FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS UNIVERSIDAD DE CHILE

PROGRAMA DE CURSO

Código	Nombre			
CC3002	**Metodologías de Diseño y Programación**			
Nombre en Inglés				
Design and Programming Methodologies				
SCT	Unidades Docentes	Horas de Cátedra	Horas Docencia Auxiliar	Horas de Trabajo Personal
6	10	3	1.5	5.5
Requisitos			Carácter del Curso	
CC3001(S) Algoritmos y Estructuras de Datos.			Obligatorio Licenciatura en Ciencias mención Computación.	
Resultados de Aprendizaje				
Al término del curso se espera que el estudiante desarrolle software orientado a objetos fácil de entender, extender y mantener en el tiempo. El alumno, diseña y programa buenos objetos, utiliza la herencia sólo cuando ésta provee ventajas reales, integra objetos para resolver un problema complejo, diseña y resuelve problemas usando patrones de diseño, evalúa diseños usando métricas y enfrenta desarrollo de software de pequeña y mediana complejidad usando metodologías estándares.				

Metodología Docente	Evaluación General
Clases de cátedra, clases auxiliares, trabajo individual y en grupo.	3 controles Examen Tareas

Unidades Temáticas

Número	Nombre de la Unidad	Duración en Semanas
1	Introducción	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Revisión del programa del curso • Introducción a la complejidad en el desarrollo de software. • Metodologías para enfrentar la complejidad en el desarrollo de software.	Al término de la unidad, el alumno identifica los problemas existentes en el desarrollo de software en general. Reconoce algunas metodologías usadas para el desarrollo de software.	[1]

Número	Nombre de la Unidad	Duración en Semanas
2	Clases y objetos	2

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Clases y objetos. • Objetos: estado, comportamiento, identidad. • Cómo definir buenos objetos? • Diseño de buenos tipos de datos abstractos. • Contratos • De tipos de datos abstractos a clases. • Ejemplos en java y c++ • Diseño de una buena clase	Al término de la unidad, el alumno – programa usando clases y objetos. - domina los conceptos básicos de orientación a objetos. - diseña e implementa buenos objetos. - entiende código escrito java y en c++ - programa correctamente en java o en c++	[1], [3],[8],[9],[11],[12]

Número	Nombre de la Unidad	Duración en Semanas
3	Herencia y subtipos	3,5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Motivando el uso de herencia • Polimorfismo y enlace dinámico. • Tipos y subtipos • Relaciones de subtipos entre: conjuntos, tipos estructurados y funciones. • Contraste entre herencia y subtipos. • Tipos de herencia • Contratos y herencia	Al término de la unidad, el alumno - programa usando herencia. - domina los conceptos tipos y subtipos y su relación con la herencia. - definir contratos usando herencia. - diseña e implementa buenos objetos usando herencia.	[1], [3],[8],[9],[11],[13]

Número	Nombre de la Unidad	Duración en Semanas
4	Interfaces al usuario	1,5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Diseño de interfaces al usuario • Técnicas y elementos disponibles • Programación de interfaces al usuario	Al término de la unidad, el alumno programa buenas interfaces .	[4],[11]

Número	Nombre de la Unidad	Duración en Semanas
5	Diseño de software orientado objetos	3

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
● Cómo encontrar objetos? ● Cómo definir si una funcionalidad debe ser un método o un objeto? ● Relaciones entre clases ● Diagramas de clases (UML) ● Cuando usar la relación de composicion? ● Cuando usar la relación de herencia. ● Patrones de diseño: adaptador, iterador, observador, puente, singleton y fábrica abstracta, entre otros. ● Evaluación de Diseños: métricas	Al término de la unidad, el alumno – diseña y programa usando buenas prácticas sistemas de software de mediana complejidad. - identifica relaciones entre clases y construye diagramas de clases apropiados en UML. - reconoce patrones de diseño. - aplica adecuadamente patrones de diseño en el desarrollo de una aplicación. - reconoce las distintas métricas existentes para la evaluación de software orientado a objetos, las aplica a un programa y analiza los resultados.	[4],[10]

Número	Nombre de la Unidad	Duración en Semanas
6	Validación de software orientado a objetos	1,5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
● Métodos de testing ● Testing y herencia ● Niveles de testing ● Testing el comportamiento de objetos. ● Verificación de consistencia en tiempo de ejecución. ● Validación estática	Al término de la unidad, el alumno - utiliza técnicas para hacer debugging de programas orientados a objetos. - utiliza un debugger .	[3]

Número	Nombre de la Unidad	Duración en Semanas
7	Proceso de desarrollo de software orientado a objetos	2,5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
• Proceso de desarrollo unificado (RUP): Guiado por los casos de uso, centrado en la arquitectura, iterativo e incremental. • Casos de uso • Diagramas análisis, de secuencia e interacción de objetos • Enfrentando un problema de mediana envergadura.	Al término de la unidad, el alumno - enfrenta el desarrollo de software orientado a objetos desde su etapa de análisis. - especifica casos de uso. -Crea diagramas de análisis y secuencia. - desarrolla software orientado a objetos construyendo cada una de sus etapas.	[2]

Bibliografía

[1] Bertrand Meyer. Object Oriented Software Construction. Prentice Hall. 1997. Second Edition.
[2] Ivar Jacobson, Grady Booch, James Rumbaugh. The unified software development process. Addison Wesley, 2000.
[3] A Eliëns. Principles of object oriented software development. Addison Wesley, 1995.
[4] Setrag Khoshafian, RazmikAbnous. Object orientation: concepts, languages, databases and user interfaces. John Wiley&Sons Inc. 1990.
[5] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of reusable object oriented software. Addison Wesley. 1995.
[6] Cay Horstmann. Object-Oriented Design & Patterns John Wiley & Sons, Inc., 2004.
[7] Timothy C. Lethbridge, Robert Laganiere. Object oriented software engineering. McGraw-hill Education. 2001
[8] Bjarne Stroustrup. What is object oriented programming? IEEE Software. 1988.
[9] Bertrand Meyer. Applying design by contract. Computer. 1990
[10] Daniel Halbert y Patrick O'Brien. Using types and inheritance in object oriented languages. European conference on object-oriented programming on ECOOP '87. 1987.
[11] Developing software for the user interface. The SEI series in software Engineering. Adisson Wesley. 1991.
[12] Bjarne Stroustrup. The {C}++ Programming Language}. Addison-Wesley, 2002. Edición especial.
[13] Java, http://sunsite.dcc.uchile.cl/SunSITE/java/docs/tut-java122-sun/index.html

Vigencia desde:	Otoño 2009
Elaborado por:	Nancy Hitschfeld Kahler
Revisado por:	ADD (noviembre 2009)