INSTITUTO DE DESARROLLO ECONÓMICO E INNOVACIÓN
Año: 2017

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur.

PROGRAMA DE LA ASIGNATURA:
Paradigmas y Lenguajes de Programación (IF020)

CÓDIGO: IF020
AÑO DE UBICACIÓN EN EL PLAN DE ESTUDIOS: 4 año
FECHA ULTIMA REVISIÓN DE LA ASIGNATURA: 2017-03-31
CARRERA/S: Licenciatura en Sistemas V2

CARÁCTER: CUATRIMESTRAL (1ro)
TIPO: OBLIGATORIA
NIVEL: GRADO
MODALIDAD DEL DICTADO: PRESENCIAL
MODALIDAD PROMOCION DIRECTA: SI
CARGA HORARIA SEMANAL: 8.00 HS
CARGA HORARIA TOTAL: 120.00 HS

EQUIPO DOCENTE

Nombre y Apellido	Cargo	e-mail
Adriana Urciuolo	Profesora Titular	aurciuolo@untdf.edu.ar
Matías Gel	Asistente Principal	mgel@untdf.edu.ar

1. FUNDAMENTACION

El estudio de conceptos de lenguajes y la realización de las actividades propuestas en la asignatura contribuyen al desarrollo de capacidades cognitivas superiores del alumno, como son la capacidad de analizar y resolver problemas, de razonar de manera crítica y tomar decisiones en el contexto del proceso de desarrollo de software, y de aplicar los conocimientos a la práctica.
En el marco del Plan de Estudios, la materia contribuye a la comprensión de los distintos modelos de computación y paradigmas de programación y al conocimiento y análisis de la estructura de los lenguajes de programación, sus construcciones y descripción. De esta manera, el alumno va adquiriendo la capacidad de evaluar y comparar lenguajes, lo cual enriquece los conceptos anteriores adquiridos en materia de desarrollo de software, desde un aspecto no contemplado hasta esta instancia de la carrera.
Con respecto a la ubicación de la materia dentro del Plan de estudios, cabe destacar que se trata de una asignatura del cuarto año de la carrera; se debe tener en cuenta por lo tanto, que el alumno ya tiene incorporados conceptos de programación, análisis y diseño, sistemas de información y lenguajes, que constituirán la base de trabajo. En esta instancia se formalizan los conceptos sobre lenguajes adquiridos previamente en materias de Programación y de introducción a la informática: En las asignaturas Elementos de Informática y Algorítmica y Programación el alumno ha adquirido los conceptos introductorios relativos a: niveles y evolución de los lenguajes de programación, programación procedural; en las asignaturas Algorítmica y Programación II y Programación y Diseño Orientado a Objetos, los relativos a distintos paradigmas de programación, lenguajes PASCAL y JAVA. Sobre esta base teórica se avanza en el conocimiento de los constructores y mecanismos que proveen los distintos lenguajes de programación en el marco de diferentes paradigmas.

2. OBJETIVOS

a) OBJETIVOS GENERALES

Se pretende que el alumno adquiera la capacidad de realizar una evaluación crítica de los lenguajes de programación existentes y futuros desde distintos puntos de vista, y cuente con los fundamentos para seleccionar los lenguajes apropiados a cada tipo de aplicación.

b) OBJETIVOS ESPECIFICOS

mediante el dictado de la materia se pretende que el alumno sea capaz de:
- Adquirir los conceptos fundamentales sobre los cuales se construyen los diferentes lenguajes de programación.
- Evaluar y comparar los lenguajes de programación en términos de su desempeño para determinados propósitos y áreas de aplicación.
- Desarrollar los criterios necesarios para realizar dicha evaluación.
- Seleccionar el lenguaje de programación más apropiado para el tipo de aplicación a desarrollar.
- Incrementar su capacidad para pensar aproximaciones distintas a problemas reales.
- Utilizar en forma más eficiente los Lenguajes de programación
- Comprender el rol de los lenguajes de programación en el proceso de desarrollo de software.
- Analizar comparativamente distintos paradigmas de lenguajes de programación.
- Adquirir los conceptos respecto de los mecanismos y herramientas con que cuentan los lenguajes para implementar las características de cada paradigma.

