PARADIGMAS DE PROGRAMACIÓN

PROGRAMA ANALÍTICO

PLAN DE ESTUDIOS	**2008**
ORDENANZA CSU. Nº	**1150**
HORAS/AÑO:	**128**
OBLIGATORIA	**X**
ELECTIVA	
ANUAL	
PRIMER CUATRIMESTRE	
SEGUNDO CUATRIMESTRE	**X**
NIVEL / AÑO	**2º**
HORAS CÁTEDRA SEMANALES	**8**

OBJETIVOS

OBJETIVO GENERAL

Desde la ingeniería en Sistemas de Información, los lenguajes pueden ser considerados como herramientas para formular, representar y resolver problemas. Bajo esta perspectiva el paradigma que sustenta un lenguaje y la programación en el mismo, estaría dictando una forma de modelar o ver un problema, por lo tanto el análisis de los diferentes paradigmas, además de revelar las características de estos, está brindando un panorama de los distintos enfoques con que se puede representar un problema en términos de programación.

En esta asignatura se analizan cuatro paradigmas que presentan de alguna manera enfoques totalmente diferentes de formular la programación, como lo son los paradigmas: imperativo, orientado a objetos, lógico y funcional. Si bien el paradigma imperativo no se trata en detalle, ya que el mismo se analizó en las asignaturas previas, se lo utiliza en el desarrollo de Abstracción de Datos y en comparaciones permanentes con otros paradigmas.

Es importante que el alumno adquiera los elementos que le permitan analizar los conocimientos tecnológicos que se desarrollan en los últimos tramos de la carrera, profundizar el estudio de lenguajes, y además brindarle los elementos que lo apoyen en su actividad profesional, en la cual tendrá: una fuerte interacción con lenguajes de programación, ya sea por su participación en la implementación de sistemas, como por la selección de ambientes de desarrollo y tecnología, para lo cual son sumamente importantes los contenidos estudiados en esta asignatura. Por lo tanto, para el dictado de la asignatura en la carrera Ingeniería en Sistemas de Información, se propone la presentación de nociones de los aspectos formales de cada paradigma, sin alcanzar el grado de profundidad

que debería desarrollarse en una carrera en Ciencias de la Computación; no obstante se revisan sintéticamente los fundamentos de los mismos ya que dan la justificación conceptual de su existencia.

OBJETIVOS ESPECÍFICOS

OBJETIVOS DE LA UT I. PARADIGMAS DE PROGRAMACIÓN BÁSICOS

Introducir al alumno los conceptos fundamentales de los paradigmas de programación básicos.

OBJETIVOS DE LA UT II. PARADIGMA DE PROGRAMACIÓN IMPERATIVO

Incorporar el concepto de abstracción, reforzar el concepto de modularización a través del estudio de los Tipos Abstractos de datos en el Paradigma Procedural.

OBJETIVOS DE LA UT III. PARADIGMA DE PROGRAMACIÓN ORIENTADO A OBJETOS

Introducir los conocimientos necesarios que permitan al alumno diseñar soluciones eficientes haciendo uso de mecanismos específicos del Paradigma Orientado a Objetos. Incorporar nociones básicas de análisis y diseño de sistemas Orientados a Objetos utilizando UML (Lenguaje unificado de Modelado), brindar los conocimientos necesarios del lenguaje Smalltalk y de su ambiente de trabajo mediante el cual los alumnos implementarán las soluciones obtenidas en la Programación Orientada a Objetos.

OBJETIVOS DE LA UT IV. PARADIGMA DE PROGRAMACIÓN FUNCIONAL

Introducir al alumno en el estudio de formas funcionales y reglas de computación para expresar soluciones en el Paradigma Funcional.

OBJETIVOS DE LA UT V. PARADIGMA DE PROGRAMACIÓN LÓGICO

Incorporar conceptos básicos de la Programación Lógica que permitan conocer otra metodología de resolución de problemas a través de reglas de inferencia y la introducción del lenguaje Prolog como lenguaje específico de este paradigma.

