PROGRAMA de Programación Funcional

Carreras: Tecnicatura Universitaria en Programación Informática - Licenciatura en Informática - Licenciatura en Bioinformática

Asignatura: Programación Funcional

Núcleo al que pertenece: Básico

Profesor: Pablo E. Martínez López

Asignaturas Correlativas: Estructuras de Datos

Objetivos:

Se espera que quien curse la materia:

- Comprenda, maneje y se familiarice con conceptos fundamentales de la programación y su importancia en la tarea de programar:
  1. Abstracción mediante funciones (como elementos que transforman información),
  2. Noción de funciones de alto orden y su utilidad, currificación, esquemas de programas,
  3. Inducción (y las herramientas asociadas: inducción estructural y recursión),
  4. Nociones de sistemas de tipos y su utilidad práctica,

- Sea capaz de utilizar dichas nociones para la confección de programas sencillos en un lenguaje funcional.

- Sea capaz de demostrar propiedades sencillas de programas funcionales utilizando inducción estructural.

- Sea capaz de aplicar técnicas de transformación de programas en casos particulares.

Contenidos mínimos:

- Nociones generales del paradigma funcional:

  Valores y expresiones. Las funciones como valores. Mecanismos de definición de expresiones y valores. Ecuaciones orientadas para definir funciones. Sintaxis. Sistemas de tipos Hindley-Milner. Tipos básicos. Constructores de tipos. Polimorfismo. Sintaxis para valores de cada tipo (caracteres, tuplas, listas, strings, funciones). Funciones parciales y totales. Funciones de alto orden. Currificación.

- Inducción y recursión:

Definición inductiva de conjuntos. Definición recursiva de funciones sobre conjuntos. Demostraciones inductivas sobre dichas funciones. Ejemplos: programas, expresiones aritméticas, listas.

- Listas:

  Listas como tipo inductivo. Funciones básicas sobre listas (append, head, tail, take, drop, reverse, sort, elem, etc.). Funciones de alto orden sobre listas. Patrón de recorrido: map. Patrón de selección: filter. Patrón de recursión: foldr. Listas por comprensión. Demostración de propiedades de listas y funciones sobre listas.

- Sistemas de tipos:

  Nociones básicas. Sistemas de tipado fuerte. Ventajas y limitaciones de los lenguajes de programación con tipos. Lenguaje de tipos. Asignación de tipos a expresiones. Propiedades interesantes de esta asignación. Algoritmo de inferencia. Mecanismos de definición de tipos nuevos y de funciones sobre ellos. Tipos algebraicos recursivos. Ejemplos: enumeraciones, listas, árboles binarios y generales.

- Transformación de Programas:

  Motivación. Obtención de programas a partir de especificaciones. Mejoramiento de eficiencia, con corrección por construcción. Técnicas particulares de transformación: tupling, eliminación de recursión, fusión. Transformación de listas por comprensión en expresiones que utilizan map, filter y concat.

- Lambda Cálculo:

  Definición del lenguaje. Sintaxis. Definición de sustitución. Modelo de computación. Nociones de alfa, beta y eta reducción. Semántica operacional. Lambda cálculo como modelo teórico de los lenguajes funcionales. Representación de booleanos, pares, números naturales, listas y otras construcciones.

Carga horaria semanal: 4 hs

Programa analítico:

- Nociones generales del paradigma funcional:

  Valores y expresiones. Las funciones como valores. Mecanismos de definición de expresiones y valores. Ecuaciones orientadas para definir funciones.Sintaxis. Sistemas de tipos Hindley-Milner. Tipos básicos.

Constructores de tipos. Polimorfismo. Sintaxis para valores de cada tipo (caracteres, tuplas, listas, strings, funciones). Funciones parciales y totales. Funciones de alto orden. Currificación.

- Inducción y recursión:

  Definición inductiva de conjuntos. Definición recursiva de funciones sobre conjuntos. Demostraciones inductivas sobre dichas funciones. Ejemplos: programas, expresiones aritméticas, listas.