3. CONDICIONES DE REGULARIDAD Y APROBACION DE LA ASIGNATURA

Para la aprobación del cursado de la asignatura se requiere:
a) Asistencia al 70% de las clases prácticas.
b) Aprobación de los trabajos prácticos obligatorios.
b) Aprobación de dos parciales prácticos con un mínimo de cuatro (4) puntos o sus respectivas instancias recuperatorias.
d) Presentación de dos trabajos integradores sobre: evaluación de Lenguajes de Programación y Paradigmas de Programación.
La calificación final de la cursada es el promedio de las notas obtenidas en los exámenes parciales, los trabajos de lenguajes y paradigmas y los trabajos prácticos. En ningún caso la nota obtenida podrá ser inferior a 6.
Para la aprobación de la asignatura se deberá rendiir examen final con Nota igual a cuatro (4) o superior.

Para la Promoción de la asignatura, además de las condiciones básicas de cursada, se requiere:
a) Aprobación de los dos parciales prácticos en la primera instancia, con Nota igual o superior a siete (7)
b) Aprobación de dos parciales teóricos con Nota igual o superior a siete (7)
No se rinde examan final

4. CONTENIDOS DE LA ASIGNATURA

Contenidos mínimos

• Sintaxis y semántica. Nociones básicas de semántica formal. Semántica operacional.

• Lenguajes de programación: entidades y ligaduras.
• Sistemas de tipos.
• Niveles de polimorfismo.
• Encapsulamiento y abstracción.
• Conceptos de intérpretes y compiladores.
• Criterios de diseño y de implementación de lenguajes de programación.
• Paradigmas de programación: imperativo, orientado a objetos, funcional, lógico.
• Concurrencia y paralelismo.

Programa Analítico

MÓDULO I: Conceptos introductorios
Desarrollo de software y lenguajes de programación. Razones para el estudio de lenguajes de programación. Síntesis de los paradigmas de programación. Paradigma imperativo. Conceptos fundamentales. Paradigma Declarativo. Conceptos fundamentales. Aportes de cada paradigma. Dominios de aplicación. Lenguajes característicos de los diferentes dominios. Criterios para el estudio, análisis, proyecto y evaluación de lenguajes. Importancia del estudio de conceptos comparados en lenguajes. Evolución de los lenguajes de programación. Perspectiva histórica.

MÓDULO II: Sintaxis y semántica de Lenguajes de Programación
Sintaxis. Criterios generales. Elementos sintácticos de un Lenguaje. Descripción de la sintaxis de un LP. Definición formal de la sintaxis de un lenguaje. Gramática formal: Gramática BNF. BNF extendido. Árboles sintácticos. Diagramas de Sintaxis. Sintaxis abstracta y concreta. Ambigüedad. Semántica. Semántica estática. Gramática de atributos. Semántica dinámica. Semántica operacional. Introducción a la semántica formal de lenguajes: Semántica axiomática y denotativa. Procesadores de Lenguajes. Métodos de interpretación y compilación. Etapas de un proceso de traducción. Análisis del Programa fuente: Análisis léxico, sintáctico y semántico.

MÓDULO III: Variables
Variables. Atributos: Nombres y ámbito de nombres. Valor izquierdo. Valor derecho. Tiempo de vida. Tipos. Ámbito. Concepto de Binding. Tiempo de vinculación. Binding de tipos: Binding estático y dinámico de tipos. Binding de almacenamiento y tiempo de vida. Variables estáticas. Variables dinámicas de pila. Variables dinámicas de heap. Chequeo de tipos. Tipado fuerte. Análisis de lenguajes fuertemente tipados. Compatibilidad de tipos. Compatibilidad nominal y estructural. Subtipo. Tipo derivado. Ámbito. Ámbito estático y dinámico. Evaluación de ámbito estático y dinámico. Entorno de referencia. Inicialización de variables. Variables no inicializadas. Asignación estática y dinámica de memoria.

