

**SEMINARIO DE LENGUAJES
(OPCION .NET)**

Año 2025

Carrera/ Plan:

Licenciatura en Informática Plan 2021/Plan 2015
Licenciatura en Sistemas Plan 2021/Plan 2015
Analista Programador Universitario Plan 2021/Plan 2015
Analista en Tecnologías de la Información y la Comunicación
Plan 2021/Plan 2017

Año: 2°**Régimen de Cursada:** *Semestral***Carácter:** *Electiva***Correlativas:** *Taller de Programación.***Profesor/es:** *Leonardo Corbalan***Hs. semanales teoría:** 3**Hs. semanales práctica:** 3**FUNDAMENTACIÓN**

La plataforma .NET proporciona un modelo de programación coherente e independiente del lenguaje para todas las capas o niveles de una aplicación. Se utiliza para la creación de todo tipo de aplicaciones: web, para dispositivos móviles, escritorio, juegos, IoT entre otras. Se admite en Windows, Linux y macOS. El lenguaje de programación C#, diseñado especialmente para la plataforma .NET, es un lenguaje moderno que se actualiza constantemente y se encuentra entre los más utilizados por la comunidad de desarrollo de software actual.

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 un lenguaje de programación soportado por la plataforma .NET, 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 en el que se desarrollen aplicaciones concretas. En lo posible la oferta de lenguajes será variable y actualizada con el cambio tecnológico.

PROGRAMA ANALÍTICO

Unidad 1: Plataforma .NET: Características. Common Language Runtime (CLR). Microsoft Intermediate Language (MSIL). Compilador Just-In-Time (JIT). Common Type System (CTS). Base Classes Library (BCL).

Unidad 2: Lenguaje C#: Características del lenguaje: Estructuras de control. Tipos básicos. Operadores. Ámbito de las variables. Métodos y parámetros. Sistema unificado de tipos: Tipos de valor y tipos de referencia. Conversiones de tipo explícitas e implícitas. Conversiones *boxing* y *unboxing*. Manejo de *strings*, arreglos y colecciones. Formatos compuestos e interpolación de strings. Manejo estructurado de excepciones.

Unidad 3: Programación orientada a objetos con C#: Clases y objetos. Campos y métodos de instancia. Constructores y finalizadores. Sobrecarga de métodos y de constructores. Miembros estáticos (de clase). Encapsulamiento. Propiedades e Indizadores. Herencia. Invalidación de métodos. Modificadores de acceso. Polimorfismo. Principio Open/Close. Interfaces. Implementación de múltiples interfaces. Implementación explícita de miembros de interfaces. Uso de interfaces conocidas: IComparable, IComparer, IEnumerable e IEnumerator de la plataforma .NET. Iteradores.

Unidad 4: Conceptos avanzados del lenguaje C# y la plataforma .NET: Delegados. Métodos anónimos y expresiones lambda. Uso de delegados para implementar el pasaje de métodos por parámetro y la generación de eventos. Convenciones de nomenclatura para delegados y métodos involucrados en el lanzamiento y manejo de un evento. Métodos genéricos. Tipos genéricos (clases, interfaces y delegados genéricos). Programación asincrónica. Patrón asincrónico basado en Tareas. Métodos asincrónicos. Modificador `async` y operador `await`

Unidad 5. Persistencia de datos: Introducción a Entity Framework Core. LINQ. Operaciones CRUD sobre base de datos relacionales.

Unidad 6. Aplicaciones Web: Introducción al desarrollo de aplicaciones web con ASP.NET Core. Inyección de dependencias. Uso de contenedores DI. Desarrollo de aplicación web de página única (SPA) con ASP.NET Core Blazor. Componentes Razor. Conceptos de Arquitectura Limpia.

BIBLIOGRAFÍA

No se utiliza bibliografía obligatoria. Los contenidos publicados por la cátedra en la plataforma IDEAS cubren los requerimientos del curso. Sin embargo, se aconseja adquirir el hábito de consultar la documentación de Microsoft para usuarios finales, desarrolladores y profesionales de TI (<https://docs.microsoft.com/es-es/>) en relación con la plataforma .NET y al lenguaje C#.

