Facultad Regional Córdoba

Carrera: Ingeniería en Sistemas de Información

Asignatura: Paradigmas de Programación

Planificación a partir del Ciclo Lectivo 2025

1. Datos administrativos de la asignatura			
Nivel en la carrera	2	Duración	Cuatrimestral
Plan	2023		
Bloque curricular:	Tecnologías Básicas		
Carga horaria presencial semanal (hs. cátedra):	8	Carga Horaria total (hs. reloj):	96
Carga horaria no presencial semanal (hs. reloj) (si correspondiese)		% horas no presenciales (hs. reloj) (si correspondiese)	

2. Presentación, Fundamentación

La asignatura, Paradigmas de Programación perteneciente a la carrera de Ingeniería en Sistemas de Información, se encuentra dentro del bloque de tecnologías básicas, dentro del área de conocimiento desarrollo de software, en el diseño curricular de la carrera descripto mediante la Ord. 1877 del Concejo Superior de la Universidad Tecnológica Nacional.

La construcción de software y sistemas de software asociados a los sistemas de información, son realizados a través de lenguajes de programación los cuales están agrupados y responden en lo que se denominan paradigmas de programación.

Un paradigma resume en su esencia, un conjunto de características básicas para enfrentar y dar solución a un conjunto de problemas.

Los lenguajes de programación han ido evolucionando a través de más de medio siglo, con diversas filosofías en su construcción y forma de utilización para la resolución de problemas que dan origen a diferentes filosofías, con características propias, denominadas paradigmas de programación.

Los lenguajes de programación constituyen una herramienta imprescindible para la construcción del software y sistemas de Software asociados a los Sistemas de Información. La forma de abordaje de los problemas, su formulación, representación y la resolución de estos, quedarán íntimamente relacionados con los paradigmas utilizados. Su correcta compresión contribuye a la formación de un profesional integral, con capacidad de resolver problemas a través del empleo de diferentes herramientas de programación, interactuando con diferentes entornos y lenguajes de programación, seleccionando la mejor opción de acuerdo con el tipo de problema y las capacidades del equipo de desarrollo del que dispone. Por tanto, debe tener un conocimiento acabado de las virtudes, fortalezas y prestaciones de los lenguajes de programación, para de esta manera hacer

un uso eficiente de estas herramientas. Para lograr lo anteriormente expuesto debe conocer los principios constitutivos de los lenguajes y los paradigmas de manera de asociarlos a la mejor opción en la resolución de un problema, y ejercitar su utilización de manera de producir un aprendizaje significativo en el futuro profesional.

Relación de la asignatura con el perfil de egreso
La asignatura aporta a la formación del egresado, de una sólida base científica y técnica necesaria para la resolución eficaz, y eficiente de problemas mediante el empleo de Lenguajes de Programación, asociados a los Paradigmas fundamentales de programación.
Por otro lado, brinda herramientas concretas, para la identificación de los elementos constitutivos de los diferentes Lenguajes y Paradigmas, que son utilizados en la construcción de Lenguajes de Programación, los cuales definen las características de los Lenguajes y las diferentes estrategias definidas para representar una solución determinada.

Relación de la asignatura con los alcances del título
Al ser una asignatura perteneciente al área de Desarrollo de Software, se relaciona directamente con la construcción e implementación de una solución real a problemas informáticos, con la aplicación de buenas prácticas de programación, que son necesarias en el desempeño que el Ingeniero en Sistemas de Información, las cuales, deberá aplicar en el desarrollo de todas sus actividades profesionales.
Además, se fundamenta en el uso eficiente de las características esenciales de los denominados paradigmas fundamentales de programación, y sus lenguajes de programación dentro de cada paradigma, para ser aplicados en la construcción de soluciones informáticas, aportando directamente a los alcances AR1, AL4 y AL9, y en menor medida, ya que solo se cubren los aspectos conceptuales básicos y fundamentales a los alcances AR3 y AR4.

3. Relación de la asignatura con las competencias de egreso de la carrera

En la tabla siguiente se establece la relación de la asignatura con las competencias de egreso: Específicas, Genéricas Tecnológicas y Genéricas Sociales, Políticas y Actitudinales de la carrera. Se incluyen las competencias de egreso a las que tributa, aportes reales y significativos de la asignatura, y en qué nivel (no aporta, bajo, medio, alto).