- Listas:

  Listas como tipo inductivo. Funciones básicas sobre listas (append, head, tail, take, drop, reverse, sort, elem, etc.). Funciones de alto orden sobre listas. Patrón de recorrido: map. Patrón de selección: filter. Patrón de recursión: foldr. Listas por comprensión. Demostración de propiedades de listas y funciones sobre listas.

- Sistemas de tipos:

  Nociones básicas. Sistemas de tipado fuerte. Ventajas y limitaciones de los lenguajes de programación con tipos. Lenguaje de tipos. Asignación de tipos a expresiones. Propiedades interesantes de esta asignación. Algoritmo de inferencia. Mecanismos de definición de tipos nuevos y de funciones sobre ellos. Tipos algebraicos recursivos. Ejemplos: enumeraciones, listas, árboles binarios y generales.

- Transformación de Programas:

  Motivación. Obtención de programas a partir de especificaciones. Mejoramiento de eficiencia, con corrección por construcción. Técnicas particulares de transformación: tupling, eliminación de recursión, fusión. Transformación de listas por comprensión en expresiones que utilizan map, filter y concat.

- Lambda Cálculo:

  Definición del lenguaje. Sintaxis. Definición de sustitución. Modelo de computación. Nociones de alfa, beta y eta reducción. Semántica operacional. Lambda cálculo como modelo teórico de los lenguajes funcionales. Representación de booleanos, pares, números naturales, listas y otras construcciones.

Bibliografía obligatoria:

- Lipovaca, M. Learn you a haskell for great good!: A beginner's guide. No starch press. 1st Edition. 2011
- Christopher Allen, Julie Moronuki. Haskell programming from first principles. Allen and Moronuki Publishing, 2016
- Bird, R. Thinking functionally with Haskell. Cambridge University Press. 2014
- Michaelson, G. An introduction to functional programming through lambda calculus. Courier Corporation. 2011
- Bird Richard, Wadler Philip, Introduction to Functional Programming. First Edition, Prentice Hall, Oxford, 1998.
- A. Davie, An Introduction to Functional Programming Systems using Haskell, Cambridge University Press, 1992.
- Huday, Peterson and Fasel, A Gentle Introduction to Haskell. First Edition, Prentice Hall, 2000. URL: http://www.haskell.org/tutorial/
- John Hughes, Why Functional Programming Matters? Computer Journal 32 (2), 1989.

Bibliografía de consulta:

- Simon Thompson, The Craft of Functional Programming. Addison Wesley, 3ra Edición, 2011
- Peter A. Fejer, Dan Simovici, Mathematical Foundations of Computer Science. Volume I: Sets, Relations and Induction. Springer Verlag, 1991.

Organización de las clases:

Las clases están organizadas en dos partes, teóricas y prácticas. Durante las teóricas se presenta el contenido formal y se ahonda en las características del paradigma funcional. Las clases teóricas están acompañadas por un apunte en forma de diapositivas que sirve a quienes cursen la materia como material de referencia más allá de la bibliografía sugerida. Durante las clases prácticas se resuelven ejercicios de programación haciendo hincapié en las características de los lenguajes de

programación funcional. Los ejercicios son resueltos en el pizarrón y discutidos con las personas que cursan, quienes pueden probar su funcionamiento en las computadoras del laboratorio. Además, se responden consultas, principalmente las generadas por los trabajos prácticos de la materia.

Los trabajos prácticos cubren los contenidos de la siguiente manera:

Práctica 1: Modelo de cómputo

TEMAS: Valores y expresiones. Las funciones como valores. Mecanismos de definición de expresiones y valores. Ecuaciones orientadas para definir funciones. Sintaxis. Funciones de alto orden. (Del ítem “Nociones generales del paradigma funcional”)

OBJETIVOS: El objetivo de esta práctica es presentar las nociones iniciales elementales del paradigma funcional, tales como valores y expresiones, funciones como valores, mecanismos de definición mediante ecuaciones orientadas, algunas nociones iniciales de sintaxis, y presentar la idea de funciones de alto orden.