BIBLIOGRAFÍA COMPLEMENTARIA

- Professional C# 7 and .NET Core 2.0, Christian Nagel, Wrox; Edición: 7. (2018).
- Illustrated C# 7: The C# Language Presented Clearly, Concisely, and Visually. Daniel M. Solis. Apress 5th ed. (2018).
- Dependency Injection in .NET Core 2.0, Marino Posadas y Tadish Dash. Packt Publishing (2017)
- ASP.NET Core in Action. Andrew Lock. Manning Publications; Edición: 1st (2018)
- Microsoft Blazor Building Web Applications in .NET. Second Edition. Peter Himschoot
- Microsoft documentation for end users, developers, and IT professionals. <https://docs.microsoft.com>

METODOLOGÍA DE ENSEÑANZA

La asignatura se organiza en clases teóricas, prácticas y de explicaciones de práctica. Estas actividades se desarrollan íntegramente en la sala de PC. La razón de ello es maximizar la interacción del alumno con el lenguaje, la plataforma y el ambiente de desarrollo, aún en las clases teóricas donde abundan ejercicios de codificación propuestos a los estudiantes con la intención de discutir y asimilar conceptos teóricos a partir de los resultados obtenidos. Durante las consultas de práctica, además de atender las dudas individuales de los alumnos, se ofrecen explicaciones generales sobre algunos de los ejercicios más representativos.

El material del curso se compone de 13 clases teóricas y 11 trabajos prácticos no entregables. Cada uno de los ejercicios publicados en los trabajos prácticos incluye algún concepto concreto que la cátedra pretende ilustrar. En consecuencia, es sumamente importante que los alumnos resuelvan todos los ejercicios y participen activamente en las consultas de práctica para despejar cualquier duda que les pueda sobrevenir.

Durante el desarrollo del curso, se plantea la realización de un trabajo grupal de desarrollo que constará de dos entregas: una parcial y una final. Para ambas entregas, es requisito que se acompañe un informe escrito que detalle los aspectos relevantes del trabajo. Se requiere que todos los integrantes del grupo conozcan los detalles del trabajo, los cuales serán evaluados en un coloquio individual hacia el final del curso. Una vez concluidas las clases teóricas, el horario reservado para las mismas es utilizado para consulta y para la realización de los coloquios.

Se utiliza la plataforma IDEAS para la publicación del material (contenido teórico, trabajos prácticos y trabajos de programación obligatorios) y la comunicación con los alumnos a través de la mensajería, cartelera de novedades y foro.

Aunque la orientación del curso es predominantemente práctica, no se desatienden aspectos teóricos importantes que el alumno debe conocer para comprender con claridad los conceptos inherentes a un lenguaje orientado a objetos como es el caso de C#.

El curso promueve el trabajo constante y la participación de los alumnos en clase. A partir de la segunda teoría, una vez concluidas las actividades relacionadas con los nuevos conceptos presentados, se discuten con la intervención activa de los estudiantes, los detalles más significativos de la práctica resuelta la semana anterior. Por lo tanto, es importante que los alumnos trabajen todas las semanas llevando al día la realización de las prácticas para así poder aprovechar esta instancia de fijación de conceptos.

Completan estas instancias de fijación y clarificación de conceptos una serie de autoevaluaciones realizadas a lo largo del curso. Estas autoevaluaciones son de carácter formativo, no se utilizan para examinar a los alumnos sino para que ellos mismos puedan valorar de qué forma están transitando el proceso de aprendizaje. Una autoevaluación consiste en una serie de ejercicios con preguntas y opciones de respuestas presentados a la clase de a uno por vez utilizando un proyector multimedia. Concluido el tiempo otorgado en cada ejercicio para que el alumno piense su respuesta se señala la opción correcta. Durante la ejecución de la prueba los estudiantes calculan su propio puntaje, información mantenida sólo para sí. Este espacio es utilizado también por la cátedra para despejar dudas y clarificar conceptos relacionados con los ejercicios presentados.

