Carrera: Ingeniería en Sistemas de Información

Asignatura: Desarrollo de Aplicaciones con Objetos

Planificación a partir del Ciclo Lectivo 2025

1. Datos administrativos de la asignatura

Nivel en la carrera	4	Duración	Cuatrimestral
Plan	2023		
Bloque curricular:	Asignatura Electiva		
Carga horaria presencial semanal (hs. cátedra):	4	Carga Horaria total (hs. reloj):	48
Carga horaria no presencial semanal (hs. reloj) (si correspondiese)		% horas no presenciales (hs. reloj) (si correspondiese)	

2. Presentación, Fundamentación

En la actualidad los lenguajes de programación ofrecen generalmente enfoques multiparadigma, permitiendo que una misma pieza de software pueda ser escrita aprovechando los paradigmas estructurado, orientado a objetos y funcional. Sin embargo el paradigma más ampliamente utilizado en la actualidad es el de la programación orientada a objetos. Diversas herramientas involucradas en las diferentes etapas del procesado de desarrollo están fuertemente apoyadas en dicho paradigma. Esta situación se presenta tanto en el desarrollo de software dirigido al usuario final como en la creación de frameworks o librerías ofrecidas a programadores. Por tales motivos es necesario que todo actor involucrado en tareas de programación conozca los fundamentos del enfoque y las técnicas asociadas con el mismo. Otras asignaturas del área incorporan en parte de su planificación las ideas iniciales y la aplicación de las mismas en la creación de programas. La asignatura el plantea el aprendizaje del paradigma de orientación a objetos con énfasis en un lenguaje de uso masivo, aplicándolo en el modelado de soluciones con patrones de diseño y en el desarrollo de sistemas informáticos de apoyo a sistemas de información.

Asimismo, y ante la necesidad de algunas asignaturas que requieren por parte de los estudiantes la creación de aplicaciones con interfaces gráficas de usuario, esta asignatura brinda conocimientos y herramientas necesarias para que puedan analizar un problema de programación, identificar alternativas de resolución y crear el software que resuelva el problema. También

presenta algunos de los patrones de diseño más frecuentemente utilizados, su ámbito de aplicación y sus alternativas de implementación.

- Relación de la asignatura con el perfil de egreso: Brinda habilidades técnicas necesarias en la elaboración de software.
- Relación de la asignatura con los alcances del título: Brinda conocimientos y herramientas básicas para aprender a modelar soluciones a problemas conocidos utilizando el paradigma más ampliamente aplicado y aprovechando las ideas de los patrones de diseño.

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.	No aporta
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

No corresponde

5. Objetivos establecidos en el DC

No corresponde

6. Resultados de aprendizaje

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

Identificador de RA	Redacción
RA1	Aplicar la programación orientada a objetos para desarrollar programas usando Python como lenguaje de programación.
RA2	Aplicar patrones de diseño en el modelado de clases para resolver problemas de ingeniería teniendo en cuenta la programación orientada a objetos.
RA3	Desarrollar programas en Python con alcances de ABM de un conjunto de datos persistentes en una base de datos relacional y con interfaz gráfica de usuario para aplicar la programación orientada a objetos y los patrones de diseño.

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	-	-	-	-	-	-	-	-	X	-
RA2	-	-	X	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	X	-
RA3	-	-	X	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	X	-

8. Asignaturas correlativas previas

Para cursar y rendir debe tener cursadas:

- Asignatura/s:
  Sintaxis y Semántica de los Lenguajes

Para cursar y rendir debe tener aprobada:

- Asignatura/s:
  Algoritmo y Estructura de Datos

9. Asignaturas correlativas posteriores

Indicar las asignaturas correlativas posteriores:

- Asignatura/s:
  No corresponde

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: Elementos del lenguaje Python

Contenidos: Instalación y configuración de entorno. Variables y control de flujo. Estructuras de datos provistas por el lenguaje. Lectura de archivos csv/json. Declaración y uso de funciones. Tipo de dato Función. Colecciones de funciones. Funciones de orden superior: map, filter. Funciones lambda.

Carga horaria por Unidad: 16 hs

Unidad Nº: 2

Título: Programación Orientada a Objetos

Contenidos: Clases. Objetos. Atributos. Métodos. Encapsulamiento. Composición: relaciones 1 a 1 / 1 a muchos. Herencia. Polimorfismo. Polimorfismo con colecciones. Excepciones.

Carga horaria por Unidad: 14 hs.

