FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS UNIVERSIDAD DE CHILE

PROGRAMA DE CURSO

Código	Nombre			
CC4101	**Lenguajes de Programación**			
Nombre en Inglés				
Programming Languages				
SCT	Unidades Docentes	Horas de Cátedra	Horas Docencia Auxiliar	Horas de Trabajo Personal
6	10	3	1,5	5,5
Requisitos			Carácter del Curso	
CC3102 Teoría de la Computación			Obligatorio para Licenciatura en Computación.	
Resultados de Aprendizaje				
Al término del curso, el alumno demuestra que maneja los lenguajes de programación, sus aplicaciones, semántica, e implementación. Reconoce la definición de varios lenguajes a través de su intérprete, estudiando paso a paso distintos mecanismos, como funciones, recursión, estado, y finalmente objetos. Además, identifica mecanismos para extender lenguajes existentes, en particular macros. Como la gran mayoría del trabajo de definición de lenguajes se hace en Scheme, el alumno maneja este lenguaje, y programación funcional en general. Además, el curso entrega elementos precisos de comparación y descripción de varios lenguajes, algunos ampliamente usados y otros emergentes, como C, Java, Lisp, bash, Haskell, ML, JavaScript, Scala, Self, y Smalltalk.				

Metodología Docente	Evaluación General
Clases expositivas del profesor de cátedra. De manera de relacionar lo conceptual con lo concreto, el profesor combina explicación teórica en la pizarra con demostración en vivo de la implementación de los conceptos vistos, a través de la proyección con data show de la programación de varios lenguajes. Clases auxiliares dedicadas a repasar puntos delicados vistos en clase, explicar ejemplos más extensos, resolver ejercicios propuestos, y preparación pre y post controles.	Se realizan tres controles para evaluar si se han cumplido los objetivos. El primero evalúa las unidades 1 y 2, el segundo las unidades 3-6 y el tercero las unidades 7-9. El examen evalúa todas las unidades, en particular la parte de la 9 que no llega a evaluarse completamente en el control. Tanto los controles como el examen se enfocan en evaluar que el alumno haya comprendido los conceptos como sus implicancias concretas. La Nota de Control se calcula de la siguiente manera: Promedio ponderado del examen (40%) y del promedio de los controles (60%) Se realizan tres tareas a lo largo del curso. Las tareas consisten generalmente en implementar lenguajes con mecanismos varios, vistos o no en clase. Las tareas buscan fundamentalmente que el alumno comprenda como definir un lenguaje, y como/cuando aplicar los mecanismos ofrecidos por este.

Metodología Docente	Evaluación General
	Las tareas son individuales y se promedian a partes iguales para formar la nota de tareas. Controles y tareas se aprueban por separado y deben ser igual o superior a 4.0. La nota final es 2/3 de la nota de controles y 1/3 de la nota de tareas.

UNIDADES TEMATICAS

Número	Nombre de la Unidad	Duración en Semanas
1	Elementos básicos de programación funcional	2

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Definición inductiva de datos - Abstracción de datos - Definición de funciones - Manejo de listas y otras estructuras. - Uso de funciones de orden superior y primera clase.	Al término de la unidad se espera que el alumno: - Reactualice la metodología de programación basada en abstracción de datos, y el procesamiento de datos por funciones recursivas. - Comprenda el uso práctico de funciones de orden superior y de primera clase. Conexión con: C, Java	[2] Cap 1-2 [3] [4]

Número	Nombre de la Unidad	Duración en Semanas
2	Substitución y funciones de primer orden	2

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Interpretar un lenguaje de cálculo aritmético. - Introducción de identificadores en el lenguaje. - Substitución: definición del significado de los identificadores. - Substitución perezosa o temprana. - Funciones simples (primer orden).	Al término de la unidad se espera que el alumno: - Comprenda la estructura básica de un intérprete. - Comprenda el significado de identificadores a través de substitución. Implementa un lenguaje con funciones simples.	[1] Cap 1-4

Número	Nombre de la Unidad	Duración en Semanas
3	Funciones de primera clase y alcance	1.5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Introducción de ambientes para diferir substitución. - Control del alcance de las variables. - Alcance estático vs. dinámico. - Funciones de primera clase. - Funciones anónimas. "Closures" vs. punteros de función.	Al término de la unidad se espera que el alumno: - Reconozca la noción de alcance de los identificadores. - Comprenda la diferencia entre alcance estático y dinámico, y en que casos sirven. Comprenda como proveer funciones de primera clase preservando alcance estático. Conexión con: Common Lisp, bash, TeX, excepciones, Java, C.	[1] Cap 5-6 [2] Cap 3.1-3.5