A continuación, se ofrecen precisiones sobre la forma en que se alcanzan y evalúan las competencias enunciadas previamente:

CGS2- Comunicarse con efectividad en forma oral y escrita

El informe escrito que los alumnos deben presentar, junto con los ejercicios de programación obligatorios, es utilizado para evaluar el contenido técnico, la estructura, organización, sintaxis y claridad conceptual. Además, estos trabajos deben exponerse en un coloquio ante la cátedra que evalúa los conocimientos adquiridos, forma de presentación y expresión de los alumnos. Los resultados se reflejan en planillas escritas que conforman documentación de evaluación del coloquio.

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

Se incluyen actividades planificadas para los alumnos proponiendo “desafíos” que deben convertirse en “ideas proyecto” y posteriormente en potenciales desarrollos del alumno. El objetivo es que el alumno logre abstraer una serie de pasos que respondan a una metodología clásica de investigación y lo ayuden a formarse en esta competencia:

- Búsqueda de bibliografía actualizada sobre el tema.
- Abstracción del desafío/problema como una “idea proyecto a resolver”.
- Expresión sintética de la especificación del proyecto, con recursos humanos requeridos y plan de tareas.
- Implementación y defensa oral/escrita de la solución al desafío.

La cátedra acompaña el proceso del alumno, para consolidar sus habilidades para esta competencia. La evaluación de esta competencia se refleja en una planilla detallada, donde se indica la capacidad del alumno para desarrollar su aprendizaje y la formulación de la solución al desafío en forma autónoma.

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

Se plantean actividades planificadas para los alumnos proponiendo el estudio de la tecnología existente y prevista para un tipo de problema y se los “desafía” a presentar la posible evolución de la solución para ese tipo de problema y en QUE podría mejorarse la solución/soluciones actuales.

Esto lleva al alumno a buscar bibliografía relacionada con el cambio tecnológico y formarse un criterio sobre las tendencias (por ejemplo, en los procesadores a utilizar, el tipo de topología de red o la migración de aplicaciones a móviles o el modo de extraer conocimiento de grandes volúmenes de datos).

La cátedra acompaña el proceso del alumno, para contrastar las conclusiones del alumno y validar su habilidad para esta competencia. La evaluación de esta competencia se refleja en una planilla detallada, donde se indican los resultados del estudio bibliográfico del alumno y su capacidad para formular las ventajas potenciales del cambio tecnológico en el problema/área de conocimiento planteada.

CGT1- Identificar, formular y resolver problemas de Informática.

En la cátedra se pone énfasis en el proceso de identificación de problemas del mundo real, especificación de los mismos como problemas resolubles desde la informática y en el desarrollo de soluciones verificables para los mismos.

La evaluación de esta competencia se realiza por medio de las evaluaciones de los informes escritos, trabajos de programación obligatorios, coloquios y examen final de la asignatura y se refleja en la corrección de las pruebas escritas del alumno.

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

En la cátedra se pone énfasis en la capacidad del alumno para conocer técnicas y herramientas de aplicación en Informática (en lo posible siguiendo las tendencias marcadas por el cambio tecnológico) y en la aplicación efectiva de las mismas.

La cátedra acompaña el proceso con materiales para que el alumno estudie casos y valore la selección y empleo eficiente de herramientas y técnicas determinadas para cada problema.

La evaluación de esta competencia se realiza por medio de las evaluaciones de los informes escritos, trabajos de programación obligatorios, coloquios y examen final de la asignatura y se refleja en la corrección de las pruebas escritas del alumno.

EVALUACIÓN

Son requisitos para aprobar la cursada:

- Cumplir con las entregas parciales que se soliciten y aprobar con nota mayor o igual a 6 (seis) la entrega final del trabajo de desarrollo integrador.
- Aprobar un coloquio final.