Competencias	Nivel
Competencias genéricas tecnológicas (CG):	
CG.1. Identificación, formulación y resolución de problemas de ingeniería en sistemas de información/informática.	Bajo
CG.2. Concepción, diseño y desarrollo de proyectos de Ingeniería en Sistemas de Información/Informática	No aporta
CG.3. Gestión, planificación, ejecución y control de proyectos de Ingeniería en Sistemas de Información/Informática.	No aporta
CG.4. Utilización de técnicas y herramientas de aplicación de Ingeniería en Sistemas de Información/Informática.	Bajo
CG.5. Generación de desarrollos tecnológicos y/o innovaciones tecnológicas.	No aporta

Competencias genéricas sociales, políticas y actitudinales (CG)	
CG.6. Fundamentos para el desempeño en equipos de trabajo.	No aporta
CG.7. Fundamentos para una comunicación efectiva.	No aporta
CG.8. Fundamentos para una actuación profesional ética y responsable.	No aporta
CG.9. Fundamentos para evaluar y actuar en relación con el impacto social de su actividad profesional en el contexto global y local.	No aporta
CG.10. Aprender en forma continua y autónoma.	Bajo
CG.11. Fundamentos para el desarrollo de una actitud profesional emprendedora	No aporta
Competencias Específicas de la carrera	
CE1.1.Especificar, proyectar y desarrollar sistemas de información para concebir soluciones tecnológicas que permitan resolver situaciones en las organizaciones mediante el empleo de metodologías de sistemas y tecnologías asociadas a los sistemas de información.	No aporta
CE1.2.Especificar, proyectar y desarrollar sistemas de comunicación de datos, evaluando posibles soluciones tecnológicas disponibles para dar soporte a los sistemas de información en lo referido al procesamiento y comunicación de datos.	No aporta
CE1.3. Especificar, proyectar y desarrollar software para la elaboración de soluciones informáticas con el propósito de resolver problemas estratégicos y operativos, así como de servicios y de negocios, en el marco de una actividad económica que sea social y ambientalmente sustentable.	Bajo
CE2.1. Proyectar y dirigir lo referido a seguridad informática para seleccionar y aplicar técnicas, herramientas, métodos y normas, garantizando la seguridad y privacidad de la información procesada y generada por los sistemas de información.	No aporta
CE.3.1. Establecer métricas y normas de calidad de software para medir, evaluar, controlar y monitorear el rendimiento, impulsando mejoras de acuerdo a técnicas y normas vigentes definidas por los organismos de estandarización.	No aporta
CE.4.1. Certificar el funcionamiento, condición de uso o estado de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software para asegurar la generación de los resultados deseados en función de restricciones de tiempo y recursos establecidos.	No aporta

CE.5.1. Dirigir y controlar la implementación, operación y mantenimiento de sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software, a los fines de alcanzar los objetivos fijados por la organización.	No aporta
CE.6.1. Asesorar y capacitar a organizaciones, empresas, organismos públicos o privados en la adquisición, instalación y uso, en lo que respecta a sistemas de información, sistemas de comunicación de datos, software, seguridad informática y calidad de software, a los fines de un uso correcto de los sistemas intervinientes.	No aporta
CE.7.1. Realizar pericias, tasaciones y arbitrajes relacionados con su actividad profesional, respetando marcos normativos y jurídicos con el objeto de asesorar a las partes o a los tribunales de Justicia.	No aporta

4. Contenidos Mínimos

-Concepto de Paradigmas de Programación.
- Paradigma Funcional.
- Lenguajes de Programación Funcional.
- Paradigma Lógico.
- Lenguaje de Programación Lógica.
- Paradigma Orientado a Objetos.
- Lenguajes de Programación Orientados a Objetos.

5. Objetivos establecidos en el DC

- Comprender los fundamentos de los paradigmas de programación asociados a lenguajes de programación concretos.
- Aplicar los diferentes paradigmas en la resolución de problemas.
- Adquirir criterios para la selección del paradigma de programación a utilizar en un caso concreto.

6. Resultados de aprendizaje

Los siguientes resultados de aprendizaje se promueven en el desarrollo de la asignatura

Identificador de RA	Redacción	
RA1	Identificar las características constitutivas de cada uno de los paradigmas y lenguajes de programación para asociarlos con diferentes prototipos de problemas que sean factibles de resolver de manera eficiente con cada uno de ellos.	
RA2	Emplear los principios constitutivos y filosóficos del paradigma orientado a objetos, haciendo uso de un lenguaje puro de dicho paradigma, para resolver diferentes dominios de problemas de ingeniería.	
RA3	Aplicar expresiones propias de un lenguaje dentro del Paradigma funcional, haciendo uso de funciones, para la resolución de problemas ingenieriles.	
RA4	Utilizar predicados y cláusulas de primer orden, bajo el paradigma de programación lógico, para resolver problemas ingenieriles de diferentes dominios.	Desa aplica hacie razon

7. Relación de los RA y las competencias

En la tabla siguiente se indica con X la tributación de cada Resultado de Aprendizaje con las competencias de egreso: específicas, genéricas tecnológicas, sociales, políticas y actitudinales de la carrera.

RA	CE1.1	CE1.2	CE1.3	CE2.1	CE3.1	CE4.1	CE5.1	CE6.1	CE7.1	CG1	CG2	CG3	CG4	CG5	CG6	CG7	CG8	CG9	CG10	CG11
RA1	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	X	-
RA2	-	-	X	-	-	-	-	-	-	X	-	-	X	-	-	-	-	-	X	-
RA3	-	-	X	-	-	-	-	-	-	X	-	-	X	-	-	-	-	-	X	-
RA4	-	-	X	-	-	-	-	-	-	X	-	-	X	-	-	-	-	-	X	-

8. Asignaturas correlativas previas

Para cursar y rendir debe tener cursadas:

- Asignatura/s:

  Lógica y Estructuras Discretas.

