INSTITUTO DE DESARROLLO ECONÓMICO E INNOVACIÓN
Año: 2024

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:** 2022-03-22 **CARRERA/S:** Licenciatura en Sistemas 049/2017,
CARÁCTER: CUATRIMESTRAL (1ro) **TIPO:** OBLIGATORIA **NIVEL:** GRADO **MODALIDAD DEL DICTADO:** PRESENCIAL **MODALIDAD PROMOCION DIRECTA:** SI **CARGA HORARIA SEMANAL:** 8 HS **CARGA HORARIA TOTAL:** 120 HS	**EQUIPO DOCENTE**

Nombre y Apellido	Cargo	e-mail
Matías Gel	Profesor Adjunto	mgel@untdf.edu.ar
Martin Villarreal	Profesor Adjunto	mvillarreal@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.

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. Con fuerte énfasis en la programación funcional.
- 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

Entrega de los trabajos prácticos.
Trabajo de programación en relación a los paradigmas aprendidos que será practico, de una semana de duración y defendido por el alumno. Este corresponderá al primer parcial.
Se tomara como segundo parcial la preparación y presentación de un lenguaje elegido de los propuestos por la cátedra. El alumno deberá presentar el lenguaje bajo los conceptos adquiridos en la materia. Esta presentación cumplirá como segundo parcial.

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 Promoción de la asignatura, además de las condiciones básicas de cursada, se requiere:
Aprobación de los dos parciales prácticos en la primera instancia, con Nota igual o superior a siete (7)
Se tomara un examen escrito de conceptos teoricos de la materia.

Para la aprobación en condición libre de la materia el alumno deberá rendir una instancia practica resolviendo ejercicios prácticos en los 3 paradigmas dados, según el programa y un examen teórico que cubra los temas del programa de la asignatura.

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: Paradigma de Programación funcional.
Fundamentos de la programación funcional.
Estudio y práctica con Scheme como lenguaje representativo de este paradigma.
Aspectos avanzados de programación funcional.
Mecanismo de los lenguajes imperativos que dan soporte a la programación funcional.

MÓDULO IV: Paradigma de Programación lógica
Fundamentos de la programación lógica. Estudio y práctica de PROLOG como lenguaje representativo de este paradigma.

MÓDULO V: Paradigma de Programación orientada a objetos
Fundamentos de la programación Orientada Objetos Estudio y práctica del Lenguaje Smalltalk como representativo del paradigma OO. Ejemplos comparativos en diferentes lenguajes OO como C++, Python, Java.

MÓDULO VI: 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 VII: 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. Clases. Estructura de tipos de algunos
lenguajes: C++, Java, Python. Lenguajes con sistemas fuertemente tipados. Constructores y destructores. Gestión de almacenamiento.

MÓDULO VIII: 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. . Ejemplos de expresiones y estructuras de control
en C++, Python, Java.

MÓDULO IX: 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 C++, Python, Java. Comparaciones. Computación manejada por eventos

MÓDULO X: 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 C++, Python, Java.

MÓDULO XI: Concurrencia
Concurrencia. Conceptos fundamentales de concurrencia a nivel de subprogramas. Niveles de concurrencia. Threads. Sincronización de cooperación y competencia. Comunicación entre procesos. Procesos. Tareas. Modelos de implementación. Ejemplos en diferentes lenguajes.

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	Kenneth, Guezzi, Pratt; Sethi
2	II	Conceptos básicos de sintaxis y semántica. Gramáticas formales. Práctica: Gramática EBNF. Diagramas de sintaxis, árboles sintácticos y ambigüedad	Kenneth, Guezzi, Pratt; Sethi
3	III	Paradigma de Programación funcional desde un punto de vista practico. Lenguaje Scheme. Manejo de Listas y recursividad en Scheme.Ejercitación básica en Lenguaje Scheme	Kenneth, Kent
4	III	Fundamentos de del Paradigma funcional. Cálculo Lambda. Continua practica Scheme	Kenneth