OBJETIVOS DE LA UT VI. COMPARACIÓN ENTRE PARADIGMAS

Realizar un paralelismo entre los distintos paradigmas de manera que el alumno pueda visualizar ventajas y desventajas, y posibilidades de programación que brinda cada uno de ellos.

CONTENIDOS

CONTENIDOS SINTÉTICOS

- 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 Orientados a Objetos.
- Extensiones al Modelo Básico de Objeto en un Lenguaje Particular.

CONTENIDOS ANALÍTICOS

UNIDAD TEMÁTICA I. PARADIGMAS DE PROGRAMACIÓN BÁSICOS

- Presentación de los cuatro paradigmas de la materia: Imperativo, Orientado a objetos, Funcional y Lógico.
- Ventajas y desventajas de cada uno de ellos.

TIEMPO ESTIMADO: 12 HORAS

UNIDAD TEMÁTICA II. PARADIGMA DE PROGRAMACIÓN IMPERATIVO

- Características del paradigma imperativo. Concepto de variable. Tipo de datos.
- Abstracción de datos. Ocultamiento de la información.
    - Tipos abstractos de datos simples (TAD)
    - Tipos abstractos de datos compuestos (TAD).
    - Resolución de problemas aplicando técnicas de descomposición (diseño descendente) en el manejo de los datos: divide y vencerás.
    - Tipos abstractos de datos (TAD) en ADA. Tipos de datos abstractos genéricos. Ejemplos.

TIEMPO ESTIMADO: 20 HORAS

UNIDAD TEMÁTICA III. PARADIGMA DE PROGRAMACIÓN ORIENTADO A OBJETOS

- Introducción al Paradigma Orientado a Objetos. Conceptos fundamentales. Abstracción de datos y ocultamiento de la información. Estructura de un objeto. Métodos y mensajes. Clasificación de mensajes. Clase.
- Concepto de generalización-especialización. Herencia, simple y múltiple. Polimorfismo y binding dinámico.
- Introducción al Análisis y Diseño Orientado a Objetos utilizando como notación el lenguaje gráfico UML (Lenguaje Unificado de Modelado). Diagramas Estáticos de UML (de clase) para diseñar la Estructura del Sistema. Diagramas de Secuencia UML para representar la interacción entre objetos del Sistema en construcción.
- Introducción a la sintaxis de Smalltalk. Expresiones de asignación y variables. Asignación. Tipos de variables. Expresiones de mensaje. Sintaxis de un mensaje. Tipos de mensajes: unario, binario y palabra clave. Orden de precedencia en la evaluación de expresiones. Expresión de mensajes en cascada.
- Implementación de estructuras de control en Smalltalk: condicional simple y compuesto, iteración y repetición.
- Jerarquía de clases. Especialización y generalización. Definición de una clase. Métodos de clase e instancia. Sintaxis de la definición de un método. Significado de la expresión de retorno. Empleo de self y super.
- Desarrollo de un ejemplo de aplicación empleando los conceptos desarrollados. Análisis de clases de Smalltalk: Magnitude, OrderedCollection, etc.

TIEMPO ESTIMADO: 48 HORAS

UNIDAD TEMÁTICA IV. PARADIGMA DE PROGRAMACIÓN FUNCIONAL

- Introducción al Paradigma Funcional. Desarrollo del concepto de función matemática y su empleo en un lenguaje de programación. Transparencia referencial.
- Reglas de computación. Estrategias de reducción. Formal Normal. Grafo de reducción. Formas funcionales. Composición. Introducción a LISP.
- Técnicas de especificación y diseño de problemas basados en el uso del mecanismo de "recursión" y de "backtraking"
- Expresiones de orden superior: funciones como argumento y resultado de la evaluación de funciones.

- Resolución de ejemplos de aplicación empleando el enfoque funcional.

