CONCEPTOS Y PARADIGMAS DE LENGUAJES DE PROGRAMACIÓN

Año 2026

Carrera/ Plan:

Licenciatura en Informática Plan 2021/Plan 2015
Licenciatura en Sistemas Plan 2021/Plan 2015
Analista Programador Universitario Plan 2021/Plan 2015

Año: 3er
Régimen de Cursada: *Semestral*
Carácter (Obligatoria/Optativa): Obligatoria
Correlativas: SI203 (Algoritmos y Estructuras de Datos) / SI207 (Seminario de Lenguajes) / SI208 (Taller de lecto-comprensión y Traducción en inglés)
Profesor/es: C.C. Viviana Ambrosi /Mg. Harari, Ivana / Lic. Juan Devicenzi
Hs. Semanales teoría: 3hs
Hs. Semanales práctica: 6hs

FUNDAMENTACIÓN

El conocimiento profundo de los conceptos intrínsecos de los lenguajes de programación potencia y mejora el desarrollo del software proporcionando las herramientas necesarias para construir criterios sólidos de evaluación y comparación entre diferentes lenguajes.

Este aprendizaje amplía y consolida los conocimientos previamente adquiridos, al tiempo que incorpora nuevos elementos teóricos y prácticos para abordar, analizar y desarrollar distintos paradigmas y lenguajes de programación.

Los conceptos aprendidos serán aplicados en trabajos colaborativos, promoviendo el trabajo en equipo y el intercambio de ideas, y servirán como formación complementaria que fortalecerá las competencias de exposición oral y escrita. La participación activa, la producción de trabajos y la capacidad de comunicación técnica serán aspectos alentados y evaluados a lo largo de la cursada.

OBJETIVOS GENERALES

Introducir, analizar, comparar y evaluar los conceptos subyacentes de los lenguajes de programación en distintos paradigmas de programación.

Adquirir la capacidad de evaluar lenguajes de programación desde distintos puntos de vista, ya sea como diseñador, implementador o usuario del lenguaje.

RESULTADOS DE APRENDIZAJE

3.5. Aplicar las correspondientes competencias prácticas y de programación en la creación de programas informáticos y/u otros dispositivos informáticos (Básico).

COMPETENCIAS

- CGS1- Desempeñarse de manera efectiva en equipos de trabajo, con capacidad para organizarlos y liderarlos.

- CGS2- Comunicarse con efectividad en forma oral y escrita.

- CGT1- Identificar, formular y resolver problemas de Informática.

- CGT10 - Capacidad para realizar investigaciones bibliográficas y de diferentes fuentes de información a fin de obtener conocimiento actualizado en temas de la disciplina.

- LI - CE4 – Planificar, dirigir, realizar y/o evaluar proyectos de relevamiento de problemas del mundo real, especificación formal de los mismos, diseño, implementación, prueba, verificación, validación, mantenimiento y control de calidad de sistemas de software/sistemas de información que se ejecuten sobre equipos de procesamiento de datos, con capacidad de incorporación de tecnologías emergentes del cambio tecnológico. Capacidad de análisis, diseño y evaluación de interfaces humano-computador y computador-computador.

- LS - CE1 – Planificar, dirigir, realizar y/o evaluar proyectos de relevamiento de problemas del mundo real. Especificación formal, diseño, implementación, prueba, verificación, validación, mantenimiento y control de calidad de sistemas de software que se ejecuten sobre sistemas de procesamiento de datos, con capacidad de incorporación de tecnologías emergentes del cambio tecnológico. Capacidad de análisis, diseño y evaluación de interfaces humano-computador y computador-computador.

CONTENIDOS MÍNIMOS (de acuerdo al Plan de Estudios)

- Sintaxis y semántica.
- Semántica operacional.
- Entidades atributos y ligaduras.
- Sistemas de tipos.
- Encapsulamiento y abstracción.
- Intérpretes y Compiladores.
- Paradigmas de lenguajes (imperativo, orientado a objetos, funcional, lógico).
- Programación basada en scripting.