Práctica 2: Sistemas de tipos

TEMAS: Sistemas de tipos Hindley-Milner. Tipos básicos. Constructores de tipos. Polimorfismo. Sintaxis para valores de cada tipo (caracteres, tuplas, listas, strings, funciones). (Del ítem “Nociones generales del paradigma funcional”) Nociones básicas. Sistemas de tipado fuerte. Ventajas y limitaciones de los lenguajes de programación con tipos. Lenguaje de tipos. Asignación de tipos a expresiones. Propiedades interesantes de esta asignación. Algoritmo de inferencia. (Del ítem “Sistemas de tipos”)

OBJETIVOS: Presentar la idea de sistemas de tipos fuertes, su utilización en programación y sus ventajas. Presentar el sistema de tipos Hindley-Milner, la idea de asignación de tipo y de inferencia de tipos en este sistema, y las ideas de tipos funcionales, polimorfismo, constructores de tipo, así como la sintaxis de los valores de cada tipo.

Práctica 3: Currificación

TEMAS: Currificación.

OBJETIVOS: Incorporar el concepto de currificación y su utilidad, y familiarizarse con la notación específica que lo facilita.

Práctica 4: Reducción

TEMAS: Funciones parciales y totales. Funciones estrictas y no estrictas

OBJETIVOS: Presentar el mecanismo de cómputo por reducción, y conocer sus

límites. Descubrir la existencia de funciones parciales y totales y de funciones estrictas y no estrictas como consecuencia de esos límites.

Práctica 5: Tipos algebraicos

TEMAS: Constructores de tipos. (Del ítem "Nociones generales del paradigma funcional"). Mecanismos de definición de tipos nuevos y de funciones sobre ellos. (Del ítem "Sistemas de tipos")

OBJETIVOS: Practicar la definición de tipos nuevos de datos y de funciones sobre ellos, en particular de tipos algebraicos no recursivos, a través del uso de constructores.

Práctica 6: Propiedades y demostraciones

TEMAS: Propiedades y demostraciones (no inductivas).

OBJETIVOS: Repasar la idea de propiedades y demostraciones, establecer las reglas específicas utilizadas en el paradigma funcional (e.g. principio de extensionalidad) y practicar la confección de demostraciones y su interpretación.

Práctica 7: Inducción/Recursión I

TEMAS: Definición inductiva de conjuntos. Definición recursiva de funciones sobre conjuntos. Demostraciones inductivas sobre dichas funciones. Ejemplos: programas, expresiones aritméticas, listas. (Del ítem "Inducción y Recursión")

OBJETIVOS: Realizar la definición inductiva de conjuntos, la definición recursiva de funciones y las demostración de propiedades inductivas a través de tipos algebraicos recursivos simples (lineales) no parametrizados.

Práctica 8: Inducción/Recursión II

TEMAS: Listas como tipo inductivo. Funciones básicas sobre listas (append, head, tail, take, drop, reverse, sort, elem, etc.). Demostración de propiedades de listas y funciones sobre listas. (Del ítem "Listas").

OBJETIVOS: Presentar el tipo inductivo de las listas como una estructura de datos recursiva parametrizada y trabajar la utilización de listas para diferentes propósitos en la confección de programas. Practicar la definición de funciones recursivas y la demostración de propiedades por inducción sobre listas.

Práctica 9: Inducción/Recursión III

TEMAS: Tipos algebraicos recursivos. Ejemplos: enumeraciones, listas, árboles binarios y generales. (Del ítem "Sistemas de tipos")

OBJETIVOS: Presentar tipos recursivos más complejos, de estructura arbórea, tanto parametrizados como sin parametrizar.

Práctica 10: Inducción/Recursión IV

TEMAS: Tipos algebraicos recursivos. Ejemplos: enumeraciones, listas, árboles binarios y generales. (Del ítem “Sistemas de tipos”)

OBJETIVOS: Combinar las técnicas de definición de funciones sobre tipos algebraicos recursivos con funciones de alto orden de forma simple.

