

EX-2024-00149385- -UNC-ME#FAMAF

PROGRAMA DE ASIGNATURA	
ASIGNATURA: Lenguajes y Compiladores	AÑO: 2024
CARACTER: Obligatoria	UBICACIÓN EN LA CARRERA: 5° año 1° cuatrimestre
CARRERA: Licenciatura en Ciencias de la Computación	
REGIMEN: Cuatrimestral	CARGA HORARIA: 120 horas

ASIGNATURA: Lenguajes y Compiladores	AÑO: 2024
CARACTER: Optativa	UBICACIÓN EN LA CARRERA: 5° año 1° cuatrimestre
CARRERA: Licenciatura en Matemática Aplicada	
REGIMEN: Cuatrimestral	CARGA HORARIA: 120 horas.

FUNDAMENTACIÓN Y OBJETIVOS

El establecer el significado de las frases de un lenguaje de programación es un problema de múltiples aristas en tanto puede tener variados objetivos, que van desde la necesidad de comprensión humana, hasta el imperativo de que una máquina los pueda interpretar o traducir a una secuencia de instrucciones ejecutables. Un manual de usuario/a, una estrategia de compilación, o alguna herramienta teórica destinada a desentrañar los principios básicos de su diseño, constituyen todas vertientes de significado que responden a distintos intereses y usos de los lenguajes de programación. En las últimas décadas variados desarrollos matemáticos y lógicos dieron forma a una teoría que se posicionó en un lugar privilegiado para el acceso a la comprensión profunda del significado de un lenguaje. La misma permite conectar la descripción intuitiva de un sentido finito y dinámico (un manual), con una modalidad estática del significado, vigente en la lógica formal y la matemática (denotación). A partir del desarrollo de la Teoría de Dominios la semántica denotacional adquiere una relevancia especial, no sólo por tratarse de objetos matemáticos perfectamente definidos en el contexto de una teoría particular, sino además porque comienza a ser utilizada como "la definición" del lenguaje y luego, si se proponen otras semánticas (operacional, axiomática), se las demuestra correctas con respecto a dicha definición.

El objetivo general de la asignatura es lograr que los/las estudiantes se apropien de las herramientas más importantes que actualmente se utilizan para definir sintácticamente y dar significado a las frases de un lenguaje de programación, poniendo énfasis en la utilidad de estas herramientas para comprender los principios básicos que subyacen en su diseño.

Dentro de los objetivos específicos, mencionamos como relevantes:

- apropiarse de conceptos fundamentales sobre la estructura gramatical de lenguajes de programación, tales como sintaxis abstracta, variables ligadas y alcance,
- tomar contacto con un lenguaje teórico basado en Standard ML, en tanto lenguaje que ha sido definido formalmente de manera completa, y cuyos principios básicos coinciden con los lenguajes más populares,
- acceder al uso de herramientas matemáticas apropiadas para el estudio de los lenguajes de programación,
- disponer de recursos para evaluar las características principales de lenguajes cercanos a lenguajes reales actualmente en uso,
- reconocer propiedades deseables en lenguajes de programación y las herramientas para garantizarlas,
- proveer de recursos para que el/la estudiante pueda diseñar e implementar lenguajes de programación.

CONTENIDO

I - Herramientas básicas para dar significado a los lenguajes de programación

- Nociones en relación a la sintaxis: gramática, gramática abstracta, sintaxis abstracta, lenguaje y metalenguaje.
- Distintas formas de dar significado a los lenguajes de programación. Semántica denotacional: las nociones de frase, dominio semántico y función semántica. Semántica operacional: las nociones de configuración, regla de transición y ejecución.
- Nociones en relación a la definición del significado: dirección por sintaxis, semántica composicional.
- Variables y ligadura (en la lógica de predicados). Sustitución y el problema de la captura. Propiedades de coincidencia y renombre.
- El problema de dar significado a ecuaciones recursivas. Dominios, función continua y teorema del menor punto fijo. Análisis de las soluciones de una ecuación recursiva a la luz del teorema del menor punto fijo.