MÓDULO IV: Tipos de Datos
Tipos de datos. Tipos built-in y primitivos. Tipos ordinales definidos por el usuario. Tipos agregación y su implementación: Producto cartesiano. Uniones y uniones discriminadas. Mapeos Finitos. Arrays. Categorías de arrays: Estáticos, de pila dinámica, dinámicos de heap. Tipos Secuencia. Decisiones de diseño en Strings. Conjunto Potencia. Tipos Recursivos. Tipo Puntero. Inseguridad de los punteros. Punteros colgantes. Objetos perdidos. Estrategias de recolección de basura. Sistemas de tipos. Tipos monomórficos vs. Tipos polimórficos. Tipos de datos abstractos. Lenguajes que soportan TAD. Tipos abstractos parametrizados. Estructura de tipos de algunos lenguajes: Pascal, C++, Java, Ada. Lenguajes con sistemas fuertemente tipados. Constructores y destructores. Gestión de almacenamiento.

MÓDULO V: Estructuración de cómputos: Expresiones y estructuras de control
Expresiones. Expresiones aritméticas, relacionales y booleanas. Reglas de Precedencia. Asociatividad. Paréntesis. Sentencias de asignación. Asignación como expresión. Asignación en

modo mixto. Evaluación corto-circuito vs, Evaluación estricta. Sobrecarga de operadores. Conversiones de tipo. Coerciones. Conversión de tipo explícita. Estructuras de control. Enunciados compuestos. Enunciados de selección. Selectores anidados. Selección Múltiple. Enunciados iterativos. Ejecución condicional. Mecanismos de control de bucle. Salto incondicional. Enunciados goto, break, continue. Ejemplos de expresiones y estructuras de control en ADA, C, Pascal y FORTRAN.

MÓDULO VI: Manejo de Excepciones
Estructuras de control de Nivel Unidades. Gestión de excepciones. Eventos excepcionales. Programa tolerante a fallos. Lanzamiento de una excepción. Manejador de excepciones. Propagación de una excepción. Formas de manejar excepciones. Facilidades de lenguajes para el manejo de excepciones. Manejo de excepciones en PL/I, Java, C++. Comparaciones. Computación manejada por eventos

MÓDULO VII: Estructuración de programas
El concepto de abstracción procedural. Fundamentos de Subprogramas. Parámetros. Procedimientos y Funciones. Métodos de paso de parámetros. Modos de implementación. Cuestiones de Diseño. Métodos utilizados por los distintos lenguajes. Subprogramas como parámetros. Tipos de valores de retorno. Subprogramas sobrecargados. Subprogramas polimórficos. Implementación de subprogramas. Soporte de modularidad. Encapsulación. Interface e Implementación. Separación de Interface e Implementación. Compilación separada e independiente. Librerías de módulos. Características de diferentes lenguajes para la organización de programas: Cómo se provee encapsulación, interface e implementación, organización general del programa. Facilidades de compilación. Unidades genéricas. Estructuras de datos genéricas. Algoritmos genéricos. Módulos genéricos. Genericidad en ADA y C++.

MÓDULO VIII: Concurrencia
Concurrencia. Conceptos fundamentales de concurrencia a nivel de subprogramas. Niveles de concurrencia. Threads. Sincronización de cooperación y competencia. Comunicación entre procesos. Corrutinas. Corrutinas en Modula-2. Cuestiones de diseño para LP con facilidad de concurrencia. Procesos. Tareas. Modelos de implementación. Mecanismos de Sincronización y comunicación: Semáforos. Monitores y señales. Rendezvous. Ejemplos en diferentes lenguajes.