Unidad Nº: 3

Título: Interfaces gráficas de usuario

Contenidos: Ventanas y controles. Controles básicos de interacción (cuadros de texto, botones, casillas de selección múltiples y mutuamente excluyentes). Controles de listas. Controles de grillas.

Carga horaria por Unidad: 10 hs.

Unidad Nº: 4

Título: Patrones de diseño y técnicas de persistencia

Contenidos: Patrones GoF. Necesidad y uso. Implementación de patrones GoF: Singleton, Builder/Factory, Iterator, State/Strategy. Modelos de persistencia. Sistemas de altas, bajas y modificación y consultas (ABMC). Sanitización de entradas. Validaciones. Reglas de negocio. Reportes.

Carga horaria por Unidad: 24 hs

Carga horaria por tipo de formación práctica de toda la asignatura

Tipo de formación práctica	Horas reloj
Formación experimental	0
Análisis y resolución de problemas de ingeniería y estudios de casos	34
Formulación, análisis y desarrollo de proyectos.	0

Bibliografía Obligatoria:

- Apunte de la catedra (2025).
- The Python Software Foundation. (6 de noviembre de 2022). The Python Standard Library. https://docs.python.org/3/library/index.html

Bibliografía optativa y otros materiales a utilizar en la asignatura:

- Shvets, A. (2019). Dive Into Design Patterns. Refactoring.Guru.
- Gamma, E. (2003). Patrones de Diseño. Pearson Addison Wesley.

11. Metodología de enseñanza

- Los estudiantes disponen anticipadamente de los contenidos teóricos presentados en formato audiovisual (alojados en YouTube o similar) los cuales deben ser visualizados antes de cada clase presencial.
- Se fomenta que los estudiantes aprovechen las clases presenciales para interactuar con el cuerpo docente y sus compañeros, concentrando las actividades de ejercitación y de consultas. Asimismo, se espera que el estudio individual del temario, tanto desde el material bibliográfico como del audiovisual sea realizado fuera de las horas presenciales. En cada clase se entrega como tarea domiciliaria obligatoria la visualización de los videos de los temas correspondientes a la siguiente clase.
- Las clases inician con una introducción del tema a partir de los videos que los estudiantes deben haber visualizado anticipadamente, fomentando el intercambio entre los docentes y los estudiantes. A continuación, el docente plantea uno o más casos de aplicación del tema presentado, analizando estrategias de soluciones y finalmente muestra la implementación de la solución diseñada para los casos. Finalmente se plantea ejercitación para que los estudiantes desarrollen programas en forma individual o grupal. Cada clase cierra con un análisis de las soluciones obtenidas por los estudiantes.
- Se definirá una serie de cuestionarios con periodicidad quincenal y de corrección automática sobre los contenidos teóricos.
- Durante el cursado se lleva adelante el desarrollo de un trabajo práctico integrador grupal. El mismo tiene planificado su inicio durante las clases presenciales pero su envergadura y dificultad requiere que los estudiantes avancen con la solución fuera de hora. La consigna del mismo es presentada durante las primeras semanas del cursado y presenta una recomendación de desarrollo incremental con hitos de guía. Estos hitos se plantean a nivel de ofrecer un hilo conductor del desarrollo. Se exigirán entregas parciales no calificables, a los efectos de medir el avance de los grupos. Sin embargo, durante el cursado se reforzará la idea de que el cumplimiento de tales hitos contribuye doblemente a la ejercitación y al desarrollo del trabajo práctico.

12. Recomendaciones para el estudio
Se recomienda que previo a cada clase cada estudiante revise el material teórico correspondiente a la clase planificada, para esto el docente informa en cada clase el detalle de los temas y de los videos involucrados en la siguiente. Durante cada clase se espera que los estudiantes participen activamente de los espacios de exposición dialogada aportando consultas, opiniones y experiencias. Cuando el docente indique ejercitación se recomienda que cada estudiante realice en forma individual o grupal en la resolución de los ejercicios solicitados, registrando y consultando acerca de las dificultades o inconvenientes que surjan con motivo de la ejercitación. Luego de finalizada cada clase se recomienda que cada estudiante intente individualmente volver a resolver los ejercicios si en clase los desarrollo en forma grupal. Asimismo que resuelva otros ejercicios del tema visto en clase, obtenidos del material bibliográfico o recomendados por el docente.