PROGRAMA ANALÍTICO

1. **Lenguajes de Programación.** Lenguajes como herramientas para producir software. Objetivo del estudio de sus conceptos. Distintos puntos de vista. Capacidad de comunicación. Relación e integración de los lenguajes de programación respecto a otros conceptos.
2. **Evaluación**. Evaluación de los lenguajes de programación a través de las características del software que producen. Principios de diseño de los lenguajes. Perspectiva histórica de los lenguajes de programación motivación, herencia, características, evolución. Lenguajes funcionales. Lenguajes Orientados a objetos. Abstracción: qué papel juega.

   Necesidad de estandarizar: ¿cuándo y cómo hacerlo?
3. **Estructura de un lenguaje.** Sintaxis y semántica. Sintaxis: Características de las sintaxis. Elementos de las sintaxis. Estructuras sintácticas. Reglas léxicas y sintácticas. Tipos de sintaxis. Sintaxis abstracta y concreta. Formas de definir la sintaxis. Gramáticas. (Backus Naum Forma). Árboles sintácticos y de derivación. Gramáticas recursivas. Subgramáticas. Gramáticas ambiguas. EBNF. Diagramas sintácticos (CONWAY). Semántica estática.
4. **Semántica.** Tipos de semánticas. Formas de definir la semántica de un lenguaje de programación. Nociones de semántica formal. Procesamiento de un lenguaje interpretación y traducción. Tipos de traductores. Comparación entre Traductor e Intérprete. Combinación de ambas técnicas. Compiladores. Etapas de Análisis y Síntesis. Optimización.
5. **Semántica Operacional.** Ligadura. Descriptores. Momentos de ligadura. Estabilidad. Variables. Arquitectura Von Newman. Atributos. Momentos y estabilidad. Nombre: características. Alcance: visibilidad, reglas. Tipo: definición, clasificación. L-valor: tiempo de vida, alocación. R-valor: constantes, inicialización. Alias. Unidades. Atributos. Representación en ejecución. Elementos.

Unidades recursivas. Unidades genéricas. Alias y sobrecarga. Procesador abstracto: elementos, instrucciones. Procesamiento de un lenguaje: clasificación. Lenguaje estático. Entidades locales. Rutinas internas. Compilación separada. Lenguajes basados en pila. Unidades recursivas, implementación. Estructura de bloque. Datos semidinámicos y dinámicos. Lenguajes dinámicos.

6. **Compartir Datos.** Ambiente común. Acceso al ambiente no-local. Parámetros. Ventajas. Evaluación de los parámetros reales y ligadura con los parámetros formales. Clase de parámetros: Datos y Subprogramas. Modos de pasaje de parámetros datos. Pasaje de Rutinas como parámetros.
7. **Sistema de tipos.** Tipos predefinidos, tipos definidos por el usuario, tipos estructurados, tipos abstractos. Implementación de datos, su representación. Sistema de tipos: lenguajes seguros y fuertemente tipados. Seguridad en el manejo de tipos. Encapsulamiento y abstracción. Evolución de los tipos. Tipos Abstractos. Equivalencia de tipos.
8. **Abstracción de Control.** Estructuras de control: Definición de estructuras de control a nivel de sentencia y a nivel de unidad. Tipos de estructuras de control a nivel de sentencia. Diferencia entre sentencia de asignación y expresión. Evolución de las sentencias de selección.
9. **Abstracción de control a nivel de unidad**. Excepciones: Definición. Modelos de Terminación y Reasunción. Distintos modelos de manejo de excepciones. Comparación.
10. **Paradigma funcional.** Características. Comparación de lenguaje imperativo con lenguaje funcional. Definiciones de funciones. Script. Expresión y valor. Transparencia referencial. Evaluación de las expresiones, mecanismo de reducción o simplificación. Orden aplicativo, orden normal (lazy evaluation). Tipos de datos básicos y derivados. Tipos de funciones. Expresiones polimórficas. Currificación. Cálculo Lambda. Dominios de Aplicación.
11. **Paradigma Orientado a Objetos**. Características. Elementos básicos de la programación orientada a objetos: objetos, mensajes, métodos, clases. Conceptos de generalización, especificación y herencia. Diferentes tipos de herencia. Lenguajes híbridos, características principales. Dominios de aplicación. Programación Orientada a Aspectos.
12. **Paradigma lógico.** Características. Elementos de la programación lógica: variables, constantes, términos compuestos, listas. Cláusulas y predicados. Reglas y hechos. Dominios de aplicación.
13. **Programación basada en scripting.** Definición. Introducción histórica. Características. Tipos. Dominios de interés. Los lenguajes de scripting y la WWW. Aspectos innovadores.