  Algoritmos y Estructuras de Datos.

Para cursar y rendir debe tener aprobada:

- Asignatura/s:

  Ninguna

9. Asignaturas correlativas posteriores

Indicar las asignaturas correlativas posteriores:

- Asignatura/s:

Desarrollo de Software.

Diseño de Sistemas de Información (Integradora).

Ingeniería y Calidad de Software.

10. Programa analítico

Este programa analítico contempla los contenidos mínimos, previstos en el DC vigente, y aquellos que se consideran necesarios para desarrollar los resultados de aprendizaje propuestos.

Unidad Nº:1

Título: **Características esenciales de los Lenguajes y Paradigmas de Programación.**

Contenidos: Conceptos generales: Programas, paradigmas, lenguajes de programación, y programación.
Paradigmas fundamentales: Clasificación y evolución histórica, definición, lenguajes asociados, ventajas, limitaciones y áreas de aplicación.
Diferencia entre lenguaje y paradigma de programación.
Lenguajes de programación: Conceptos, criterios de evaluación, reseña histórica y evolución y tipos de lenguajes: híbridos y puros.

Carga horaria por Unidad:

8 horas totales de clases.

Unidad Nº:2

Título: **Paradigma Imperativo: Secuencial o Estructurado.**

Contenidos: Programación estructurada. Características generales: Variables locales y globales. Estructuras de control. Modularización (funciones y procedimientos). Lenguaje Python: Tipos de datos. Declaración de variables. Operadores. Procedimientos. Funciones. Estructuras de control. Funciones de entrada y salida.

Lenguaje asociado: Python. Entornos Integrados de Desarrollo.

Carga horaria por unidad:

8 horas totales de clases.

Unidad Nº:3

Título: **Paradigma de Programación Orientado a Objetos.**

Contenidos: Presentación del paradigma. Conceptos fundamentales. Abstracción de datos y ocultamiento de la información. Estructura de un objeto. Métodos y mensajes. Clasificación. Clase. Concepto de generalización-especialización. Composición Herencia: Estrategias y modelos. Polimorfismo. Polimorfismo: Definición, tipos. Conceptos del modelo de objetos en SmallTalk. Desarrollo de la sintaxis de objetos en Smalltalk. Expresiones literales. Caracteres, secuencia de caracteres, símbolos y números. Expresiones de asignación y variables. Asignación. Tipos de variables. Variables privadas: de instancia nombradas e indexadas, argumentos y temporales. Variables compartidas: de clase, globales y pool. Variables especiales: self y super. 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. Expresiones de bloque. Clase Context. Bloques con y sin argumentos. Evaluación de bloques. Métodos y expresiones de retorno. Sintaxis de la definición de un método. Significado de la expresión de retorno. Métodos de clase e instancia. Implementación de Composición en Smalltalk. Implementación de Herencia en Smalltalk (Definición de una Subclase, uso de super, herencia de variables, inicialización de atributos en una clase Hija, herencia de métodos, clases abstractas). Implementación de Polimorfismo en Smalltalk. Colecciones en Smalltalk: Introducción, jerarquía, colecciones básicas (Set, Bag, OrderedCollection, SortedCollection, Array, Dictionary), operaciones básicas, conversión entre colecciones.

Lenguaje asociado: Smalltalk
Entorno asociado: Pharo. Imagen, ambiente de objetos, definición y uso de clases y objetos.

Carga horaria por unidad:

48 horas totales de clases.

Unidad Nº:4

Título: **Paradigma de Programación Funcional.**

Contenidos: Presentación del paradigma. Historia. Características. Ventajas/Desventajas. Áreas de Aplicación. Ejemplos de implementaciones. Familia de Lenguajes. Conceptos generales: Funciones matemáticas, Sintaxis en el paradigma funcional. Abstracción Funcional.

Funciones de orden superior. Cálculo Lambda, evaluación postergada. Lenguaje Haskell. Introducción. Entorno de Haskell – HUGS. Sintaxis. Comentarios. Tipos de datos. Sistemas de inferencia de tipos. Flujo de control. Definición de Funciones. Currificación. Reducción de expresiones. Evaluación. Tuplas. Recursividad. Listas: definición, listas por comprensión. Tipos definidos por el usuario. Tipos polimórficos.

Lenguaje asociado: Haskell
Entorno asociado: WinHugs

Carga horaria por unidad:

28 horas totales de clases.

Unidad Nº:5

Título: **Paradigma de Programación Lógico.**

Contenidos: Presentación del paradigma. Fundamentación lógica. Predicados y términos. 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. Interpretación algorítmica: Procedimientos y programación. Intérprete no determinista. Estrategia de evaluación. PROLOG Intérprete determinístico, “backtracking”. Orden de evaluación de cláusulas. Terminación. Sintaxis PROLOG. Cláusulas, predicados y términos. Distintos tipos de datos. Recursión en PROLOG. Tipos de datos recursivos, lista. Concepto de variable o incógnita. Unificación. Múltiples resultados. Inversibilidad. Aritmética, evaluación de expresiones aritméticas. Negación. Functores.