El trabajo integrador puede presentarse de manera individual o grupal. El coloquio es individual. La entrega final del trabajo integrador y el coloquio contarán con una instancia de recuperación en caso de ser necesario.

Régimen de promoción: El alumno que apruebe la cursada también obtendrá la promoción. La calificación final se definirá en función del resultado de las entregas y del coloquio.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	Semana del 10/03/2025	Teoría 1. Generalidades de la plataforma .NET. Introducción al lenguaje C#. Características. Tipos integrados. Variables, constantes, literales, operadores. Espacios de nombres. Estructuras de control.
2	Semana del 17/03/2025	Teoría 2. Sistema unificado de tipos. Tipos de valor y tipos de referencia. Conversiones de tipo explícitas e implícitas, <i>boxing</i> y <i>unboxing</i> . Métodos y parámetros. Pasaje de parámetros por la línea de comandos.
3	Semana del 31/03/2025	Teoría 3. Inferencia de tipos. Tipos anónimos. Tipo Dynamic. Formatos compuestos e interpolación de strings. Arreglos. Colecciones. Manejo de Excepciones.

4	Semana del 07/04/2025	Teoría 4. Conceptos introductorios de programación orientada a objetos. Clases. Miembros de instancia. Constructores. Sobrecarga de constructores y de métodos.
5	Semana del 14/04/2025	Teoría 5. Miembros estáticos. Encapsulamiento. Control de acceso a la representación interna de los objetos por medio de propiedades e indizadores.
6	Semana del 21/04/2025	Teoría 6. Herencia. Especialización de clases. Invalidación de métodos. Modificadores de acceso: public, protected, private, e internal. Polimorfismo. Principio Open/Close
7	Semana del 28/04/2025	Teoría 7. Interfaces. Implementación de múltiples interfaces. Implementación explícita de miembros de interfaces. Principio de inversión de dependencias (DIP) y Patrón de Inyección de Dependencias. Conceptos de Arquitectura Limpia.
8	Semana del 05/05/2025	Teoría 8. Uso de las interfaces IComparable, IComparer, IEnumerable e IEnumerator de la plataforma .NET. Iteradores. Delegados. Pasaje de métodos como parámetro. Métodos anónimos. Expresiones lambda.
9	Semana del 12/05/2025	Teoría 9. Métodos Genéricos. Tipos genéricos (Clases, interfaces y delegados).
10	Semana del 19/05/2025	Teoría 10. Persistencia de datos. SQLite. LINQ. ORM Entity Framework Core
11	Semana del 26/05/2025	Teoría 11. Contenedor de inyección de dependencias integrado en la plataforma .NET. Introducción al desarrollo de aplicaciones web con ASP.NET Core Blazor. Componente Razor.
12	Semana del 02/06/2025	Teoría 12. Desarrollo de aplicaciones web de página única (SPA por Single Page App), integrando conceptos de Arquitectura Limpia, persistencia de datos con Entity Framework y desarrollo de la interfaz de usuario con ASP.NET Core Blazor.
13	Semana del 09/06/2025	Teoría 13. Inicio a la Programación asincrónica. Patrón Asincrónico basado en Tareas (TAP). Métodos asincrónicos y tareas. Modificador async y operador await
		El resto de las clases se utiliza para consulta, corrección y devolución de trabajos y para la toma de coloquios.