BIBLIOGRAFIA

Bibliografía básica:

- **Ghezzi, C., & Jazayeri, M.** (2003). Programming Language Concepts (3rd ed.). John Wiley & Sons.
- **Sebesta, R. W.** (2019). Concepts of Programming Languages (12th ed.). Pearson.
- **Louden, K. C., & Lambert, K. A.** (2011). Programming Languages: Principles and Practice (3rd ed.). Cengage Learning.
- **Pratt, T. W., & Zelkowitz, M. V.** (2001). Programming Languages: Design and Implementation (4th ed.). Prentice Hall.
- **Sethi, R.** (1996). Programming Languages: Concepts and Constructs (2nd ed.). Addison-Wesley.
- **Scott, M. L.** (2021). Programming Language Pragmatics (5th ed.). Morgan Kaufmann/Elsevier.

Bibliografía complementaria de la cátedra:

- **Friedman, D. P., Wand, M., & Haynes, C. T.** (2008). Essentials of Programming Languages (3rd ed.). The MIT Press.
- **Horowitz, E., & Sahni, S.** (1990). Fundamentals of Programming Languages (2nd ed.). Springer-Verlag.
- **Tucker, A. B., & Noonan, R. E.** (2007). Programming Languages: Principles and Paradigms (2nd ed.). McGraw-Hill.
- **Watt, D. A., & Brown, D. F.** (2000). Programming Language Design Concepts. John Wiley & Sons.
- **Krishnamurthi, S.** (2012). Programming Languages: Application and Interpretation (2nd ed.). Disponible gratis en línea en Brown University. Para paradigmas modernos.
- **Pierce, B. C.** (2002). Types and Programming Languages. The MIT Press. Para profundizar teoría de tipos, para nuevos lenguajes.
- **Van Roy, P., & Haridi, S.** (2004). Concepts, Techniques, and Models of Computer Programming. The MIT Press. Para multiparadigma: funcional, lógico, concurrente.
- **Nanz, S., & Furia, C. A.** (2017). Programming Language Concepts (1st ed.). Springer. Para estudios avanzados.

METODOLOGÍA DE ENSEÑANZA

Se pone énfasis en el proceso de identificación de problemas del mundo real desde la visión del desarrollador, basados en la evaluación, comportamiento y características de los conceptos que atraviesan a los lenguajes de programación.

Las actividades se desarrollan bajo la modalidad de teorías y prácticas distribuidas equitativamente de acuerdo con el Plan de Estudios vigente.

Los contenidos del programa se presentan y analizan en las clases teóricas y se consolidan con los trabajos prácticos. Existe una estrecha relación entre la teoría y la práctica. En la teoría, que toma sentido y se fortalece con las prácticas, se realiza un análisis vertical, indagando sobre los conceptos y mostrando ejemplos de la aplicación de los mismos en diferentes lenguajes. En la práctica, se realiza un análisis horizontal debido a que se consideran uno o más lenguajes y se aplican los conceptos vistos en la teoría en cada uno de ellos.

Los estudiantes desarrollan actividades individuales y grupales (entre 2 y 3 alumnos) para realizar un seguimiento y avance gradual sobre los temas abordados. En las prácticas, se presentan ejercicios que forman parte de un trabajo integrador, cuya entrega es obligatoria. En los mismos se plantean diferentes actividades de investigación y desarrollo que posibilitan poner en práctica sus propios conocimientos.