5	III	Aspectos avanzados de Programación funcional. Lenguaje Scheme.Practica: continuan lenguaje Scheme	Kenneth, Kent
6	IV	Paradigma de Programación Lógica. Conceptos básicos e Introducción a PROLOG. Práctica: Consultas de Bases de datos en PROLOG	Kenneth, Kent
7	IV	Paradigma de Programación Lógica. Conceptos básicos e Introducción a PROLOG. Práctica: Consultas de Bases de datos en PROLOG	Kenneth, Bramer
8	V	Introducción Paradigma OO. Práctica: Clases y Objetos y poliformismo en Smalltalk.	Kenneth, Bramer
9	III-V	Presentación Trabajo final sobre Paradigmas de LP. Parcial Práctico Nº 1	
10	VI	Conceptos de Variables, binding, ámbito. Práctica en Laboratorio. Ejercicios de binding y ámbito. Ejemplos en diferentes lenguajes.	
11	VII	Tipos de Datos y estructuración. Práctica en Laboratorio. Comparación de manejo de strings, punteros, tipos definidos por el usuario . Tipos abstractos	Kenneth, Guezzi, Pratt; Sethi
12	VIII	Estructuras de control. Práctica en Laboratorio: Expresiones y estructuras de control en diferentes lenguajes.	Kenneth, Guezzi, Pratt; Sethi
13	IX	Manejo de excepciones. Práctica en Laboratorio: Ejercitación sobre manejo de excepciones comparando diferentes lenguajes.	Kenneth, Guezzi, Pratt; Sethi
14	X	Abstracción Procedural, Modularidad, Genericidad Práctica en Laboratorio: Implementación de subprogramas, módulos y unidades genérica.	Kenneth, Guezzi, Pratt; Sethi
15	XI	Concurrencia paralelismo. Práctica en Laboratorio: Concurrencia en Python.	Kenneth, Guezzi, Pratt; Sethi
16	VI-XI	Presentación oral de Trabajos sobre lenguajes de programación Evaluación y comparación de lenguajes. Parcial Práctico Nº 2	
17	I-Xi	Trabajo Final para promoción - 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
Louden, Kenneth C	2004	Lenguajes de programación : principios y práctica	1-12	España	Thomson
Carlo GHEZZI, Jazayeri Mehdi	1997	Programming Languages Concepts, 3rd ed.	1-8	New York	Wiley

Sethi, Ravi	1992	Lenguajes de programación : Conceptos y constructores	1-12	España	Addison-Wesley Iberoamericana
Pratt Terrence	1983	Lenguajes de Programación, Diseño e Implementación, 2ª edición	1-9	New Jersey	Prentice Hall
Robert SEBESTA	2019	Concepts of Programming Languages, 12ª ed.	1-16 complementaria	USA	Pearson
Wilf R. LaLonde, John R. Pugh	1990	Inside Smalltalk V.I	1-8	EEUU	Prentice Hall
Stéphane Ducasse, Dmitri Zagidulin, Nicolai Hess, Dimitris Chloupis	2015	Pharo By example	Complementaria	EEUU	http://files.pharo.org/bookspdfs/updated-pharo-byexample/2017-01-14-UpdatedPharoByExample.pdf
R. Kent Dybvig	2009	Scheme Programming Labguage	1-6	Massachuttes	MIT Press
S. L. Peyton Jones	1987	The implementation of functional programming languages	1-5 complementaria	New Jersey	Prentice-Hall - C.A.R. Hoare Series
Max Bramer	2005	Logic Programming with Prolog	1-7	Reino Unido	SpringerLink
Ulf Nilsson and Jan Maluszy?nski	2012	Logic Programming and PROLOG, 2a ed	1-5 complementaria	Suecia	Nilsson and Maluszynski

Firma del docente-investigador responsable

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

Este programa de estudio tiene una validez de hasta tres años o hasta que otro programa lo reemplace en ese periodo