Lenguaje asociado: Prolog.
Entorno asociado: WinProlog.

Carga horaria por unidad:

28 horas totales de clases.

Unidad Nº:6

Título: **Elementos constitutivos de Lenguajes y Paradigmas**

Contenidos: Conceptos lógicos y transversales. Tipos de datos: Teoría, clasificación, verificación, sistema de tipos, conversión y ejemplos en los diferentes lenguajes de programación.
Mecanismos de control de flujo: Organización y ejemplos en los diferentes lenguajes de programación.
Abstracción y modularización: definición y mecanismos de implementación.

Carga horaria por unidad:

8 horas totales de clases.

Carga horaria por tipo de formación práctica de toda la asignatura

Tipo de formación práctica	Horas reloj
Formación experimental	12
Análisis y resolución de problemas de ingeniería y estudios de casos	24
Formulación, análisis y desarrollo de proyectos.	12

Bibliografía Obligatoria:

- [Cátedra Paradigmas de Programación, 2021] Material correspondiente a Unidad 1 elaborado por los docentes de la Cátedra y puesto a disposición de los estudiantes en la UV. Accesible desde la web: https://ria.utn.edu.ar/xmlui/handle/20.500.12272/5213.

- [Cátedra Paradigmas de Programación, 2021] Material correspondiente a Unidad 2 elaborado por los docentes de la Cátedra y puesto a disposición de los estudiantes en la UV. Accesible desde la web: http://hdl.handle.net/20.500.12272/5214.

- [Cátedra Paradigmas de Programación, 2021] Material correspondiente a Unidad 3 elaborado por los docentes de la Cátedra y puesto a disposición de los estudiantes en la UV. Accesible desde la web: http://hdl.handle.net/20.500.12272/5215.

- [Cátedra Paradigmas de Programación, 2021] Material correspondiente a Unidad 4 elaborado por los docentes de la Cátedra y puesto a disposición de los estudiantes en la UV. Accesible desde la web: http://hdl.handle.net/20.500.12272/5217.

- [Cátedra Paradigmas de Programación, 2021] Material correspondiente a Unidad 5 elaborado por los docentes de la Cátedra y puesto a disposición de los estudiantes en la UV. Accesible desde la web: http://hdl.handle.net/20.500.12272/5218.

- [Cátedra Paradigmas de Programación, 2021] Material correspondiente a Unidad 6 elaborado por los docentes de la Cátedra y puesto a disposición de los estudiantes en la UV. Accesible desde la web: http://hdl.handle.net/20.500.12272/5219.

- [Cátedra de Paradigmas de Programación, 2023 - Guía de trabajos prácticos en UV (Moodle de Cada Curso)

Bibliografía optativa y otros materiales a utilizar en la asignatura:

- [Sebesta, 2011] Robert Sebesta (2011) - Concepts of Programming Languages
- [Lalonde 1991] Lalonde y J. Pugh; Inside Smalltalk. Vol. I, II: Prentice Hall International.
- [Goldberg 1983] Adele Goldberg and David Robson, Smalltalk 80- The Language its implementation. ISBN 10: 0201113716 ISBN 13: 9780201113716
- [Wirfs-Brock 1986] Wirfs-Brock, R. &Et all, Designing Object-Oriented Software, Prentice Hall. Accesible desde http://stephane.ducasse.free.fr/FreeBooks/BlueBook/Bluebook.pdf
- [Varela 2011] Carlos Varela. “Programming Languages”. Rennselaer Polytechnic Institute. USA. 2011.
- [Jiménez 2010] Alonso Jiménez, José. “Programación Declarativa: Definición de listas por comprensión”. Departamento de Ciencias de la Computación e I. A. Universidad de Sevilla. 2010.

- [Pereira 1987] Fernando C. N. Pereira and Stuart M. Shieber. Prolog and Natural Language Analysis. Cambridge University Press, ISBN: 0-9719997-0-4, 1987

11. Metodología de enseñanza

Se aplicará el enfoque de enseñanza basado en competencias, bajo el cual consideraremos a la estudiante o al estudiante, como centro principal del proceso educativo. De esta manera, los estudiantes podrán ser coprotagonistas junto con nosotros, y responsables de su propio aprendizaje.

En general, se emplearán las siguientes estrategias didácticas:

- Lección magistral participativa.
- Preguntas exploratorias.
- Presentación de problemas típicos factibles de resolver con cada paradigma.
- Foros de debate virtual.
- Resolución de ejercicios y problemas de programación para cada uno de los paradigmas: orientado a objetos, funcional y lógico.
- Formación experimental en laboratorios de acceso local.
- Aprendizaje cooperativo en grupos pequeños.

