FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS UNIVERSIDAD DE CHILE

PROGRAMA DE CURSO

Código	Nombre
CC7125	INTRODUCCIÓN A COQ: LÓGICA, TIPOS Y VERIFICACIÓN
Nombre en Inglés	
INTRODUCTION TO COQ: LOGIC, TYPES AND VERIFICATION	

SCT	Unidades Docentes	Horas de Cátedra	Horas Docencia Auxiliar	Horas de Trabajo Personal
6	10	3	1.5	5,5

Requisitos	Carácter del Curso
CC4101 / (MA3705 / MA4702) / Autor El curso no presume ningún conocimiento previo en lógica o lenguajes de programación, aunque un cierto grado de madurez matemática y/o de exposición a lenguajes será de ayuda.	Electivo para Magister y Doctorado en Computación. Electivo avanzado para pregrado.

Resultados de Aprendizaje

El alumno manejará las bases matemáticas de la construcción de software confiable, incluyendo conceptos básicos de lógica, demostraciones de teoremas asistidas por computador, el asistente de pruebas Coq, programación funcional, semántica operacional, lógica de Hoare, y sistemas de tipos estáticos. Dado que el contenido del curso es 100% formalizado y verificado por la máquina, el alumno tendrá una sólida experiencia inicial en el uso de Coq, tanto para definir programas o lenguajes, como para enunciar propiedades formales precisas al respecto, y producir demostraciones rigurosas automáticamente verificadas de dichas propiedades.

Metodología Docente	Evaluación General
Clases expositivas del profesor de cátedra, que incluyen momentos dedicados a repasar puntos delicados vistos anteriormente, explicar ejemplos extensos, y resolver ejercicios. Para relacionar lo conceptual con lo práctico, el profesor combina explicación teórica en la pizarra con exposición en vivo de la implementación de los conceptos vistos, a través de la proyección con data show de la programación de varios ejemplos ilustrando dichos conceptos. Se incita a los alumnos a que lleven consigo un computador para que puedan programar los ejemplos y ejercicios de manera directa, fomentando así el aprendizaje activo y la interacción en clases. Las clases auxiliares, opcionales, serán clases prácticas de ayuda para la realización de las mini-tareas en Coq.	Se realiza una evaluación continua a través de “mini-controles” y “mini-tareas” regulares, en particular cada vez que se completa el estudio de una unidad temática. Los mini-controles tienen una duración de 15-30 min. Apuntan a evaluar la buena asimilación de los conceptos, con preguntas enfocadas y pequeños ejercicios de programación/formalización. Las notas de controles se promedian para formar la nota de control, que cuenta por 50% de la nota final. Las mini-tareas son ejercicios en Coq, para validar la capacidad del alumno a aplicar los conceptos y mecanismos vistos en clase en ejemplos de complejidad mediana. Las tareas son individuales y se promedian a partes iguales para formar la nota de tareas, que cuenta por 30% de la nota final. Para aprobar el curso, la nota de control y la nota de tareas deben ser ambas igual o superior a 4.0. Se realizará además un proyecto final, que consiste de un estudio de un tema avanzado a elección del alumno, con una presentación final delante del curso. Cuenta por 20% en la nota final. Si el promedio final del alumno es igual o superior a 5.0, éste no tendrá que rendir un examen final.

FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS UNIVERSIDAD DE CHILE

Unidades Temáticas

Número	Nombre de la Unidad	Duración en Semanas
1	Programación Funcional en Coq	0.5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Tipos de datos y definiciones básicas Demostraciones por simplificación, por reescritura, por análisis de casos	Nivel básico de programación en Coq y demostraciones simples con tipos definidos por casos.	[1] Basics.html [2]

Número	Nombre de la Unidad	Duración en Semanas
2	Demostraciones por Inducción	0.5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Demostraciones por inducción Demostraciones dentro de demostraciones	Manejo de demostraciones por inducción en Coq, y organización de demostraciones en sub-demostraciones.	[1] Induction.html [2]

Número	Nombre de la Unidad	Duración en Semanas
3	Datos Estructurados	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Pares y Listas Razonar sobre listas Opciones Mapas parciales	Modelar y razonar sobre estructuras de datos estándares.	[1] Lists.html [2]

Número	Nombre de la Unidad	Duración en Semanas
4	Polimorfismo y Funciones de Orden Superior	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Listas polimórficas, pares polimórficos Inferencia y argumentos implícitos Funciones como datos, map, fold, etc.	Usar Coq como un lenguaje de programación funcional polimórfico.	[1] Poly.html [2]

Número	Nombre de la Unidad	Duración en Semanas
5	Tácticas Básicas	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Tácticas esenciales: apply, inversión, unfold, destruct Controlar la hipótesis de inducción Resumen de tácticas básicas	Manejar técnicas básicas de demostraciones sobre programas funcionales, y saber detectar cuándo y cómo ajustar una hipótesis de inducción.	[1] Tactics.html [2]