La metodología de investigación que propone la cátedra propicia que el alumno realice las siguientes actividades:

- Búsqueda de bibliografía actualizada sobre el tema.
- Consulta a profesionales y catedráticos sobre temas investigados
- Discusión de alternativas tecnológicas para resolver el tema propuesto en diferentes lenguajes de programación.

Por otro lado, desde la cátedra se pretende fomentar el trabajo en equipo y la adquisición de experiencia en la comunicación escrita y oral de sus trabajos.

Los alumnos también cuentan con "cuestionarios online" que les dan la oportunidad de autoevaluarse para saber si han comprendido los conceptos enseñados.

Se ofrecen cuatro horarios de práctica distribuidos semanalmente en franjas horarias (mañana, tarde y noche) a fin de cubrir todas las posibilidades que faciliten la asistencia.

Para el dictado de las clases teórico-prácticas se utilizan diferentes herramientas didácticas que posibiliten asistencia presencial y remota.

Los estudiantes cuentan con un entorno educativo web que permite mantener una comunicación dinámica. Se hace uso de la misma para:

- Medio informativo: Contiene el cronograma de toda la cursada, el programa de la materia, bibliografía, guías de clases teóricas, ejercicios integradores y trabajos prácticos.
- Medio de comunicación: Desde el punto de vista de los docentes: Contiene novedades de la cursada, fechas de los parciales y entregas y, resultados de las correcciones de las diferentes evaluaciones. Desde el punto de vista de los estudiantes brinda la posibilidad de comunicarse con los profesores y jefes de trabajos prácticos a través de sus cuentas de e-mail y foro de consultas
- Medio de evaluación: Recurso utilizado para armar encuestas, cuestionarios autoevaluativos, mini evaluaciones y para subir entrega de tareas con calificación.

Paralelamente, el equipo docente mantiene reuniones al inicio de cada ciclo lectivo para delinear la planificación anual, analizar los resultados del año precedente, y analizar la encuesta enviada por la Facultad realizada a los estudiantes. Mensualmente se realizan reuniones con todos los integrantes de la materia, desde profesores hasta adscriptos. Se realiza una revisión de los temas desarrollados para contemplar temáticas presentadas a lo largo de la cursada o la inclusión de nuevos conceptos y lenguajes.

EVALUACIÓN

Durante todo el desarrollo de la cursada se realiza un seguimiento del estudiante, registrando en una planilla su participación en diferentes actividades planteadas por la cátedra. En dicha planilla se registran actividades obligatorias y no obligatorias. Las actividades que no son de carácter obligatorio están pensadas para incentivar al estudiantado a tener una participación activa a lo largo de la cursada. Si bien no es obligatoria su realización, algunas de ellas, las llamadas “Evaluaciones Mínimas Teóricas” (EMT) son evaluadas y, aquellos estudiantes que las rinden y aprueban el 60% de las mismas tienen un beneficio adicional, que está relacionado con la obtención de la promoción de la materia.

Respecto a la aprobación de la cursada, los estudiantes deben aprobar obligatoriamente:

- un **trabajo integrador**, que consiste en la investigación de los diferentes conceptos vistos a lo largo de la materia aplicados a los lenguajes de programación asignados por la cátedra y,
- La aprobación de dos **exámenes teóricos-prácticos** en donde se plantean diferentes ejercicios que deben ser desarrollados, relacionando los conceptos abordados en la materia.

El trabajo integrador tiene dos fechas de entrega y se presenta al iniciar la cursada. En la primera fecha se evalúa la primera parte del desarrollo y, en la segunda entrega se evalúa el trabajo en forma completa, incluyendo las modificaciones recomendadas en la primera evaluación. Es obligación su "entrega" y "aprobación" para poder rendir el examen teórico-práctico.

El desarrollo de investigación que propone el trabajo integrador debe ser presentado, por el grupo de estudiantes, a través de un informe escrito que será evaluado por los docentes teniendo en cuenta el contenido técnico, la estructura, la organización, la sintaxis, la claridad conceptual, y la bibliografía consultada y citada rigurosamente.