La aplicación de las estrategias didácticas anteriormente mencionadas, serán desarrolladas con la finalidad de facilitar en los estudiantes la construcción de aprendizajes significativos y continuos, fomentando su trabajo autónomo como así también en pequeños grupos, lo cual les permitirá a los estudiantes el desarrollo de su capacidad creativa, crítica y reflexiva para la resolución de los diferentes problemas de programación propuestos en las guías de actividades prácticas, como así también en los trabajos prácticos integradores.

La cátedra pone a disposición de los estudiantes, como parte básica del material didáctico de estudio provisto: los apuntes de cátedra los cuales son actualizados para cada ciclo lectivo, como así también, las diapositivas de clases, además de las guías de actividades prácticas para cada uno de los paradigmas de programación bajo estudio. Tanto los materiales didácticos, como así también la totalidad de las actividades de enseñanzas, aprendizajes y evaluativas, son gestionadas desde las aulas virtuales de la plataforma Moodle.

En general, cada clase es iniciada con una breve revisión de actividades y desempeños de la clase anterior para integrarlos luego con los propios de la actual clase, y de esta manera facilitar en los estudiantes la construcción de aprendizajes significativos y continuos.

Los docentes han diseñado actividades de ejercitación para cada uno de los paradigmas de programación, con la finalidad de propiciar en los estudiantes experiencias de aprendizajes significativos. Así mismo, durante las clases, los docentes también serán facilitadores de estas experiencias de aprendizajes, y acompañarán, orientarán, apoyarán y guiarán en el desarrollo de

las diferentes actividades y desempeños implicados en la resolución de ejercicios y problemas de programación por parte de los estudiantes. De esta manera, se pretende el desarrollo integral del saber conocer, saber hacer, saber ser y saber estar de los estudiantes, para su continuo mejoramiento.

12. Recomendaciones para el estudio

- Asistir a todas las clases.
- Hacer las actividades de clase, dentro o fuera de ella.
- Estudiar regularmente del material de la cátedra y del provisto por los docentes.
- Utilizar los foros habilitados en la UV (Moodle).
- Consultar a los docentes cualquier duda, inquietud o necesidad en particular.
- Reunirse con pares para estudiar y/o realizar los trabajos prácticos integradores.
- Hacer las instancias de evaluación.
- Aprovechar el detalle de las devoluciones de las evaluaciones.

13. Metodología de evaluación

El modelo de enseñanza basado en competencias implica la aplicación de metodologías e instrumentos de evaluación que permiten conocer, a docentes y estudiantes, el nivel de desarrollo de las competencias que aborda la asignatura.

Evaluaciones

Todos los estudiantes son evaluados en diferentes instancias de evaluación

- Evaluaciones diagnósticas:

  Previo al inicio del cursado y de cada unidad temática se realiza un diagnóstico en forma oral con la participación activa de los estudiantes, en donde se validen los saberes ya incorporados y que son necesarios para continuar con el desarrollo de la asignatura.

- Evaluaciones sumativas parciales:

  Seis instancias basadas en cuestionarios de múltiple opción en donde cada pregunta tiene asignada una cantidad de puntaje igualitario, se utiliza el aula virtual (plataforma Moodle) para realizarla.

  Dos instancias basadas en un caso de estudio que plantee una situación problemática que el estudiante resuelve, según el paradigma que se evalúe, con los conceptos aprendidos y la práctica realizada. También se le provee una rúbrica de cada una de estas instancias de evaluación para que pueda ponderar el impacto que tendrá en la sumatoria de puntaje el realizar en forma correcta el ítem solicitado.

- Evaluaciones integradoras de resolución de problemas:

  El estudiante deberá desarrollar y presentar en tiempo y forma de manera grupal 3(tres) prácticos integradores uno por cada uno de los paradigmas: orientado a objetos, lógico y funcional. En donde desarrollarán soluciones a problemas situacionales, debiendo entregar la codificación en el lenguaje asociado para su evaluación.

A continuación, se detallan todos los resultados de aprendizajes con sus contenidos a desarrollar para alcanzarlos, la mediación pedagógica, metodologías y estrategias de evaluación, tiempo en horas reloj.

Resultados de Aprendizaje	Contenidos según programa	Mediación Pedagógica	Metodología y Estrategias de Evaluación	Tiempos en hora reloj
RA1	Conceptos generales: Programas, paradigmas, lenguajes de programación. Paradigmas fundamentales: Clasificación y evolución histórica, definición, lenguajes asociados, ventajas, limitaciones y áreas de aplicación. Diferencia entre lenguaje y paradigma de programación. Lenguajes de programación: Conceptos, criterios de evaluación, reseña histórica y evolución y tipos de lenguajes: híbridos y puros. Conceptos lógicos y transversales. Tipos	**Estrategia Docente**: Lección magistral participativa. Preguntas exploratorias. Presentación de problemas típicos factibles de resolver con cada paradigma. Foros de debate virtual. **Actividades**: Interacción entre otros estudiantes y el docente a través de la formulación de preguntas, desarrollo de una construcción común de conceptos durante las clases. Resolución en forma individual de cuestionarios.	**Instrumentos:** Evaluación diagnóstica de contenidos previos. Dos cuestionarios de múltiple opción. Cada cuestionario es de múltiple opción en donde cada pregunta tiene asignada una cantidad de puntaje igualitario, se utiliza el aula virtual (plataforma Moodle) para resolverlo en máquina. Al finalizar se entrega un reporte con la nota obtenida por el estudiante.	Total de horas presenciales teórico/prácticas: 12 Total de horas extra áulicas: 6