II - Lenguajes imperativos

- Conjunto de estados. Semántica denotacional de las construcciones básicas de un lenguaje imperativo.
- El problema de dar significado a la iteración. Significado de la iteración utilizando el teorema del menor punto fijo.
- Propiedades de coincidencia y renombre.
- Fallas y manejo de excepciones. Output. Input.
- Semántica operacional para el lenguaje imperativo.
- Corrección respecto de la semántica denotacional.

III - Lenguajes aplicativos

- Las nociones de reducción y evaluación en el Cálculo Lambda. El problema de la terminación. La noción de forma canónica. Modalidad de evaluación: Eager y Normal.
- El problema de la semántica denotacional: el modelo D infinito y sus variantes.
- Lenguaje aplicativo. Sintaxis. Semántica operacional eager y normal: la noción de evaluación, formas canónicas y reglas de evaluación. Tratamiento de errores.
- Semántica denotacional directa del lenguaje aplicativo. Sintaxis y semántica de la recursión en las modalidades eager y normal. Propiedades.

IV - Lenguajes aplicativos con una componente imperativa.

- Los problemas de la combinación de paradigmas.
- Las nociones de estado, ambiente, identificador y variable.
- Un lenguaje que combina los paradigmas. Semántica denotacional y operacional.
- Construcciones imperativas como abreviaturas. Propiedades.
- Funciones y procedimientos. Pasaje de parámetros.

V - Otros tópicos de interés

- Sistema de tipos simples para el cálculo lambda.
- Semántica de continuaciones.
- Tuplas, patrones y unión disjunta en los lenguajes aplicativos puros.

BIBLIOGRAFÍA

BIBLIOGRAFÍA BÁSICA

Fridlender, Daniel y Gramaglia Héctor. Apuntes de Cátedra (basados en el libro de Reynolds). 2022.

Reynolds, John. Theories of Programming Languages, Cambridge University Press, 1998.

BIBLIOGRAFÍA COMPLEMENTARIA

EX-2024-00149385- -UNC-ME#FAMAF

Harper, R. Practical foundations for programming languages. Cambridge University Press, 2013.
Tennet, R., Semantic of Programming Languages, Prentice Hall. 1991.
Hindley, R, Selding, J. Lambda-Calculus and Combinators, an Introduction, Cambridge University Press, 2008

EVALUACIÓN

FORMAS DE EVALUACIÓN

La materia cuenta con un aula virtual donde se encuentra información más detallada, como las fechas de las evaluaciones.

- Se tomarán 2 (dos) exámenes parciales. Las evaluaciones parciales serán sobre contenidos teórico-prácticos. El formato de estas evaluaciones consistirá en la resolución de actividades en el aula.
- Los/as estudiantes realizarán un taller (como parte de la carga horaria de práctico) que consiste en la elaboración de un intérprete de un lenguaje de programación.
- La materia contempla un régimen de promoción. El/la estudiante accederá a la condición de estudiante promocional con la aprobación de los dos parciales con promedio al menos 7, y con una nota de 6 puntos o más en cada evaluación. Además deberá aprobar el coloquio al finalizar el cursado.
- La aprobación de la materia se dará por promoción, o mediante la aprobación de un examen final en las fechas destinadas a exámenes en el calendario académico. El examen final constará de una evaluación escrita con un formato similar al de los parciales sobre contenidos teórico-prácticos; en caso que el tribunal lo considere necesario, puede complementarse el examen teórico con preguntas orales.

REGULARIDAD

Aprobar las dos evaluaciones parciales o sus correspondientes recuperatorios. Se podrá recuperar uno de los dos parciales en la última semana de dictado de la materia. Aprobar el taller.

PROMOCIÓN

Aprobar todas las evaluaciones parciales con una nota no menor a 6 (seis), y obteniendo un promedio no menor a 7 (siete). Aprobar el taller. Aprobar el coloquio.

CORRELATIVIDADES

Como Optativa de la Licenciatura en Matemática Aplicada:

Para Cursar:

Algoritmos y Estructuras de Datos - Aprobada

Matemática Discreta II - Regular

Para Rendir:

Algoritmos y Estructuras de Datos - Aprobada

Matemática Discreta II - Aprobada