	Carrera/ Plan:
SEMINARIO DE LENGUAJES (OPCIÓN RUST)	*Licenciatura en Informática* Plan 2021/Plan 2015 *Licenciatura en Sistemas* Plan 2021/Plan 2015 *Analista Programador Universitario* Plan 2021/Plan 2015 *Analista en Tecnología de la Información y la Comunicación* Plan 2021/Plan 2017
	Año: 2
	Régimen de Cursada: *Semestral*
	Carácter (Obligatoria/Optativa): *Obligatoria*
Año 2026	**Correlativas**: *Taller de Programación*
	Profesor/es: ***Javier Díaz - Emanuel Borda***
	Hs. semanales teoría: 4 hs
	Hs. semanales práctica: 4hs

FUNDAMENTACIÓN

Dentro del marco de la materia genérica Seminario de Lenguajes, esta opción introduce el lenguaje Rust, un lenguaje de programación de propósito general, multiparadigma y multiplataforma. Este lenguaje aporta al aprendizaje del alumno y refuerza los conceptos vistos hasta el momento en la carrera. Es un lenguaje moderno y seguro, con amplio crecimiento en su uso en los últimos años.

OBJETIVOS GENERALES

Profundizar los conocimientos obtenidos por el alumno en los primeros cursos vinculados con Algoritmos y Programación, permitiéndole desarrollar un estudio teórico-práctico de algún lenguaje de programación (el lenguaje puede variar con los cambios tecnológicos), poniendo énfasis en el análisis formal de las características del lenguaje y su comparación con los que el alumno conociera a ese momento.

RESULTADOS DE APRENDIZAJE

1.3. Describir los avances informáticos actuales e históricos y demostrar cierta visión sobre tendencias y avances futuros (Básico).

3.1. Definir y diseñar hardware/software informático/de red que cumpla con los requisitos establecidos (Básico).

3.3. Elegir y utilizar modelos de proceso adecuados, entornos de programación y técnicas de gestión de datos con respecto a proyectos que impliquen aplicaciones tradicionales así como aplicaciones emergentes (Básico).

3.4. Describir y explicar el diseño de sistemas e interfaces para interacción persona-ordenador y ordenador-ordenador (Básico).

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

6.1. Organizar su propio trabajo de manera independiente demostrando iniciativa y ejerciendo responsabilidad personal (Básico).

6.3. Planificar su propio proceso de aprendizaje autodidacta y mejorar su rendimiento personal como base de una formación y un desarrollo personal continuos (Básico).

COMPETENCIAS

CGS2- Comunicarse con efectividad en forma oral y escrita.

CGS4- Aprender en forma continua y autónoma, con capacidad de planificar este aprendizaje.

CGS6- Capacidad para interpretar la evolución de la informática con una visión de las tendencias tecnológicas futuras.

CGT1- Identificar, formular y resolver problemas de Informática.

CGT5- Utilizar de manera efectiva las técnicas y herramientas de aplicación de la Informática.

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 MINIMOS (de acuerdo al Plan de Estudios)

Estudio de un lenguaje de programación emergente, con el que se desarrollan aplicaciones concretas, seguras y de alta performance. Fundamentos del lenguaje, estructuras de control, tipos de datos, manejo de memoria, objetos en Rust. Implementación de test.

PROGRAMA ANALÍTICO

Unidad 1 -Historia. Conceptos básicos. Instalación. Variables. Inmutabilidad. Constantes

Unidad 2- Estructuras de control: condicionales, iterativas y operadores lógicos. Tipos de datos básicos.

Unidad 3- Tipos de datos complejos: Collections, Struct, Enum, Option

Unidad 4- Funciones, Datos genéricos. Traits y ciclo de vida.

Unidad 5- Concepto de ownership y borrowing. Manejo de errores.

Unidad 6- Iterators, Closures, Prelude

Unidad 7- Escritura de tests. Archivos.

Unidad 8- Smart pointers, crates

Unidad 9- Clases y objetos en Rust.