TIEMPO ESTIMADO: 16 HORAS

UNIDAD TEMÁTICA V. PARADIGMA DE PROGRAMACIÓN LÓGICO

- Introducción a la Programación Lógica. Formas clausales. Proposiciones. Cláusulas. Cláusulas de Horn. Introducción al cálculo de predicados y términos (constante simbólica, variable y función). Predicados de primer orden. Introducción a Prolog.
- Interpretación lógica. Conceptos de: teoría, consecuencia lógica y programa. Regla de inferencia: RESOLUCION.
- Unificación. Unificador más general. Evaluación de programas en lógica. Desarrollo de árboles de derivación.
- Especificación y diseño de problemas aplicando mecanismos de "recursión" y "backtraking".
- Interpretación algorítmica: Procedimientos y programación. Intérprete no determinista. Estrategia de evaluación.

TIEMPO ESTIMADO: 16 HORAS

UNIDAD TEMÁTICA VI. COMPARACIÓN ENTRE PARADIGMAS

- Puntos de comparación. Diferencias y similitudes entre un TAD y una clase. Overloading de Ada vs. Polimorfismo de Smalltalk. Estructuras de control de Ada vs. estructuras de control de Smalltalk. Funciones matemáticas vs. funciones procedurales" o "imperativas".

TIEMPO ESTIMADO: 8 HORAS

EXÁMENES: 8 HORAS

BIBLIOGRAFÍA OBLIGATORIA

TITULO	AUTOR	EDITORIAL	AÑO DE EDICIÓN/ ISBN	EJEMP. DISP.
Descubra SmallTalk	Lalonde, Wilf	Addison-Wesley Iberoamericana	2004	1
El lenguaje unificado de modelado	Booch Grady, Jacobson Ivar, Rumbaugh J.	Addison-Wesley Iberoamericana	2006/ 2007	4 (2006) 4 (2007)
Introducción a la programación Orientada a Objetos	Timothy Budd	Addison Wesley	2002	3
Construcción de Software Orientado a Objetos.	Bertrand Meyer.	Prentice Hall	2000	-
Programación Lógica: Teoría y Práctica	Pascual Julián Iranzo	Pearson Education	2007	-
An Introduction to Functional Programming	Richard Bird and Philip Wadler	Prentice-Hall	1998	-

BIBLIOGRAFÍA COMPLEMENTARIA

U.T.N. F.R.L.P. FOLIO Nº 78

TITULO	AUTOR	EDITORIAL	AÑO DE EDICIÓN /ISBN	EJEMP. DISP.
Análisis y diseño orientado a objetos con aplicaciones.	Booch Grady	Addison-Wesley Iberoamericana	2007	-
Smalltalk With Style	Suzanne Skublics, Edward J. Klimas, David A. Thomas, John Pugh	Pearson Education	2001	-
Comparative Programming Languages	Wilson, Leslie; Clark, Robert.	Addison Wesley	2001	-

CARACTERÍSTICAS DE LA ACTIVIDAD CURRICULAR

DESCRIPCIÓN

A continuación se describen brevemente las actividades curriculares, las tareas a realizar por los docentes y alumnos, como así también los materiales didácticos que se requieran para desarrollarla.

- ✓ Las clases teóricas incluirán gran contenido de práctica, serán participativas y de debate. Se desarrollarán de manera expositiva los conceptos fundamentales, con presentación de casos de estudio prácticos de aplicación inmediata del tema presentado. Se promoverá la investigación y la búsqueda de información. Se exigirá un trabajo continuo al alumno en la actividad áulica y fuera de ella.
- ✓ Las clases prácticas incluirán presentación del tema, casos de estudio abiertos a debate y de construcción colectiva y consulta individual de ejercicios.
- ✓ Los trabajos prácticos contendrán una lista de ejercicios a resolver. A medida que avance el tratamiento de las unidades temáticas, un trabajo práctico incluirá los conceptos aprendidos en los anteriores de modo que desde la presentación hasta el contenido serán cada vez más completos a medida que se avance en la cursada. Paralelamente se refuerzan los conocimientos por aplicación de la noción de repaso permanente de los conceptos adquiridos.
- ✓ Los alumnos deben asistir y participar en las clases teóricas, a las prácticas y de laboratorio. Deben aprobar el parcial en primera instancia o en opciones recuperatorias. Deben aprobar el laboratorio – taller de programación en Smalltalk.
- ✓ Las comisiones de trabajos en el laboratorio serán de dos/tres personas como número considerado ideal para este tipo de talleres.