Evaluaciones previstas	Fecha
Entrega final del trabajo integrador	16/06/2025



Coloquios

A partir del 30/06/2025 se asignarán turnos para rendir coloquio en los horarios de práctica y teoría

CONTACTO DE LA CÁTEDRA:

Mail: slpnet@lidi.info.unlp.edu.ar

Plataforma IDEAS (<https://ideas.info.unlp.edu.ar>) **Curso:** "Seminario de Lenguajes - Opción .NET 1er. Semestre 2025"

Firma del/los profesor/es

Leonardo Corbalán
Profesor Adjunto

**REDICTADO SEMINARIO DE
LENGUAJES
(OPCION .NET)**

Año 2025

Carrera/ Plan:

Licenciatura en Informática Plan 2021/Plan 2015
Licenciatura en Sistemas Plan 2021/Plan 2015
Analista Programador Universitario Plan 2021/Plan 2015
Analista en Tecnologías de la Información y la Comunicación
Plan 2021/Plan 2017

Año: 2°**Régimen de Cursada:** Semestral**Carácter:** Electiva**Correlativas:** Taller de Programación**Profesor/es:** Leonardo Corbalan**Hs. semanales teoría:** 3**Hs. semanales práctica:** 3**FUNDAMENTACIÓN**

La plataforma .NET proporciona un modelo de programación coherente e independiente del lenguaje para todas las capas o niveles de una aplicación. Se utiliza para la creación de todo tipo de aplicaciones: web, para dispositivos móviles, escritorio, juegos, IoT entre otras. Se admite en Windows, Linux y macOS. El lenguaje de programación C#, diseñado especialmente para la plataforma .NET, es un lenguaje moderno que se actualiza constantemente y se encuentra entre los más utilizados por la comunidad de desarrollo de software actual.

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 un lenguaje de programación soportado por la plataforma .NET, 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 en el que se desarrollen aplicaciones concretas. En lo posible la oferta de lenguajes será variable y actualizada con el cambio tecnológico.

PROGRAMA ANALÍTICO

Unidad 1: Plataforma .NET: Características. Common Language Runtime (CLR). Microsoft Intermediate Language (MSIL). Compilador Just-In-Time (JIT). Common Type System (CTS). Base Classes Library (BCL).

Unidad 2: Lenguaje C#: Características del lenguaje: Estructuras de control. Tipos básicos. Operadores. Ámbito de las variables. Métodos y parámetros. Sistema unificado de tipos: Tipos de valor y tipos de referencia. Conversiones de tipo explícitas e implícitas. Conversiones *boxing* y *unboxing*. Manejo de *strings*, arreglos y colecciones. Formatos compuestos e interpolación de strings. Manejo estructurado de excepciones.

Unidad 3: Programación orientada a objetos con C#: Clases y objetos. Campos y métodos de instancia. Constructores y finalizadores. Sobrecarga de métodos y de constructores. Miembros estáticos (de clase). Encapsulamiento. Propiedades e Indizadores. Herencia. Invalidación de métodos. Modificadores de acceso. Polimorfismo. Principio Open/Close. Interfaces. Implementación de múltiples interfaces. Implementación explícita de miembros de interfaces. Uso de interfaces conocidas: IComparable, IComparer, IEnumerable e IEnumerator de la plataforma .NET. Iteradores.

Unidad 4: Conceptos avanzados del lenguaje C# y la plataforma .NET: Delegados. Métodos anónimos y expresiones lambda. Uso de delegados para implementar el pasaje de métodos por parámetro y la generación de eventos. Convenciones de nomenclatura para delegados y métodos involucrados en el lanzamiento y manejo de un evento. Métodos genéricos. Tipos genéricos (clases, interfaces y delegados genéricos). Programación asincrónica. Patrón asincrónico basado en Tareas. Métodos asincrónicos. Modificador `async` y operador `await`

Unidad 5. Persistencia de datos: Introducción a Entity Framework Core. LINQ. Operaciones CRUD sobre base de datos relacionales.

Unidad 6. Aplicaciones Web: Introducción al desarrollo de aplicaciones web con ASP.NET Core. Inyección de dependencias. Uso de contenedores DI. Desarrollo de aplicación web de página única (SPA) con ASP.NET Core Blazor. Componentes Razor. Conceptos de Arquitectura Limpia.