BIBLIOGRAFÍA

- Rust. https://doc.rust-lang.org/stable/book/
- Programming Rust, 2nd Edition, 2021 ISBN 9781492052593
- Beginning Rust: Get Started with Rust 2021 Edition 2nd ed. Edición, ISBN:978-1484272077
- Hands-On Data Structures and Algorithms with Rust ISBN: 978-1788995528
- Rust Essentials - Second Edition ISBN: 978-1788390019
- Rust in Action 1st Edición, Tim McNamara. ISBN-13: 978-1617294556

METODOLOGÍA DE ENSEÑANZA

La asignatura es de tipo taller, la teoría y práctica se encuentran estrechamente vinculadas. Las clases teóricas, donde se desarrollan los aspectos conceptuales del lenguaje, se dictan utilizando presentaciones del tipo slideshow. Se incluyen ejemplos de código para resolver distintas situaciones problemáticas. Estas clases no son obligatorias. En las clases prácticas se explicará un tema y un problema particular, acorde al cronograma de la materia, donde los estudiantes deberán resolver de manera individual o manera grupal dependiendo de la consigna. Las clases prácticas se realizan en la sala de PC de la facultad.

Esta metodología se complementa con la plataforma virtual Moodle, donde los estudiantes deberán realizar entregas específicas de resoluciones de ejercicios prácticos de carácter obligatorio.

Se utilizará también una herramienta para el seguimiento de la cátedra, de la asistencia de los alumnos a las prácticas y del aprovechamiento de cada práctica.

La materia reconoce que las herramientas de Inteligencia Artificial Generativa (IAG) forman parte del entorno actual de desarrollo de software. En consecuencia, se adopta un enfoque de uso responsable y explícito.

El estudiante es siempre responsable intelectual y técnicamente del código entregado. La evaluación contempla no sólo el funcionamiento del programa sino también la comprensión conceptual y las decisiones de diseño adoptadas.

Se considera:

Uso permitido:

- *Consulta conceptual.*
- *Análisis de errores del compilador.*
- *Generación de ejemplos o casos de prueba.*

Uso no permitido:

- *Generación íntegra de soluciones evaluables.*
- *Copia de código sin comprensión.*
- *Uso en instancias evaluativas individuales sin autorización explícita.*

El uso parcial de IAG deberá ser declarado y podrá requerir defensa oral para verificar comprensión.

EVALUACIÓN

Durante la cursada a medida que se desarrollen los contenidos los alumnos deberán entregar 3 ejercicios prácticos/actividades de manera individual donde en cada uno de ellos se obtendrá una nota, a su vez al finalizar el curso se deberá realizar la entrega de un trabajo integrador final grupal donde se desarrolle los conceptos explicados del lenguaje . El promedio de dichas notas será la nota final de la materia. Además se solicita que realicen todos los cuestionarios en moodle referentes a clases teóricas.
Toda entrega deberá incluir una declaración explícita sobre el uso de herramientas de Inteligencia Artificial Generativa (IAG), indicando si se utilizaron y, en caso afirmativo, una breve descripción del tipo de asistencia recibida. La omisión de esta información podrá considerarse falta académica. La evaluación contemplará no sólo el correcto funcionamiento del código sino también la adecuada aplicación de los conceptos del lenguaje, la justificación de las decisiones de diseño adoptadas y la capacidad del estudiante para explicar y fundamentar su solución. La cátedra podrá requerir instancias de defensa individual de cualquier entrega a efectos de verificar comprensión.

CRONOGRAMA DE CLASES Y EVALUACIONES TENTATIVO

Semana	Temas	Práctica
1 (23/3)	Introducción a Rust: historia, conceptos básicos, instalación. Variables, inmutabilidad, constantes.	1
2(30/3)	Estructuras de control: condicionales, iterativas y operadores lógicos. Tipos de datos básicos.	2
3(6/4)	Colecciones. Funciones.	2
4(13/4)	Struct. Enum.	3
5(20/4)	Option. Datos genéricos.	3
6(27/4)	Traits. Ciclo de vida.	4
7(4/5)	Ownership y borrowing. Manejo de errores.	4
8(11/5)	Closures. Iterators.	5
9(18/5)	Prelude. Escritura de tests.	5
10(1/6)	Archivos. Smart pointers.	6
11(8/6)	Crates. Clases y objetos en Rust.	6
12(15/6)	Explicación para el trabajo final parte I.	
13(22/6)	Explicación para el trabajo final parte II.	
14(29/6)		
15(6/7)		
16(13/7)	Entrega y exposición del trabajo final.	
17(3/8)	Reentrega del trabajo final.	

- **Contacto de la cátedra (mail, sitio WEB, plataforma virtual de gestión de cursos):**
  - **Plataforma virtual: https://catedras.info.unlp.edu.ar/**
  - **contacto: rust@linti.unlp.edu.ar**