MÓDULO IX: Paradigma de Programación orientada a objetos
Conceptos relevantes de programación orientada a objetos. Clases y objetos. Constructores. Herencia. Polimorfismo. Binding dinámico. El Sistema OO y su ejecución. Herencia. Subclasificación. Jerarquías de clases. Herencia simple y herencia múltiple. Clases abstractas. Intefaces. Herencia de implementación y de interface. Identidad y manipulación de objetos. Operaciones sobre referencias (clonación, comparación,..).
Persistencia de objetos. Composición de objetos. Características OO en lenguajes de programación: Smalltalk, C++, Ada95, Eiffel y Java. Estudio y práctica del Lenguaje Smalltalk como representativo del paradigma OO.

MÓDULO X: Paradigma de Programación funcional.
Tipos de datos. Tipos built-in y primitivos. Tipos ordinales definidos por el usuario. Tipos agregación y su implementación: Producto cartesiano. Uniones y uniones discriminadas. Mapeos Finitos. Arrays. Categorías de arrays: Estáticos, de pila dinámica, dinámicos de heap. Tipos Secuencia. Decisiones de diseño en Strings. Conjunto Potencia. Tipos Recursivos. Tipo Puntero. Inseguridad de los punteros. Punteros colgantes. Objetos perdidos. Estrategias de recolección de basura. Sistemas de tipos. Tipos monofórficos vs. Tipos polimórficos.Tipos de datos abstracto. Lenguajes que soportan TAD. Tipos abstractos parametrizados. Estructura de tipos de algunos lenguajes: Pascal, C++, Java, Ada. Lenguajes con sistemas fuertemente tipados. Constructores y

destructores. Gestión de almacenamiento.

MÓDULO XI: Paradigma de Programación lógica
Principios de programación lógica. Cálculo de Predicado. Definiciones básicas. Objetos y términos compuestos. Proposiciones. Conectores Lógicos. Cuantificadores. Forma Clausal. Características de forma de cláusula. Proceso de resolución. Cláusulas de Horn. Hechos, reglas, consultas y deducciones. Inferencia lógica. Politipos. Unificación. Estudio y práctica de PROLOG como lenguaje representativo de este paradigma. Características y evolución de PROLOG. Sentencias de Programas: Hechos, Reglas y Metas. Alternativas de definición de hechos. Reglas en PROLOG. Resolución simple. Ejemplo de Reglas. Metas. Reglas recursivas. Proceso de Inferencia en PROLOG. Backtracking. Tipos de Datos. Reglas de Calce. Listas. Unificación. Operadores y Aritmética. Consultas. Comparación de lenguajes imperativos y lógicos.

5. RECURSOS NECESARIOS

- Proyector
- Pc
-

6. PROGRAMACIÓN SEMANAL

Semana	Unidad / Módulo	Descripción	Bibliografía
1	I	Conceptos introductorios de Lenguajes. Práctica: Monografía Evolución de Lenguajes de Programación	Sebesta, Guezzi, Watt, Pratt; Frieman
2	II	Conceptos básicos de sintaxis y semántica. Gramáticas formales. Práctica: Gramática EBNF. Diagramas de sintaxis, árboles sintácticos y ambigüedad	Sebesta, Guezzi, Watt, Pratt, Friedman
3	III	Conceptos de Variables, binding, ámbito. Práctica en Laboratorio. Ejercicios de binding y ámbito en ADA y Phyton.	Sebesta, Guezzi, Watt, Pratt, Friedman
4	IV	Tipos de Datos y estructuración. Práctica en Laboratorio. Comparación de manejo de strings, punteros, tipos definidos por el usuario en C y ADA. Tipos abstractos en ADA y Phyton.	Sebesta, Guezzi, Watt, Pratt, Friedman
5	V	Estructuras de control. Práctica en Laboratorio: Expresiones y estructuras de control en C, Java y ADA.	Sebesta, Guezzi, Watt, Pratt, Friedman
6	VI	Manejo de excepciones. Práctica en Laboratorio: Ejercitación sobre manejo de excepciones en SCALA. Comparación con manejo de excepciones en C++ y Java.	Sebesta, Guezzi, Watt, Pratt, Friedman
7	VII	Abstracción Procedural, Modularidad, Genericidad Práctica en Laboratorio: Implementación de subprogramas, módulos y unidades genérica en ADA. Comparación con files y templates en C++ .	Sebesta, Guezzi, Watt, Pratt, Friedman
8	VIII	Concurrencia paralelismo. Práctica en Laboratorio: Concurrencia en ADA y Scala.	Sebesta, Guezzi, Watt, Pratt, Friedman
9	I-VIII	Presentación oral de Trabajos sobre lenguajes de programación Evaluación y comparación de lenguajes. Parcial Práctico Nº 1	
10	IX	Introducción Paradigma OO. Práctica: Clases y Objetos en Smalltalk. Herencia en Smalltalk y Scala.	Sebesta, Goldberg Ducasse
11	IX	Paradigma OO. Práctica: Polimorfismo en Smalltalk y Scala.	Sebesta, Goldberg, Ducasse

