objetos con aplicaciones”.
- James Martin y James Odell, “Análisis y diseño orientado a objetos”.
- Robert Martin, “UML para programadores Java”.
- Grady Booch - James Rumbaugh - Ivar Jacobson, “The Unified Modeling Language User Guide”.
- Bruce Eckel, “Thinking in Java, 3rd. edition”.
- Bjarne Stroustrup, "El lenguaje de programación C++", edición especial.
- Doug Lea, “Programación concurrente en Java”.

7. **BIBLIOGRAFÍA DE CONSULTA**

- Jakob Nielsen, “Usability Engineering”.
- Donald Norman, “The Design of Everyday Things”.

- Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, “Design Patterns”.
- Elizabeth Freeman, “Head-First Design Patterns”.
- Martin Fowler, “Refactoring: Improving the Design of Existing Code”.
- Ivar Jacobson – Grady Booch – James Rumbaugh, “El Proceso Unificado de Desarrollo de Software”.
- Kent Beck, “Extreme Programming Explained”.

8. **METODOLOGÍA DEL APRENDIZAJE**

Las clases se imparten en dos formas: teóricas con desarrollo de los conceptos a cargo del docente, y prácticas en computadoras de los temas vistos. Los alumnos deberán realizar trabajos prácticos en las formas y con las características que determinen los responsables de la asignatura.

9. **METODOLOGÍA DE EVALUACIÓN**

En esta asignatura las evaluaciones consisten en dos exámenes parciales y un final obligatorio. Cada parcial cuenta con una fecha de recuperación. Es condición necesaria para rendir cada uno de los parciales tener aprobados previamente los dos Trabajos Prácticos que se indicarán. Todos los exámenes se aprueban con nota de 4 (cuatro) o superior. La aprobación de la cursada es habilitada cuando el alumno cumple con los dos parciales aprobados en cualquier instancia y los trabajos prácticos oportunamente indicados.

10. **PLANIFICACIÓN**

CALENDARIO DE CLASES Y EVALUACIONES	
Semana 1	Unidad 1
Semana 2	Unidad 2
Semana 3	Unidad 3. Explicación Trabajo Práctico 1
Semana 4	Unidad 4
Semana 5	Unidad 4
Semana 6	Repaso. Ejercitación. Entrega Trabajo Práctico 1.
Semana 7	PRIMERA EVALUACION PARCIAL
Semana 8	Consultas. Ejercitación.
Semana 9	RECUPERATORIO PRIMERA EVALUACION PARCIAL
Semana 10	Unidad 5 Explicación Trabajo Práctico 2
Semana 11	Unidad 6
Semana 12	Unidad 7. Unidad 8
Semana 13	Repaso. Ejercitación. Entrega Trabajo Práctico 2
Semana 14	SEGUNDA EVALUACION PARCIAL
Semana 15	Consultas.
Semana 16	RECUPERATORIO SEGUNDA EVALUACION PARCIAL
Del al de	FINAL