Práctica 11: Esquemas de funciones I

TEMAS: Funciones de alto orden sobre listas. Patrón de recorrido: map. Patrón de selección: filter. Patrón de recursión: foldr. Listas por comprensión. Demostración de propiedades de listas y funciones sobre listas. (Del ítem “Listas”)

OBJETIVOS: Presentar la noción de esquemas de funciones ejemplificando sobre listas, mediante funciones de map, filter y foldr (patrón de recursión). Definir la noción de listas por comprensión y continuar trabajando demostraciones por inducción sobre la estructura de las listas.

Práctica 12: Esquemas de funciones II

TEMAS: Esquemas sobre árboles. Patrón de recursión estructural sobre árboles.

OBJETIVOS: Extender la noción de esquemas de funciones a árboles, y generalizar la idea de patrón de recursión como función (fold). Continuar trabajando demostraciones por inducción sobre estructuras arbóreas.

Práctica 13: Mónadas

TEMAS: Definición de mónadas. Propiedades de las mónadas. Mónada de error, mónada reader, monada writer. Aplicaciones de las mónadas.

OBJETIVOS: Escribir diferentes aplicaciones que requieren mónadas y definir mónadas básicas.

Práctica 14: Lambda cálculo

TEMAS: Lambda Cálculo. Definición del lenguaje. Sintaxis. Definición de sustitución. Modelo de computación. Nociones de alfa, beta y eta reducción. Semántica operacional. Lambda cálculo como modelo teórico de los lenguajes funcionales. Representación de booleanos, pares, números naturales, listas y otras construcciones.

OBJETIVOS: Presentar los mecanismos de trabajo del lambda cálculo, y focalizar sobre cómo escribir programas usuales en este lenguaje de funciones.

Práctica 15: Derivación de programas

TEMAS: Motivación. Obtención de programas a partir de especificaciones. Mejoramiento de eficiencia, con corrección por construcción. Técnicas particulares de transformación: tupling, eliminación de recursión, fusión. Transformación de listas por comprensión en expresiones que utilizan map, filter y concat. (Del ítem "Transformación de programas")

OBJETIVOS: Realizar derivaciones y síntesis de programas sencillas, practicando las técnicas vistas (tupling, eliminación de recursión, fusión, eliminación de listas por comprensión).

Modalidad de evaluación:

Los mecanismos de evaluación en modalidades libre y presencial de esta asignatura están reglamentados según los artículos del Régimen de estudios de la UNQ (Res. CS 201/18). En la modalidad de libre se evaluarán los contenidos de la asignatura con un examen escrito, un examen oral, un desarrollo de programa en el entorno de trabajo e instancias de evaluación similares a las realizadas en la modalidad presencial.

CRONOGRAMA TENTATIVO

Semana	Tema/unidad	Actividad*				Evaluación
		Teórico	Práctico			
			Res Prob.	Lab.	Otros Especificar	
1	Modelo de cómputo funcional	x	x			
2	Sistemas de tipos	x	x			
3	Currificación	x	x			
4	Reducción	x	x			
5	Tipos algebraicos	x	x	x		
6	Propiedades y demostraciones	x	x			
7	Inducción y recursión estructural	x	x			
8	Tipos algebraicos recursivos lineales	x	x	x		
9	Tipos algebraicos recursivos: árboles	x	x	x		
10	Consultas y Primer parcial				x	x
11	Esquemas de recursión en listas	x	x			
12	Esquemas de recursión en árboles	x	x	x		
13	Mónadas	x	x	x		
14	Lambda Cálculo	x	x			
15	Derivación de programas	x	x			
16	Consultas y segundo parcial				x	x
17	Consultas				x	
18	Consultas y recuperatorio				x	x
19	Integrador					x

Firmado digitalmente por: SUAREZ Mariana Alejandra
Directora del Departamento de Ciencia y Tecnología
UNIVERSIDAD NACIONAL DE QUILMES