Número	Nombre de la Unidad	Duración en Semanas
4	Regímenes de evaluación	2

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Evaluación perezosa. - Introducción al lenguaje Haskell. - Manejo de estructuras infinitas. - Implementar evaluación perezosa. - Transparencia referencial y razonamiento ecuacional.	Al término de la unidad se espera que el alumno: - Maneje distintas estrategias de evaluación. Comprenda en que casos evaluación perezosa es útil. Reconozca la importancia del razonamiento ecuacional. Conexión con: Haskell, ML, bash.	[1] Cap 7-8

Número	Nombre de la Unidad	Duración en Semanas
5	Recursión	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Funciones recursivas de primera clase. - Ambiente recursivo y punto fijo. - Recursión e iteración: recursión por la cola y optimizaciones. - Definición puramente funcional de la recursión.	Al término de la unidad se espera que el alumno: - Comprenda la noción de recursividad, tanto en términos formales como prácticos. - Comprenda la noción de recursión por la cola. Conexión con: Scheme vs. Java	[1] Cap 9-10 [2] Cap 3.6

Número	Nombre de la Unidad	Duración en Semanas
6	Representación procedural y meta-interpretación	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Usar funciones en vez de estructura de datos: abstracción procedural. - Aplicación a ambientes y definición del intérprete. - Interprete sintáctico, interprete meta. - Extender un lenguaje por absorción de primitivas.	Al término de la unidad se espera que el alumno: - Reconozca la relación entre abstracción de datos y abstracción procedural. - Comprenda distintas estrategias para definir o extender un lenguaje.	[1] Cap 11

Número	Nombre de la Unidad	Duración en Semanas
7	Estado y mutación	2.5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Mutación y orden de evaluación. - Estructuras de dato mutables. - Variables y asignación. - Extensión del modelo con el “almacén” (store). - Patrón de hilamiento en el intérprete. - Estrategias de paso de parámetros. - Recolección de basura.	Al término de la unidad se espera que el alumno: - Entienda cuando manejar estado es necesario, y porque es preferible evitarlo. - Reconozca diferentes formas de mutación, y diferentes semánticas para pasar parámetros. Entender los principios básicos de la gestión de memoria. Conexión con: C, Java	[1] Cap 12-14 [2] Cap 3.7-3.9 [1] Cap 20

Número	Nombre de la Unidad	Duración en Semanas
8	Extensión sintáctica de lenguajes	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Motivación: lenguajes específicos y lenguajes extensibles. - Introducción a macros. - Macros higiénicas, ejemplos.	Al término de la unidad se espera que el alumno: - Comprenda la ventaja de proveer las buenas abstracciones a los programadores. - Entienda los sistemas de macros higiénicas para extender lenguajes.	[1] Cap 35-37

Número	Nombre de la Unidad	Duración en Semanas
9	Objetos	2

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
- Abstracción procedural y objetos. - Objeto como unidad de encapsulación - Interfaz de un objeto - Recursión y self - Delegación y prototipos - Clases y herencia - Opciones de diseño para OOP	Al término de la unidad se espera que el alumno: - Comprenda en profundidad la conexión entre abstracción procedural y objetos. Entiende la esencia de la orientación a objetos y sus múltiples facetas. - Implementa un sistema de objetos usando macros. Conexión con: Java, JavaScript, Scala, Self, Smalltalk	[2] Cap 5 [5]

Bibliografía

[1] S. Krishnamurthi. Programming Languages: Application and Interpretation. Online. 2007. http://www.cs.brown.edu/~sk/Publications/Books/ProgLangs/

[2] D. Friedman, M. Wand, C. Haynes. Essentials of Programming Languages, 2nd edition. MIT Press. 2001.

[3] D. Friedman, M. Felleisen, G. Sussman. The Little Schemer, 4th edition. MIT Press. 1995.

[4] D. Sitaram. Teach Yourself Scheme in Finum Days. Online. 2004. http://www.ccs.neu.edu/home/dorai/t-y-scheme/

[5] T. D'Hondt. Principles of Object Oriented Languages. Online. 2004. http://prog.vub.ac.be/~tjdhondt/POOL/HTM.dir/notes.htm

Vigencia desde:	2010
Elaborado por:	Éric Tanter
Revisado por:	ADD (noviembre 2009)