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. En el inicio del cursado se llevará adelante una evaluación diagnóstica mediante un cuestionario de corrección automática con el objeto de determinar el nivel de conocimientos del alumnado en relación a las temáticas de las asignaturas correlativas y de otras asignaturas de los niveles segundo y tercero que aún sin ser correlativas pueden ser de interés o relacionadas con esta materia. La evaluación formativa consistirá en una serie de cuestionarios periódicos quincenales que medirán la evolución de los conceptos teóricos brindados durante el cursado y en el seguimiento del trabajo práctico integrador. Los resultados de los cuestionarios son promediados con promedio simple. En el caso de que existan cuestionarios no respondidos los mismos suman 0 para el cálculo del promedio. En las últimas semanas del cursado cada grupo deberá avanzar durante las clases presenciales con el desarrollo del trabajo práctico. La evaluación sumativa consistirá en:

- Parcial presencial individual. El parcial consta de dos partes. La primera consiste en el desarrollo de un programa que involucre las unidades 1 y 2. La segunda parte consiste en un cuestionario teórico práctico. El programa deberá ser resuelto utilizando las herramientas presentadas en la asignatura. La entrega del programa se realiza mediante una tarea del aula virtual en la que los estudiantes deben publicar el código fuente del programa construido. El desarrollo del programa será llevado adelante con uso libre de internet. El cuestionario teórico práctico contendrá preguntas de redacción libre en los que se abordarán temas teóricos no incluidos en la consigna de programación y preguntas prácticas relacionadas con el programa entregado. El cuestionario deberá ser respondido sin acceso a recursos de internet. Para aprobar el parcial debe estar aprobado al menos el 50% de ambas partes y un 60% del total. Los estudiantes que no alcancen una calificación igual o mayor a 6 acceden a una instancia de recuperación con la misma modalidad y temas del parcial. El desarrollo del parcial será llevado adelante con uso libre de internet.
- Recuperatorio del parcial. Con la misma modalidad y temas del parcial, durante la última semana del cursado.
- Trabajo práctico integrador. El trabajo práctico es grupal, en grupos de entre 3 y 5 integrantes y consiste en el desarrollo de un programa que aplicando los patrones aprendidos en la unidad 4 resuelva un caso provisto por la cátedra. La evaluación del trabajo analizará que el software cumpla con los requerimientos exigidos, que aplique los patrones más adecuados y que haga uso acorde de las herramientas obtenidas en las unidades 1, 2 y 3. La entrega del mismo se realiza mediante una tarea del aula virtual en la cual cada grupo debe adjuntar un archivo comprimido que contenga los archivos de código fuente. Con posterioridad a la entrega cada grupo deberá realizar un coloquio con preguntas relativas al software entregado en el que deben estar todos los integrantes del gurpo presentes. La inasistencia al coloquio es causal de desaprobación del TPI del integrante ausente. La calificación final del trabajo práctico se obtiene sumando los porcentajes de aprobación de cada parte (software y coloquio). Para aprobar el trabajo práctico integrador se requiere al menos de 50% de cada parte.
- Los grupos que no alcancen una calificación de aprobación del trabajo práctico integrador dispondrá de una instancia de reentrega en un nuevo plazo después de recibida la corrección. En la reentrega deberán reformular el software entregado para subsanar las observaciones recibidas en la corrección y alcanzar los objetivos solicitados originalmente.

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
RA 1	Lenguaje Python Programación orientada a objetos Interfaces gráficas de usuario	Estrategias: La estrategia principal es la de aula invertida, con material audiovisual disponible con antelación a cada clase. Actividades del estudiante: Los estudiantes deben visualizar los videos de contenido teórico / práctico de cada uno de los temas de las unidades paulatinamente antes de cada clase. Durante la clase los estudiantes en forma grupal o individual deberán aplicar los conceptos vistos para solucionar los problemas presentados por el docente creando programas en python en las computadoras de los laboratorios con apoyo del docente.	Instrumentos: Diagnóstico: mediante un cuestionario de corrección automática que deberá ser respondido por los estudiantes dentro de las primeras dos semanas se identificará el conocimiento previo de los temas de la asignatura. Cuestionarios: con periodicidad quincenal que los estudiantes deberán responder en forma obligatoria fuera del horario de clase. Los mismos servirán para evaluar los conocimientos y para realizar un seguimiento de la continuidad del cursado. Parcial presencial: consistirá en el desarrollo en forma individual de un programa para resolver una consigna provista por	Total de horas presenciales teórico/prácticas: 24 horas Total de horas de laboratorios: 9 horas Total de horas extra aulicas: 22 horas

			la cátedra en aulas de laboratorio. Criterios de evaluación: El estudiante crea un programa que resuelve un problema presentado. El programa funciona sin errores y cumple las consignas solicitadas. En el mismo se hace un uso adecuado de las herramientas del lenguaje. Identifica y usa las estructuras de datos más adecuadas. Diseña las clases necesarias para resolver un problema y las implementa sin errores de compilación ni ejecución. Identifica e implementa relaciones entre las clases. Diseña e implementa interfaces gráficas de usuario.	
RA 2	Patrones de diseño	Indique los contenidos. Estrategias:	Cuestionarios: con periodicidad quincenal que los estudiantes deberán responder en	Total de horas presenciales teórico/prácticas: 3 horas