Como parte de la evaluación del trabajo integrador se tendrá en cuenta el desempeño grupal e individual de cada uno de los integrantes, a través de un coloquio. Respecto a la parte grupal, los grupos responderán aspectos generales de las tareas asignadas y darán explicaciones individuales que permitirán calificar diferentes aptitudes de los miembros del equipo (conocimientos / modo de expresarse / predisposición al trabajo colaborativo). En lo que respecta a la parte individual, cada uno de los integrantes del grupo deberá exponer sobre la tarea realizada. El docente indagará sobre sus conocimientos, su claridad de la presentación, su organización y la forma de expresión.

Los exámenes teóricos-prácticos, tienen dos fechas más que son de recuperación y se establecen de acuerdo a la duración del semestre fijado por el Calendario Académico de la Facultad.

Todas las fechas son publicadas al principio del ciclo lectivo y, se organizan de forma tal de que no entorpezcan el normal desarrollo de la cursada.

En las planillas de seguimiento, a parte del registro de las evaluaciones realizadas por los estudiantes, también se registrará información relacionada con: la capacidad del mismo para desarrollar su aprendizaje, claridad de las presentaciones realizadas, forma de organización y expresión en las diferentes instancias de evaluación oral, formulación de la solución de los diferentes desafíos en forma autónoma, entre otros.

Para la evaluación final se tomará una prueba teórica individual y escrita. La misma puede ser realizada en las mesas de finales correspondientes al calendario académico o al finalizar la cursada. A esta última opción solo pueden acceder los y las estudiantes que cumplan con las condiciones establecidas por la cursada para obtener la promoción de la materia.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	**10/3**	PRESENTACIÓN DE LA ASIGNATURA: pautas Introducción, evaluación, historia de los lenguajes. ESTRUCTURA DE UN LENGUAJE: sintaxis y semántica. Sintaxis.
2	**17/3**	SEMÁNTICA. Procesamiento de un lenguaje: interpretación y compilación
3	**24/3**	SEMÁNTICA OPERACIONAL: entidades y ligaduras. Variables
4	**31/3**	SEMÁNTICA OPERACIONAL: unidades. Procesamiento de lenguajes estáticos
5	**7/4**	SEMÁNTICA OPERACIONAL: unidades. Procesamiento de lenguajes basados en pila y dinámicos.
6	**14/4**	COMPARTIR DATOS: casos. Parámetros
7	**21/4**	SISTEMA DE TIPOS: características, tipos predefinidos y tipos definidos por el usuario.
8	**28/4**	SISTEMA DE TIPOS: tipos compuestos y tipos abstractos. Lenguajes seguros, equivalencia y compatibilidad.
9	**5/5**	ABSTRACCIÓN DE CONTROL: a nivel de sentencia y a nivel de unidad.
10	**12/5**	EXCEPCIONES.
11	**19/5**	PARADIGMAS. Funcional. Orientado a Objetos. Lógico. Prog. Orientada a Aspectos.
12	**26/5**	SCRIPTING: Programación basada en scripting
13	**2/6**	Repaso

Evaluaciones previstas		Fecha
Parcial 1	**1er Fecha**	24-4
	2da Fecha	8-5
	3er Fecha	13-7
Parcial 2	**1er Fecha**	12-6
	2da Fecha	26-6
	3er Fecha	13-7
EMT	**1er Fecha**	7/4
	2da Fecha	5/5
	3er Fecha	2/6

TRABAJO FINAL	1er Entrega	15/6
	2da Entrega	6/ 7
EXÁMEN TEÓRICO PROMOCIÓN		7/7

Contacto de la cátedra (mail, sitio WEB, plataforma virtual de gestión de cursos):

vambrosi@info.unlp.edu.ar / iharari@info.unlp.edu.ar / jdevicenzi@info.unlp.edu.ar

Prof. Viviana M. Ambrosi

Lic. Juan Devicenzi

Firma del/los profesor/es