BIBLIOGRAFÍA

No se utiliza bibliografía obligatoria. Los contenidos publicados por la cátedra en la plataforma IDEAS cubren los requerimientos del curso. Sin embargo, se aconseja adquirir el hábito de consultar la documentación de Microsoft para usuarios finales, desarrolladores y profesionales de TI (<https://docs.microsoft.com/es-es/>) en relación con la plataforma .NET y al lenguaje C#.

BIBLIOGRAFÍA COMPLEMENTARIA

- Professional C# 7 and .NET Core 2.0, Christian Nagel, Wrox; Edición: 7. (2018).
- Illustrated C# 7: The C# Language Presented Clearly, Concisely, and Visually. Daniel M. Solis. Apress 5th ed. (2018).
- Dependency Injection in .NET Core 2.0, Marino Posadas y Tadish Dash. Packt Publishing (2017)
- ASP.NET Core in Action. Andrew Lock. Manning Publications; Edición: 1st (2018)
- Microsoft Blazor Building Web Applications in .NET. Second Edition. Peter Himschoot
- Microsoft documentation for end users, developers, and IT professionals. <https://docs.microsoft.com>

METODOLOGÍA DE ENSEÑANZA

La asignatura se organiza en clases teóricas, prácticas y de explicaciones de práctica. Estas actividades se desarrollan íntegramente en la sala de PC. La razón de ello es maximizar la interacción del alumno con el lenguaje, la plataforma y el ambiente de desarrollo, aún en las clases teóricas donde abundan ejercicios de codificación propuestos a los estudiantes con la intención de discutir y asimilar conceptos teóricos a partir de los resultados obtenidos. Durante las consultas de práctica, además de atender las dudas individuales de los alumnos, se ofrecen explicaciones generales sobre algunos de los ejercicios más representativos.

El material del curso se compone de 13 clases teóricas y 11 trabajos prácticos no entregables. Cada uno de los ejercicios publicados en los trabajos prácticos incluye algún concepto concreto que la cátedra pretende ilustrar. En consecuencia, es sumamente importante que los alumnos resuelvan todos los ejercicios y participen activamente en las consultas de práctica para despejar cualquier duda que les pueda sobrevenir.

Durante el desarrollo del curso, se plantea la realización de un trabajo grupal de desarrollo que constará de dos entregas: una parcial y una final. Para ambas entregas, es requisito que se acompañe un informe escrito que detalle los aspectos relevantes del trabajo. Se requiere que todos los integrantes del grupo conozcan los detalles del trabajo, los cuales serán evaluados en un coloquio individual hacia el final del curso. Una vez concluidas las clases teóricas, el horario reservado para las mismas es utilizado para consulta y para la realización de los coloquios.

Se utiliza la plataforma IDEAS para la publicación del material (contenido teórico, trabajos prácticos y trabajos de programación obligatorios) y la comunicación con los alumnos a través de la mensajería, cartelera de novedades y foro.

Aunque la orientación del curso es predominantemente práctica, no se desatienden aspectos teóricos importantes que el alumno debe conocer para comprender con claridad los conceptos inherentes a un lenguaje orientado a objetos como es el caso de C#.

El curso promueve el trabajo constante y la participación de los alumnos en clase. A partir de la segunda teoría, una vez concluidas las actividades relacionadas con los nuevos conceptos presentados, se discuten con la intervención activa de los estudiantes, los detalles más significativos de la práctica resuelta la semana anterior. Por lo tanto, es importante que los alumnos trabajen todas las semanas llevando al día la realización de las prácticas para así poder aprovechar esta instancia de fijación de conceptos.

Completan estas instancias de fijación y clarificación de conceptos una serie de autoevaluaciones realizadas a lo largo del curso. Estas autoevaluaciones son de carácter formativo, no se utilizan para examinar a los alumnos sino para que ellos mismos puedan valorar de qué forma están transitando el proceso de

aprendizaje. Una autoevaluación consiste en una serie de ejercicios con preguntas y opciones de respuestas presentados a la clase de a uno por vez utilizando un proyector multimedia. Concluido el tiempo otorgado en cada ejercicio para que el alumno piense su respuesta se señala la opción correcta. Durante la ejecución de la prueba los estudiantes calculan su propio puntaje, información mantenida sólo para sí. Este espacio es utilizado también por la cátedra para despejar dudas y clarificar conceptos relacionados con los ejercicios presentados.