	de datos: Teoría, clasificación, verificación, sistema de tipos, conversión y ejemplos en los diferentes lenguajes de programación. Mecanismos de control de flujo: Organización y ejemplos en los diferentes lenguajes de programación. Abstracción y modularización: definición y mecanismos de implementación.	Reconocimiento de casos de estudios compatibles de resolver con cada paradigma. Lectura y comprensión de los apuntes de cátedra. Identificar los entornos de desarrollo para cada uno de los paradigmas de programación. Consulta y participación en foros y en clases.	**Criterios:** • Reconoce las características de cada uno de los paradigmas de programación. • Asocia los diferentes problemas factibles de resolver con cada uno de los diferentes paradigmas de programación. • Reconoce los elementos constitutivos, mecanismos y formas de implementación de las características de los lenguajes de programación dentro de los paradigmas de	

			programación correspondientes.	
RA2	Paradigma de orientado a Objetos: Conceptos fundamentales. Abstracción de datos y ocultamiento de la información. Estructura de un objeto. Métodos y mensajes. Clasificación. Clase. Concepto de generalización-especialización. Composición Herencia: Estrategias y modelos. Polimorfismo. Polimorfismo: Definición, tipos. Conceptos del modelo de objetos en SmallTalk. Desarrollo de la sintaxis de objetos en Smalltalk. Expresiones literales. Caracteres, secuencia de	**Estrategia Docente**: Lección magistral participativa. Preguntas exploratorias Resolución de ejercicios Formación experimental en laboratorios de acceso local. Aprendizaje cooperativo en grupos pequeños. Foros de debate virtual. **Actividades:** Participación de la lección magistral: atiende, realiza preguntas, toma notas. Resolución de ejercicios definidos en una Guía de Actividades, en forma individual y/o grupal empleando el	**Instrumentos:** Evaluación diagnóstica de contenidos previos. Trabajos a resolver por los estudiantes en forma grupal (Trabajo Práctico Integrador). Dos cuestionarios individuales de múltiple opción. Se utiliza el aula virtual (plataforma Moodle) para resolverlo en máquina. Una evaluación parcial individual que consiste en el desarrollo de una solución de un caso de estudio donde se le especifican los	Total de horas presenciales teórico/prácticas: 42 Total de horas extra áulicas: 24

	caracteres, símbolos y números. Expresiones de asignación y variables. Asignación. Tipos de variables. Variables privadas: de instancia nombradas e indexadas, argumentos y temporales. Variables compartidas: de clase, globales y pool. Variables especiales: self y super. 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. Expresiones de bloque. Clase Context. Bloques con y sin argumentos. Evaluación de bloques. Métodos y	lenguaje de Programación Orientado a Objetos: Smalltalk-Pharo. Trabajos colaborativos en grupos pequeños, necesarios para el aprendizaje cooperativo (normas de trabajo, formas de comunicación, etc.). Lectura y comprensión de los apuntes de cátedra. Consulta y participación en foros y en clases. .	requerimientos con un modelo de clases ya diseñado. Se le provee una rúbrica de cada instancia de evaluación. **Criterios:** • Codifica todas las clases involucradas en el dominio del problema, y asignación de atributos y responsabilidades de cada una. •Reutiliza los comportamientos implementados. •Usa la invocación de mensajes. •Elige y utiliza las colecciones en cada problema.	

	expresiones de retorno. Sintaxis de la definición de un método. Significado de la expresión de retorno. Métodos de clase e instancia. Implementación de Composición en Smalltalk. Implementación de Herencia en Smalltalk (definición de una Subclase, uso de super, herencia de variables, inicialización de atributos en una clase Hija, herencia de métodos, clases abstractas). Implementación de Polimorfismo en Smalltalk. Colecciones en Smalltalk: Introducción, jerarquía, colecciones básicas (Set, Bag, OrderedCollection, SortedCollection, Array, Dictionary), operaciones		•Identifica y usa métodos polimórficos. • Genera todas las salidas de información requeridas. • Realiza las pruebas de los comportamientos de los objetos.	

