UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Departamento Ingeniería en Sistemas de Información

ASIGNATURA:	PARADIGMAS DE PROGRAMACION
DEPARTAMENTO:	ING. EN SIST. DE INFORMACIÓN
AREA:	PROGRAMACIÓN
BLOQUE	TECNOLOGÍAS BÁSICAS

MODALIDAD:	Cuatrimestral
HORAS SEM.:	8 horas
HORAS/AÑO:	128 horas
HORAS RELOJ	96
NIVEL:	2º
AÑO DE DICTADO:	Plan 2008

Objetivos

- Comprender los fundamentos de los paradigmas básicos que son utilizados en los lenguajes de programación.
- Conocer el modelo formal o semiformal subyacente de cada paradigma y la forma en que el mismo es incorporado en un lenguaje de programación correcto.
- Aplicar los diferentes paradigmas en la resolución de problemas.

Contenidos Mínimos (Programa Sintético).

- Concepto de Paradigmas de Programación.
- Paradigmas Fundamentales.
- Paradigma Funcional.
- Cálculo Lambda.
- Lenguajes de Programación Funcional.
- Paradigma Lógico.
- Lógica de Predicados de Primer Orden y Formas Restringidas.
- Regla Inferencia de Resolución.
- Lenguaje de Programación Lógica.
- Paradigma Orientado a Objetos.
- Conceptos Básicos.
- Clasificación, Clase y Objeto.
- Método y Mensaje.
- Clase Abstracta y Concreta.

- Herencia y Tipos de Herencia.
- Polimorfismo y Tipos de Polimorfismo en el Modelo de Objetos.
- Lenguajes de Programación Orientado a Objetos.
- Extensiones al Modelo Básico de Objeto en un Lenguaje Particular

Contenido Analítico:

UNIDAD 1: Paradigmas de Programación
Concepto de paradigma de programación. Necesidad de la existencia de diferentes paradigmas de programación. Concepto de programa: definiciones generales y específicas. Diferencia entre lenguaje y paradigma de programación. Concepto de tipo: representación de los tipos en los diferentes lenguajes de programación. Importancia del concepto de tipo en relación a la implementación de sistemas complejos y cambiantes. Comparación de los diferentes esquemas de chequeo de tipos. Ubicación de los mecanismos de control de flujo en un programa. Declaratividad: importancia de la separación del control de flujo de la lógica del dominio a modelar. Abstracción y modularización: definición y mecanismos de implementación. Orden superior: concepto e implicancias en el desarrollo de programas. Utilización de las variantes del polimorfismo en los diferentes paradigmas. Comparación entre los diferentes paradigmas de programación.

UNIDAD 2: Paradigma de Objetos
Concepto de Objeto. Concepto de mensaje, estado y comportamiento. Encapsulamiento. Visión de programa entendido como un conjunto de objetos que envían mensajes. Ambientes de objetos: diferencia con la programación tradicional. Los métodos como mecanismo de resolución de mensajes. Concepto de polimorfismo. Concepto de Clase como modelo/molde de objetos. Delegación y responsabilidad. Concepto de referencia. Interfaz e implementación: encapsulamiento del estado interno, ocultamiento de datos. Tipos de mensaje. Herencia. Variables y métodos de clase. Igualdad e identidad. Relaciones entre clases: asociación, composición; relación con delegación. Aplicación del concepto de tipo en el paradigma de objetos. Efecto de lado y declaratividad en el paradigma de objetos. Concepto de orden superior en la programación orientada a objetos. Introducción al manejo de errores.
Lenguaje asociado: Smalltalk. Imagen, ambiente de objetos, definición y uso de clases y objetos. Herramientas de navegación (object browser, class browser, otros). Uso de workspaces. Estudio de algunas clases propias de Smalltalk: String, Integer, Date, otras. Estudio del protocolo de Colecciones. Bloques. Garbage collection.

UNIDAD 3: Paradigma Funcional
Concepto de función. La función como bloque de construcción de programas. Concepto de programa en el paradigma funcional. Efecto de lado. Concepto de

variable. Definición de tipo y valor. Definición de funciones. Funciones definidas por ramas. Pattern matching. Inferencia de tipos. Funciones recursivas. Prueba por inducción. Manejo de listas. Listas por comprensión. Funciones de orden superior. Currificación y aplicación parcial de funciones. Evaluación diferida y listas infinitas. Composición de funciones. Sistemas de tipos. Polimorfismo y tipos genéricos. Tuplas. Expresiones lambda.
Lenguaje asociado: Haskell. Entorno de trabajo, definición de programas, uso del intérprete. Notación bidimensional. Módulos. Notación de listas [n..m]. Notación de listas por comprensión. Operadores infijos y prefijos. Reglas de precedencia. Prelude de Haskell. Funciones incorporadas en el prelude para manejo de listas, de tuplas, de funciones de orden superior.

UNIDAD 4: Paradigma Lógico
Fundamentación lógica. Predicados. Razonamientos y silogismos. Relaciones, hechos y reglas. Consultas. Tipos de consultas. Definición de programa en Paradigma Lógico. Motor de inferencia, ubicación del control en un programa lógico. Diferencia entre una función y una relación. Concepto de variable o incógnita. Unificación. Múltiples resultados. Inversibilidad. Aritmética, evaluación de expresiones aritméticas. Negación. Listas. Pattern Matching. Predicados de orden superior. Functores. Polimorfismo.
Lenguaje asociado: Prolog. Entorno de trabajo, manejo de archivos. Realización de consultas. Ayuda. Trace y debug. Limitaciones de inversiblidad: generación de valores.

Bibliografía.

- *Programming Languages Concepts and Paradigms*, David Watt, Prentice Hall. 1990.
- *Concepts, Techniques, and Models of Computer Programming*, Peter Van Roy and Seif Haridi, The MIT Press. 2003.
- *Designing Object-Oriented Software*, Wirfs- Brock, Brian Wilkerson y Lauren Wiener, Prentice Hall. 1990.
- *Smalltalk, Objects and Design*, Chamond Liu., Prentice Hall., 2000.
- *Smalltalk Best Practice Patterns,* Kent Beck. Prentice Hall. 1995.
- *Smalltalk 80- The Language ,* Adele Goldberg and David Robson. Addison Wesley. 1989.

- Introduction to Functional Programming, Richard A. Bird y Philip Wadler. Prentice Hall. 1998.
- Introducción al lenguaje Haskell, José E. Labra G., Universidad de Oviedo. 1998.
- *Prolog*, Giannesini, Kanoui, Pasero y Van Caneghem, Addison, Wesley Iberoamericana. 1989.
- *Logic Programming and Knowledge Representation,* Chitta Baral and Michael Gelfond, University of Texas, Logic Programming and Knowledge Representation. 2002.
- *The Arity/ Prolog Language Reference Manual*, Arity Corporation. 1989.
- Los siguientes tutoriales, disponibles en el sitio www.haskell.org
- Real World Haskell , Bryan O'Sullivan, Don Stewart y John Goerzen, O'Reilly Media
- Learn You a Haskell for Great Good! , Miran Lipovača
- Yet Another Haskell Tutorial, Hal Daumé III
- A Gentle Introduction to Haskell , Paul Hudak, John Peterson and Joseph H.Fasel

Correlativas

Para Cursar:

Cursadas:

- Matemática Discreta
- Algoritmos y Estructuras de Datos.

Para rendir:

Aprobadas:

- Matemática Discreta
- Algoritmos y Estructuras de Datos.