		La estrategia principal es la de aula invertida, con material audiovisual disponible con antelación a cada clase. En cada clase el docente planteará uno o más casos de aplicación de los patrones analizados, y las ventajas, dificultades y alternativas de resolución. El docente planteará una o más estrategias de implementación de cada patrón y luego presentará un caso de estudio. Actividades del estudiante: Los estudiantes deben visualizar los videos de contenido teórico / práctico de cada uno de los temas de las unidades paulatinamente antes de cada clase. Durante la clase los estudiantes en forma grupal o individual deberán aplicar los conceptos vistos para solucionar los problemas presentados por el docente creando programas en python en las computadoras de	forma obligatoria fuera del horario de clase. Los mismos servirán para evaluar los conocimientos y para realizar un seguimiento de la continuidad del cursado. Los cuestionarios correspondientes al RA2 incluirán preguntas con presentación de casos de análisis en los que se requerirá por parte del alumno la selección adecuada de un patrón o de un conjunto de clases que lo implementen. Criterios de evaluación: El estudiante identifica los patrones aplicables para los casos presentados. El estudiante identifica las clases que implementan un patrón presentado. El estudiante diseña clases que implementen un patrón específico para los casos presentados.	Total de horas de laboratorios: 3 horas Total de horas extra aulicas: 6 horas

		los laboratorios con apoyo del docente.		
RA 3	Lenguaje Python Programación orientada a objetos Interfaces gráficas de usuario Patrones de diseño	Estrategias: La estrategia principal es la de aula invertida, con material audiovisual disponible con antelación a cada clase. El desarrollo del trabajo práctico aplica la estrategia de taller. Actividades del estudiante: Los estudiantes deben visualizar los videos de contenido teórico / práctico de cada uno de los temas de las unidades paulatinamente antes de cada clase. Durante la clase los estudiantes en forma grupal deberán avanzar con el desarrollo del trabajo práctico y presentar al docente el estado de dicho avance, evidenciando las dificultades o dudas que se les presenten.	Indique los contenidos. Instrumentos: La evaluación consistirá en un trabajo práctico integrador a realizar en forma grupal a lo largo de tres semanas con una consigna única que involucrará la programación de un software de ABM de un conjunto de datos en forma persistente. Criterios de evaluación: El software desarrollado aplica las herramientas más adecuadas del lenguaje de programación, funciona sin errores y cumple todos los alcances solicitados. En el diseño de las clases se identifican e implementan de los patrones más adecuados para el requerimiento solicitado.	Total de horas presenciales teórico/prácticas: 3 horas Total de horas de laboratorios: 6 horas Total de horas extra aulicas: 24 horas

			Las ventanas del software se comunican entre sí. El software graba y recupera datos de una base de datos relacional. El software valida los datos ingresados por el usuario. El software calcula y presenta los reportes solicitados.	

14. Condiciones de aprobación

Condiciones de regularización y aprobación directa:

La asignatura posee tres instancias de evaluaciones con calificación: un parcial, un trabajo práctico y un conjunto de cuestionarios.

Condiciones de regularización:

- El parcial (P) debe ser aprobado con nota igual o mayor a 6 en primera instancia o en el recuperatorio.
- El trabajo práctico (TP) debe ser aprobado con nota igual o mayor a 6 en la entrega original o en la reentrega.
- El promedio de cuestionarios (PC) deberá ser mayor o igual a 6.

Los estudiantes que alcancen la regularidad obtienen la aprobación directa. La nota final de aprobación directa se calcula como P * 0,4 + TP * 0,4 + PC * 0,2.

15. Modalidad de examen

- Todo estudiante que regularice obtiene la aprobación directa.

16. Recursos necesarios

Las clases se llevarán a cabo en laboratorio, donde se requiere:

- IDE: Visual Studio Code.
- Python 3.12.
- Acceso a Internet.
- Navegador Web.
- Proyector.