MODALIDAD DE LA ENSEÑANZA

Por tratarse de una materia con un importante respaldo conceptual, la enseñanza de la teoría ocupa un lugar destacado, y para facilitar el entendimiento de cada uno de los temas se recurre a ejemplos en forma continua. Así, la enseñanza de la materia se apoya en una estrecha coordinación entre el dictado de los conceptos teóricos, presentación de ejemplos y la aplicación de la teoría en la resolución de ejercicios. Estos ejercicios están orientados al análisis o entendimiento de las soluciones propuestas y a la concepción o diseño para responder a nuevos requerimientos.

DIRECCIÓN ACADÉMICA
ES COPIA FIEL DEL ORIGINAL
U.T.N. F.R.L.P.

Estrategias metodológicas

Las 8 horas semanales que tiene asignadas se distribuyen de la siguiente manera:

UNIVERSIDAD TECNOLÓGICA NACIONAL FACULTAD REGIONAL LA PLATA

MARÍA EUGENIA BORATTO
DIRECTORA
DIRECCIÓN ACADÉMICA
U.T.N. F.R.L.P.

- ✓ *Clases teóricas:* 3 horas para presentación y exposición de los temas, con el uso de pizarrón y apoyo de proyección de imágenes, son desarrolladas por el profesor a cargo del curso. Desarrollo de casos de estudio.
- ✓ *Clases prácticas:* 3 horas de ejercitación para resolución de problemas, trabajando en el aula, a cargo de un auxiliar.
- ✓ *Clases de laboratorio- taller:* 2 horas semanales de ejercitación sobre máquina para aplicación de un lenguaje de programación como herramienta instrumental de resolución de los problemas vistos en la práctica.
- ✓ El trabajo en taller involucra la realización de un Trabajo Práctico Grupal (no más de 3 alumnos) en máquina. En el que se resolverán problemas para la ejercitación y uso de Tads en Pascal, y del paradigma orientado a objetos se utilizará como lenguaje de programación Smalltalk/V.

El software necesario para implementar los programas es el Smalltalk Express y se encuentra disponible en el Laboratorio o en el sitio de la cátedra desde la página de la facultad a través de la url: www.frlp.utn.edu.ar/materias/paradigmas para su descarga.

Estrategias de enseñanza

- ✓ En las clases teóricas de presentación, se imparte en profundidad el contenido de la materia, explicando, aplicando y evaluando conceptos y terminologías de la sintaxis y la semántica de los lenguajes de programación. Además se desarrollan estudios de casos y se promueven debates sobre problemas y alternativas de soluciones. Además se incluyen actividades de análisis y discusión sobre las actividades de revisión propuestas a los alumnos para realizar fuera de clases.
- ✓ En las clases prácticas se completa el estudio de casos y se realiza la resolución de ejercicios sobre guías especialmente diseñadas, con nivel de complejidad creciente, disponibles anticipadamente en la semana. En estas clases también se realizaran las presentaciones de los Trabajos Prácticos.
- ✓ Las clases en laboratorio funcionan como talleres, en donde se presentan las herramientas a emplear y se orienta para su instalación y aplicación.

EVALUACIÓN

La cursada se regulariza mediante la aprobación de un Trabajo Práctico Grupal (máximo 3 integrantes), desarrollado en máquina en el Taller de Programación en Smalltalk y de 1 (un) parcial conceptual práctico.

- ✓ El Trabajo Grupal tiene 2 (dos) fechas de entrega o evaluación. Es evaluado en máquina, en papel y mediante un Coloquio individual sobre su desarrollo en taller. Es condicionante para la regularidad en la materia y deberá ser aprobado antes de la instancia de parcial.
- ✓ El parcial tendrá 3 (tres) fechas de recuperación y es de carácter práctico global incluyendo todos los temas desarrollados en la práctica.

La materia se aprueba mediante un examen final escrito, de carácter teórico – práctico, que constituye una evaluación sumativa final.