	básicas, conversión entre colecciones.			
RA3	Presentación del paradigma. Historia. Características. Ventajas/Desventajas. Áreas de Aplicación. Ejemplos de implementaciones. Familia de Lenguajes. Conceptos generales: Funciones matemáticas, Sintaxis en el paradigma funcional. Abstracción Funcional. Funciones de orden superior. Cálculo Lambda, evaluación postergada. Lenguaje Haskell. Introducción. Entorno de Haskell – HUGS. Sintaxis. Comentarios. Tipos de datos. Sistemas de inferencia de tipos. Flujo de control. Definición de Funciones.	**Estrategia Docente**: Lección magistral participativa. Preguntas exploratorias Resolución de ejercicios Formación experimental en laboratorios de acceso local. Aprendizaje cooperativo en grupos pequeños. Foros de debate virtual. **Actividades:** Participación de la lección magistral: atiende, realiza preguntas, toma notas. Resolución de ejercicios definidos en una Guía de Actividades, en forma individual y/o empleando el lenguaje de	**Instrumentos:** Evaluación diagnóstica de contenidos previos. Trabajos a resolver por los estudiantes en forma grupal (Trabajo Práctico Integrador). Un cuestionario individual de múltiple opción. Se utiliza el aula virtual (plataforma Moodle) para resolverlo en máquina. Una evaluación parcial individual que consiste en el desarrollo de una solución de un caso de estudio donde se le especifican los requerimientos con un	Total de horas presenciales teórico/prácticas: 21 Total de horas extra áulicas: 12

	Currificación. Reducción de expresiones. Evaluación. Tuplas. Recursividad. Listas: definición, listas por comprensión. Tipos definidos por el usuario. Tipos polimórficos.	Programación Funcional: Haskel - WinHugs. Trabajos colaborativos en grupos pequeños, necesarios para el aprendizaje cooperativo (normas de trabajo, formas de comunicación, etc.). Lectura y comprensión de los apuntes de cátedra. Consulta y participación en foros y en clases. .	modelo de clases ya diseñado. Se le provee una rúbrica de cada instancia de evaluación. **Criterios:** • Define funciones y tipos. • Codifica expresiones en el lenguaje funcional. • Aplica reutilización y composición de funciones. • Gestiona y usa listas y tuplas. • Define y utiliza funciones recursivas. • Codifica respetando las convenciones léxicas del lenguaje funcional.	

RA4	Presentación del paradigma lógico. Fundamentación lógica. Predicados y términos. 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. Interpretación algorítmica: Procedimientos y programación. Intérprete no determinista. Estrategia de evaluación. PROLOG Intérprete determinístico, "backtracking". Orden de evaluación de cláusulas. Terminación. Sintaxis PROLOG. Cláusulas, predicados y términos.	**Estrategia Docente**: Lección magistral participativa. Preguntas exploratorias Resolución de ejercicios Formación experimental en laboratorios de acceso local. Aprendizaje cooperativo en grupos pequeños. Foros de debate virtual. **Actividades:** Participación de la lección magistral: atiende, realiza preguntas, toma notas. Resolución de ejercicios definidos en una Guía de Actividades, en forma individual y/o empleando el lenguaje de Programación Lógica: Prolog. Trabajo colaborativo en grupos pequeños, necesarios para el	**Instrumentos:** Evaluación diagnóstica de contenidos previos. Trabajos a resolver por los estudiantes en forma grupal (Trabajo Práctico Integrador). Un cuestionario individual de múltiple opción. Se utiliza el aula virtual (plataforma Moodle) para resolverlo en máquina. Una evaluación parcial individual que consiste en el desarrollo de una solución de un caso de estudio donde se le especifican los requerimientos con un modelo de clases ya diseñado.	Total de horas presenciales teórico/prácticas: 21 Total de horas extra áulicas: 12

	Distintos tipos de datos. Recursión en PROLOG. Tipos de datos recursivos, lista. Concepto de variable o incógnita. Unificación. Múltiples resultados. Inversibilidad. Aritmética, evaluación de expresiones aritméticas. Negación. Functores.	aprendizaje cooperativo (normas de trabajo, formas de comunicación, etc.). Lectura y comprensión de los apuntes de cátedra. Consulta y participación en foros y en clases.	Se le provee una rúbrica de cada instancia de evaluación. **Criterios:** • Asigna identificadores y argumentos para predicados simples y compuestos. • Define todos los hechos necesarios para representar la base de conocimiento. • Define las reglas lógicas. • Utiliza expresiones propias del lenguaje lógico. • Especifica los objetivos solicitados.	

			• Codifica respetando las convenciones léxicas del lenguaje lógica.	

14. Condiciones de aprobación

Condiciones de aprobación

Para la aprobación de las instancias de evaluación se utiliza la siguiente escala de notas de regularidad(*)

NOTAS	PORCENTAJE	CALIFICACIÓN
1		No Aprobado
2		No Aprobado
3		No Aprobado
4	55% a 57%	Aprobado
5	58% a 59%	Aprobado
6	60% a 68%	Aprobado
7	69% a 77%	Aprobado
8	78% a 86%	Aprobado
9	87% a 95%	Aprobado
10	96% a 100%	Aprobado

(*) Escala acordada en reunión de Docentes Coordinadores

Requisitos de Regularización

•Aprobar dos evaluaciones parciales individuales en máquina, con una nota de 4 (cuatro) o superior. Se tendrá la opción de recuperación de cada evaluación parcial individual para los casos en que el estudiante no los haya aprobado o el mismo desee rendir para subir la nota obtenida.