A continuación, se ofrecen precisiones sobre la forma en que se alcanzan y evalúan las competencias enunciadas previamente:

CGS2- Comunicarse con efectividad en forma oral y escrita

El informe escrito que los alumnos deben presentar, junto con los ejercicios de programación obligatorios, es utilizado para evaluar el contenido técnico, la estructura, organización, sintaxis y claridad conceptual. Además, estos trabajos deben exponerse en un coloquio ante la cátedra que evalúa los conocimientos adquiridos, forma de presentación y expresión de los alumnos. Los resultados se reflejan en planillas escritas que conforman documentación de evaluación del coloquio.

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

Se incluyen actividades planificadas para los alumnos proponiendo “desafíos” que deben convertirse en “ideas proyecto” y posteriormente en potenciales desarrollos del alumno. El objetivo es que el alumno logre abstraer una serie de pasos que respondan a una metodología clásica de investigación y lo ayuden a formarse en esta competencia:

- Búsqueda de bibliografía actualizada sobre el tema.
- Abstracción del desafío/problema como una “idea proyecto a resolver”.
- Expresión sintética de la especificación del proyecto, con recursos humanos requeridos y plan de tareas.
- Implementación y defensa oral/escrita de la solución al desafío.

La cátedra acompaña el proceso del alumno, para consolidar sus habilidades para esta competencia. La evaluación de esta competencia se refleja en una planilla detallada, donde se indica la capacidad del alumno para desarrollar su aprendizaje y la formulación de la solución al desafío en forma autónoma.

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

Se plantean actividades planificadas para los alumnos proponiendo el estudio de la tecnología existente y prevista para un tipo de problema y se los “desafía” a presentar la posible evolución de la solución para ese tipo de problema y en QUE podría mejorarse la solución/soluciones actuales.

Esto lleva al alumno a buscar bibliografía relacionada con el cambio tecnológico y formarse un criterio sobre las tendencias (por ejemplo, en los procesadores a utilizar, el tipo de topología de red o la migración de aplicaciones a móviles o el modo de extraer conocimiento de grandes volúmenes de datos).

La cátedra acompaña el proceso del alumno, para contrastar las conclusiones del alumno y validar su habilidad para esta competencia. La evaluación de esta competencia se refleja en una planilla detallada, donde se indican los resultados del estudio bibliográfico del alumno y su capacidad para formular las ventajas potenciales del cambio tecnológico en el problema/área de conocimiento planteada.

CGT1- Identificar, formular y resolver problemas de Informática.

En la cátedra se pone énfasis en el proceso de identificación de problemas del mundo real, especificación de los mismos como problemas resolubles desde la informática y en el desarrollo de soluciones verificables para los mismos.

La evaluación de esta competencia se realiza por medio de las evaluaciones de los informes escritos, trabajos de programación obligatorios, coloquios y examen final de la asignatura y se refleja en la corrección de las pruebas escritas del alumno.

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

En la cátedra se pone énfasis en la capacidad del alumno para conocer técnicas y herramientas de aplicación en Informática (en lo posible siguiendo las tendencias marcadas por el cambio tecnológico) y en la aplicación efectiva de las mismas.

La cátedra acompaña el proceso con materiales para que el alumno estudie casos y valore la selección y empleo eficiente de herramientas y técnicas determinadas para cada problema.

La evaluación de esta competencia se realiza por medio de las evaluaciones de los informes escritos, trabajos de programación obligatorios, coloquios y examen final de la asignatura y se refleja en la corrección de las pruebas escritas del alumno.

