

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

MODALIDAD:	Cuatrimestral
HORAS SEM.:	7 horas
HORAS/AÑO:	112 horas
HORAS RELOJ	84
NIVEL:	2º Año
AÑO DE DICTADO:	Plan 95

Objetivos

- Estudiar los diferentes paradigmas de programación.
- Laboratorio asociado: Resolución de problemas en variados lenguajes de programación.

Contenidos Mínimos (Programa Sintético).

- Paradigma Funcional.
- Paradigma Lógico.
- Paradigma Orientado a Objetos.

Contenidos Analíticos:

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 inversibilidad: 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:

- Sintaxis y Semántica del Lenguaje

Aprobadas

- Algoritmos y Estructuras de Datos.

Para rendir:

Aprobadas:

- Sintaxis y Semántica del Lenguaje