•Aprobar los 6 cuestionarios de múltiple opción individuales en máquina, con una nota de 4 (cuatro) o superior, los cuales se reflejarán en el estado académico final con dos notas promedios que surgirán del promedio de tres cuestionarios: la primera nota abarca los cuestionarios 1, 2 y 3; la segunda nota abarca los cuestionarios 4, 5 y 6. Se tendrá la opción de recuperación de cada uno de los cuestionarios. Ésta opción de recuperación será para los casos en que el estudiante no haya alcanzado la aprobación de alguna de las notas o en el que el estudiante decida rendir el recuperatorio para subir la nota obtenida en estas instancias.

•Aprobar los tres trabajos prácticos integradores en tiempo y forma, o en su instancia de recuperación respectiva, con una nota de 4 (cuatro) o superior. Cada trabajo práctico integrador será resuelto en los siguientes paradigmas: orientado a objetos, funcional y lógico. El vencimiento de cada uno de los prácticos Integradores, junto con su opción de recuperación se consideran como plazos perentorios. Vencido el pazo de recuperación de cado uno de los prácticos integradores y no habiendo obtenido su aprobación, el estudiante, no podrá acceder a las opciones de regularización de la asignatura.

Requisitos de Aprobación directa

El estudiante aprueba la materia directamente, sin tener que rendir el examen final.

•Aprobar dos evaluaciones parciales individuales en máquina, con una nota de 6 (seis) o superior. Se tendrá la opción de recuperación de cada evaluación parcial individual para los casos en que el estudiante no los haya aprobado o el mismo desee rendir para subir la nota obtenida.

•Aprobar los 6 cuestionarios de múltiple opción individuales en máquina, con una nota de 6 (seis) o superior, los cuales se reflejarán en el estado académico final con dos notas promedios que surgirán del promedio de tres cuestionarios: la primera nota abarca los cuestionarios 1, 2 y 3; la segunda nota abarca los cuestionarios 4, 5 y 6. Se tendrá la opción de recuperación de cada uno de los cuestionarios. Ésta opción de recuperación será para los casos en que el estudiante no haya alcanzado la aprobación de alguna de las notas o en el que el estudiante decida rendir el recuperatorio para subir la nota obtenida en estas instancias.

•Aprobar los tres trabajos prácticos integradores en tiempo y forma, o en su instancia de recuperación respectiva, con una nota de 6 (seis) o superior. Cada trabajo práctico integrador será resuelto en los siguientes paradigmas: orientado a objetos, funcional y lógico. El vencimiento de cada uno de los prácticos Integradores, junto con su opción de recuperación se consideran como plazos perentorios. Vencido el plazo de recuperación de cado uno de los prácticos integradores y no habiendo obtenido su aprobación, el estudiante, no podrá acceder a las opciones de regularización de la asignatura.

La calificación se obtendrá con el promedio de las notas obtenidas por el estudiante y será registrada como Nota Final en Autogestión.

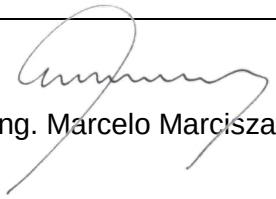
15. Modalidad de examen

Escala de Notas para Examen Final

NOTA	PORCENTAJE	CALIFICACIÓN
1		Insuficiente
2		Insuficiente
3		Insuficiente
4		Insuficiente
5		Insuficiente
6	60% a 68%	Aprobado
7	69% a 77%	Bueno
8	78% a 86%	Muy Bueno
9	87% a 95%	Distinguido
10	96% a 100%	Sobresaliente

El examen final se tomará en un único horario y será unificado para toda la cátedra.
El examen final para estudiantes en ***condición de regular***, comienza con una instancia de resolución de problema en máquina, la misma consta de evaluación de una solución de un problema determinado, codificado y resuelto, el cual será de carácter eliminatorio.
Los estudiantes que aprueben esta instancia de evaluación, pasarán a una próxima instancia de evaluación que se corresponde con un coloquio en donde se le brindan enunciados de preguntas que el estudiante deberá responder en forma satisfactoria.

16. Recursos necesarios
Espacios Físicos (aulas, laboratorios, equipamiento informático, etc.). Se requiere contar con aulas conforme a la cantidad de estudiantes inscriptos, con iluminación adecuada, conectividad wifi, enchufes para la alimentación de notebooks y cañones de proyección y computadoras acorde al software necesario. Recursos tecnológicos de apoyo (proyector multimedia, software, equipo de sonido, aulas virtuales, etc.). Se requiere contar con cañones de proyección en cada aula de laboratorio y con conectividad a internet (física o vía wifi). Plataforma Moodle y el software conforme a los lenguajes dados en la materia. En el caso de no contar con cañones se requiere VNC(Virtual Network Computing).



Ing. Marcelo Marciszack

FIRMA (Jefe o encargado de cátedra).