EVALUACIÓN

Son requisitos para aprobar la cursada:

- Cumplir con las entregas parciales que se soliciten y aprobar con nota mayor o igual a 6 (seis) la entrega final del trabajo de desarrollo integrador.
- Aprobar un coloquio final.

El trabajo integrador puede presentarse de manera individual o grupal. El coloquio es individual. La entrega final del trabajo integrador y el coloquio contarán con una instancia de recuperación en caso de ser necesario.

Régimen de promoción: El alumno que apruebe la cursada también obtendrá la promoción. La calificación final se definirá en función del resultado del trabajo integrador y del coloquio.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Fecha	Contenidos/Actividades
1	19/08/2025	Teoría 1. Generalidades de la plataforma .NET. Introducción al lenguaje C#. Características. Tipos integrados. Variables, constantes, literales, operadores. Espacios de nombres. Estructuras de control.
2	26/08/2025	Teoría 2. Sistema unificado de tipos. Tipos de valor y tipos de referencia. Conversiones de tipo explícitas e implícitas, <i>boxing</i> y <i>unboxing</i> . Métodos y parámetros. Pasaje de parámetros por la línea de comandos.
3	02/09/2025	Teoría 3. Inferencia de tipos. Tipos anónimos. Tipo Dynamic. Formatos compuestos e interpolación de strings. Arreglos. Colecciones. Manejo de Excepciones.
4	09/09/2025	Teoría 4. Conceptos introductorios de programación orientada a objetos. Clases. Miembros de instancia. Constructores. Sobrecarga de constructores y de métodos.

5	16/09/2025	Teoría 5. Miembros estáticos. Encapsulamiento. Control de acceso a la representación interna de los objetos por medio de propiedades e indizadores.
6	23/09/2025	Teoría 6. Herencia. Especialización de clases. Invalidación de métodos. Modificadores de acceso: public, protected, private, e internal. Polimorfismo. Principio Open/Close
7	30/09/2025	Teoría 7. Interfaces. Implementación de múltiples interfaces. Implementación explícita de miembros de interfaces. Principio de inversión de dependencias (DIP) y Patrón de Inyección de Dependencias. Conceptos de Arquitectura Limpia.
8	07/10/2025	Teoría 8. Uso de las interfaces IComparable, IComparer, IEnumerable e IEnumerator de la plataforma .NET. Iteradores. Delegados. Pasaje de métodos como parámetro. Métodos anónimos. Expresiones lambda.
9	14/10/2025	Teoría 9. Métodos Genéricos. Tipos genéricos (Clases, interfaces y delegados).
10	21/10/2025	Teoría 10. Persistencia de datos. SQLite. LINQ. ORM Entity Framework Core
11	28/10/2025	Teoría 11. Contenedor de inyección de dependencias integrado en la plataforma .NET. Introducción al desarrollo de aplicaciones web con ASP.NET Core Blazor. Componente Razor.
12	04/11/2025	Teoría 12. Desarrollo de aplicaciones web de página única (SPA por Single Page App), integrando conceptos de Arquitectura Limpia, persistencia de datos con Entity Framework y desarrollo de la interfaz de usuario con ASP.NET Core Blazor.
13	11/11/2025	Teoría 13. Inicio a la Programación asincrónica. Patrón Asincrónico basado en Tareas (TAP). Métodos asincrónicos y tareas. Modificador async y operador await
		El resto de las clases se utiliza para consulta, corrección y devolución de trabajos y para la toma de coloquios.

Evaluaciones previstas	Fecha
Entrega final del trabajo integrador	20/11/2025
Coloquios	A partir del 25/11/2025 se asignarán turnos para rendir coloquio en los horarios de práctica y teoría

CONTACTO DE LA CÁTEDRA:



Mail: slpnet@lidi.info.unlp.edu.ar

Plataforma IDEAS (<https://ideas.info.unlp.edu.ar>) **Curso:** "Redictado Seminario de Lenguajes - Opción .NET Segundo Semestre 2025"

Firma del/los profesor/es

Leonardo Corbalan
Profesor Adjunto