12	X	Introducción Paradigma funcional. Cálculo Lambda. Introducción a Scheme Práctica: Cálculo Lambda. Ejercitación básica en Lenguaje Scheme	Sebesta, Dybvig, Peyton Jones
13	X	Paradigma de Programación funcional. Lenguaje Scheme. Manejo de Listas y recursividad en Scheme.	Sebesta, Dybvig, Peyton Jones
14	XI	Paradigma de Programación Lógica. Conceptos básicos e Introducción a PROLOG. Práctica: Consultas de Bases de datos en PROLOG	Sebesta, Wylie Lloyd, Nilsson
15	XI	Paradigma de Programación Lógica. Práctica: Listas en PROLOG.	Sebesta, Wylie Lloyd, Nilsson
16	IX-XI	Presentación Trabajo final sobre Paradigmas de LP. Parcial Práctico Nº 2	
17	I-XI	Reunión de cátedra para evaluación del dictado de la cursada y ajuste de contenidos. Elaboración del Informe de cátedra	

7. BIBLIOGRAFIA DE LA ASIGNATURA

Autor	Año	Título	Capítulo/s	Lugar de la Edición	Editor / Sitio Web
Robert SEBESTA	2016	Concepts of Programming Languages, 11ª ed.	1-16	Colorado	Pearson
Carlo GHEZZI, Jazayeri Mehdi	2000	Programming Languages Concepts, 3rd ed.	1-8	New York	Wiley
WATT D.	2004	Programming Languaje Design Concepts	1-8	New York	WIley
Pratt Terrence	2001	Lenguajes de Programación, Diseño e Implementación, 4ª edición	1-9	New Jersey	Prentice Hall
Friedman, Wand, Haynes.	2001	Essentials of Programming Languages	1-4	Massachusetts	MIT Press
Goldberg Adele	1989	Smalltalk 80, the Language	1-11	EEUU	Addison Wesley Professional
Stéphane Ducasse, Dmitri Zagidulin, Nicolai Hess, Dimitris Chloupis	2015	Pharo By example		EEUU	http://files.pharo.org/books-pdfs/updated-pharo-by-example/2017-01-14-UpdatedPharoByExample.pdf
R. Kent Dybvig	2003	The Scheme Programming Language, 4a ed	1-6	Massachuttes	MIT Press

S. L. Peyton Jones	1987	The implementation of functional programming languages	1-5	New Jersey	Prentice-Hall - C.A.R. Hoare Series
John Wylie Lloyd	1984	Foundations of Logic Programming	1-6	Berlín, Alemania	Springer-Verlag
Ulf Nilsson and Jan Maluszynski	2012	Logic Programming and PROLOG, 2a ed	1-5	Suecia	Nilsson and Maluszynski

Firma del docente-investigador responsable

VISADO		
COORDINADOR DE LA CARRERA	**DIRECTOR DEL INSTITUTO**	**SECRETARIO ACADEMICO UNTDF**
Fecha :	Fecha :	