Número	Nombre de la Unidad	Duración en Semanas
6	Lógica y Proposiciones Inductivas	1.5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Conectivos lógicos: conjunción, disyunción, falsedad, negación, verdad, equivalencia, cuantificación existencial Programar con proposiciones Coq vs teoría de conjuntos Proposiciones y relaciones inductivas Usar evidencia en demostraciones	Describir y demostrar formulas lógicas (proposiciones) en Coq. Entender diferencias entre la lógica de Coq y la teoría de conjuntos. Definir y demostrar relaciones inductivas.	[1] Logic.html [1] IndProp.html [2]

Número	Nombre de la Unidad	Duración en Semanas
7	Correspondencia de Curry-Howard	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Scripts y objetos de demostraciones Conectivos lógicos como tipos inductivos Igualdad	Entender la conexión entre tipos y proposiciones, y entre programas y demostraciones.	[1] ProofObjects.html [2], [4]

Número	Nombre de la Unidad	Duración en Semanas
8	Programas Imperativos y Equivalencia	1,5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
El mini-lenguaje IMP Automatización de demostraciones Relación de evaluación de IMP Variables y comandos Razonar sobre programas IMP Equivalencia comportamental Transformaciones de programas	Saber modelar un lenguaje simple y establecer propiedades sobre programas, en particular equivalencia entre programas. Definir transformaciones de programas y demostrar que son correctas.	[1] Imp.html [1] Equiv.html

Número	Nombre de la Unidad	Duración en Semanas
9	Lógica de Hoare	1.5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Aserciones y triplas de Hoare Reglas de demostraciones Programas decorados Invariantes de bucles Precondiciones más débiles (según tiempo)	Manejar los principios básicos de la lógica de Hoare para hacer verificación de programas imperativos.	[1] Hoare.html [1] Hoare2.html

Número	Nombre de la Unidad	Duración en Semanas
10	Semántica Operacional	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
El mini-lenguaje TOY Reducción paso a paso Formas normales y progreso Una máquina a pilas, paso a paso	Entender la diferencia entre semánticas de paso chicos y de pasos grandes. Modelar lenguajes y máquinas paso a paso.	[1] Smallstep.html [4]

Número	Nombre de la Unidad	Duración en Semanas
11	Automatización de Demostraciones	0.5

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Automatización de demostraciones: táctica auto, calce de hipótesis con Ltac, variables existenciales con eapply y eauto.	Mejorar técnicas de automatización parcial de las demostraciones.	[1] Auto.html

Número	Nombre de la Unidad	Duración en Semanas
12	Sistemas de Tipos	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Expresiones aritméticas tipadas Coherencia de los tipos con progreso y preservación	Entender los conceptos básicos de los sistemas de tipos, así como la propiedad fundamental (coherencia) de esos sistemas.	[1] Types.html [4]

Número	Nombre de la Unidad	Duración en Semanas
13	Lambda Cálculo Tipado (STLC)	2

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Sintaxis, semántica, y tipos simples para el lambda cálculo Propiedades del STLC: progreso, preservación, unicidad de tipos Verificador de tipos para STCL	Aplicar los conceptos fundamentales de sistemas de tipos al lenguaje esencial de los lenguajes de programación modernos: el lambda cálculo. Entender la diferencia entre una especificación (p.ej. como relación) y un algoritmo.	[1] Stlc.html [1] StlcProp.html [1] Typechecking.html [4]

Número	Nombre de la Unidad	Duración en Semanas
14	Tópicos Avanzados	1

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
Esta última unidad permite al alumno explorar, a su elección, algún tema más avanzado cubierto en el material de referencia, y luego exponerlo a sus compañeros como proyecto final. Puede ser: - STLC: referencias, subtipos, records, o normalización - Coq: tácticas avanzadas y customizadas - programación y verificación con tipos dependientes	Aplicar el conocimiento adquirido durante el curso para estudiar de manera autónoma un tema avanzado y exponerlo antes una audiencia de nivel similar. Los resultados técnicos específicos dependerán de la elección del alumno.	[1] References.html [1] Sub.html [1] RecordSub.html [1] Norm.html [1] UseTactics.html [3] Part II, Part III, [4]

Bibliografía

Referencia principal del curso:

[1] B. Pierce et al., Software Foundations (version 4.2, 2017).
Available online: https://www.cis.upenn.edu/~bcpierce/sf/

Material complementario:

[2] Interactive Theorem Proving and Program Development, Y. Bertot and P. Casteran, Springer, 2004.
[3] Certified Programming with Dependent Types, A. Chlipala, MIT Press, 2013.
Available online: http://adam.chlipala.net/cpdt/
[4] Types and Programming Languages, B. Pierce, MIT Press, 2001.
[5] The Coq reference manual: http://coq.inria.fr

Vigencia desde:	Mayo 2017
Elaborado por:	Eric Tanter