Instituto de
Tecnología e Ingeniería

Fecha de aprobación del programa: 30/10/2018
Vencimiento: 30/10/2020

Programación funcional

Carrera: Licenciatura en Informática

Asignatura: Programación Funcional

Área: Algoritmos y lenguajes

Prerrequisitos: Estructuras de Datos

Carga Horaria:

- Carga horaria total 64 horas
- Carga horaria práctica: 48 hs
  - Formación Experimental: 16 hs
  - Resolución de problemas: 32 hs
- Carga horaria semanal: 4 horas por semana

Objetivos:

Se espera que el estudiante:

- Adquiera y comprenda conceptos fundamentales de la programación funcional y su importancia en la tarea de programar. Por ejemplo: abstracción mediante funciones y sistemas de tipos.
- Sea capaz de aplicar técnicas de transformación de programas en casos particulares.
- Pueda implementar programas sencillos en un lenguaje funcional. Y pueda demostrar propiedades sencillas de programas funcionales utilizando inducción estructural.

Contenidos mínimos:

Nociones generales del paradigma funcional. Valores y expresiones. Las funciones como valores. Sintaxis. Sistema de Tipos Hindley-Milner. Tipos básicos. Constructores de tipos. Polimorfismo. Funciones de alto orden. Currificación. Inducción/Recursión. Definición inductiva de conjuntos. Definición recursiva de funciones sobre esos conjuntos. Demostraciones inductivas. Inducción estructural. Listas como tipo inductivo. Funciones básicas y de alto orden sobre listas. Patrón de recorrido, selección y recursión. Sistemas de Tipos. Ventajas y limitaciones de los lenguajes de programación con tipos. Asignación de tipos a expresiones. Algoritmo de inferencia. Tipos de datos recursivos. Transformación de Programas. Obtención de programas a partir de especificaciones.

Instituto de
Tecnología e Ingeniería

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:

- Bird Richard, Wadler Philip, Introduction to Functional Programming. First Edition, Prentice Hall, Oxford, 1988.
- Bird Richard, Introduction to Functional Programming using Haskell. Second 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/

Bibliografía de consulta:

- Miran Lipovača , Learn you a Haskell for great good. A Beginner's Guide. ISBN: 978-1-59327-283-8. URL: http://learnyouahaskell.com
- Simon Thompson, The Craft of Functional Programming. Addison Wesley, 1996.
- 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 se organizan como teórico-prácticas. Durante las clases se presenta el contenido formal y se ahonda en las características de la programación funcional.

Después de completado cada tema se resuelven ejercicios de programación haciendo hincapié en las características de la programación funcional mediante las computadoras del laboratorio.

En la práctica se plantean ejercicios que pueden ser realizados como trabajo experimental usando lenguajes funcionales (Haskell).

Uso del campus virtual.

El Campus Virtual es un espacio fundamental para el desarrollo de la asignatura. En el aula virtual se propondrá material educativo, apuntes de clase, bibliografía así como también el programa y cronograma de la asignatura y las guías de Trabajos Prácticos y ejercicios.

Modalidad de evaluación:

Consistirá en dos exámenes parciales con recuperatorios, según el cronograma previsto, de la totalidad de la materia descripta en el programa. Los mismos se realizarán en las fechas que establezcan en el cronograma correspondiente.

- La calificación de cada evaluación se determinará en la escala 0 a10, con los siguientes valores: 0, 1, 2 y 3: insuficientes; 4 y 5 regular; 6 y 7 bueno; 8 y 9 distinguido; 10 sobresaliente. La materia podrá aprobarse mediante: régimen de promoción directa, exámenes finales regulares y exámenes libres.
- Régimen de promoción directa (sin examen final): los/las estudiantes deberán aprobar las materias con siete (7) o más puntos de promedio entre todas las instancias evaluativas, sean éstas parciales o sus recuperatorios, debiendo tener una nota igual o mayor a seis (6) puntos en cada una de éstas. Todas las instancias evaluativas tendrán una posibilidad de recuperación. En el caso

de los ausentes en la fecha original, el recuperatorio operará como única fecha de examen. El examen recuperatorio permite mantener la chance de la promoción siempre y cuando respete las condiciones de calificación respectiva.

- Exámenes finales regulares: para aquellos/as estudiantes que hayan obtenido una calificación de al menos de 4 (cuatro) y no se encuentren en las condiciones de promoción, deberán rendir un examen final que se aprobará con una nota no inferior a 4 (cuatro) puntos.

La asistencia no debe ser inferior al 75% en las clases presenciales.