



GUÍA DOCENTE

PARADIGMAS DE PROGRAMACIÓN

GRADO EN INGENIERÍA DEL SOFTWARE

MODALIDAD: PRESENCIAL

CURSO ACADÉMICO: 2026-2027

Denominación de la asignatura:	Paradigmas de Programación
Titulación:	Grado en Ingeniería del Software
Facultad o Centro:	Centro Universitario de Tecnología y Arte Digital
Materia:	Programación
Curso:	4
Cuatrimestre:	1
Carácter:	OB
Créditos ECTS:	3
Modalidad/es de enseñanza:	Presencial
Idioma:	Castellano
Profesor/a - email	Gonzalo Nicolás Barreales / gonzalo.nicolas@u-tad.com Ana Isabel Sanchez Gonzalez / ana.sanchez2@u-tad.com
Página Web:	http://www.u-tad.com/

DESCRIPCIÓN DE LA ASIGNATURA

Descripción de la materia

Esta asignatura pertenece a la materia de programación. Esta materia se dedica al estudio de las técnicas y los lenguajes de programación en los que se fundamentarán los estudios del grado de ingeniería del software.

Descripción de la asignatura

Esta asignatura permite comprender cuales son los distintos paradigmas de programación que existen en la actualidad, mostrando las alternativas de las que se dispone a la hora de tomar decisiones para el desarrollo de software. A lo largo de la carrera se han visto los principales paradigmas de programación de las últimas décadas: Programación estructurada, programación procedimental, programación orientada a objetos, programación concurrente... En esta asignatura nos centraremos en dos nuevos enfoques: Programación funcional y Programación reactiva

El enfoque del curso será por tanto principalmente práctico, basado en el desarrollo exhaustivo de ejemplos sobre el entorno de desarrollo más habitual en estos momentos, que es Python y sus librerías especializadas

COMPETENCIAS Y RESULTADOS DE APRENDIZAJE DE LA MATERIA

Competencias (genéricas, específicas y transversales)

COMPETENCIAS BÁSICAS Y GENERALES:

CG1 - Capacidad para entender, planificar y resolver problemas a través del desarrollo de soluciones informáticas

CG3 - Conocimiento de los fundamentos científicos aplicables a la resolución de problemas informáticos

CG4 - Capacidad para simplificar y optimizar los sistemas informáticos atendiendo a la comprensión de su complejidad

CG9 - Capacidad para aprender, modificar y producir nuevas tecnologías informáticas

CG10 - Uso de técnicas creativas para la realización de proyectos informáticos

CB1 - Que los estudiantes hayan demostrado poseer y comprender conocimientos en un área de estudio que parte de la base de la educación secundaria general, y se suele encontrar a un nivel que, si bien se apoya en libros de texto avanzados, incluye también algunos aspectos que implican conocimientos procedentes de la vanguardia de su campo de estudio

CB2 - Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio

CB3 - Que los estudiantes tengan la capacidad de reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética

CB4 - Que los estudiantes puedan transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado

CB5 - Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores

con un alto grado de autonomía

COMPETENCIAS ESPECÍFICAS

CE1 - Conocimiento de la estructura de las computadoras, de los conceptos de codificación, manipulación, tratamiento de la información y uso de lenguajes de bajo nivel

CE7 - Conocimiento de los principales tipos de estructuras de datos y utilización de las librerías y de los técnicas algorítmicas asociadas a dichas estructuras junto con los órdenes de complejidad que caracterizan a dichas técnicas

CE8 - Conocimiento de los distintos paradigmas detrás de los lenguajes de programación

CE9 - Conocimiento de las estructuras de control, variables, sintaxis de programación y gestión del uso de la memoria de manera eficaz en el desarrollo de una aplicación informática

CE10 - Capacidad para manejar un gestor de versiones de código y generar la documentación de una aplicación de forma automática.

CE15 - Conocimiento de la tolerancia a los fallos, la adaptabilidad, el balance de carga y la predictividad del sistema para el desarrollo de aplicaciones distribuidas

CE17 - Conocimiento de las características de paralelización de tarjetas gráficas y de arquitecturas de altas prestaciones para el desarrollo de aplicaciones.

CE20 - Capacidad para testar el funcionamiento y funcionalidad de una aplicación informática, elaborando planes de pruebas y empleando técnicas de diseño y programación orientado a las pruebas

CE23 - Conocimiento de los principios de la inteligencia artificial y uso de algoritmos de búsqueda deterministas y máquinas de estado

Resultados de aprendizaje

Al acabar la titulación, el graduado o graduada será capaz de:

- Entender y manejar el concepto de memoria dinámica
- Identificar clases de objetos con los datos de un problema.
- Crear clases y objetos y manipularlos.
- Entender y utilizar los mecanismos de herencia, polimorfismo y sobrecarga de operadores.
- Identificar las relaciones entre clases en distintos casos de uso.
- Dominar un lenguaje de programación orientado a objetos.
- Dominar los patrones de programación
- Conocer las distintas formas de resolución de problemas desde el punto de vista de la algoritmia, como, por ejemplo, el esquema divide y vencerás,
- programación dinámica, backtracking o algoritmos genéticos.
- Estudiar la complejidad de un determinado algoritmo, interpretar dicha complejidad y analizar posibles optimizaciones.
- Codificar un programa que sea capaz de encontrar el camino óptimo que une dos nodos de un grafo siguiendo los distintos algoritmos de pathfinding.
- Crear y entrenar redes neuronales que solucionen problemas concretos.

CONTENIDO

Programación en diferentes paradigmas de la programación

TEMARIO

1. Introducción a los paradigmas de programación
 - 1.1. Introducción a Python
2. Programación Funcional
 - 2.1. Práctica 1
3. Programación Reactiva
 - 3.1. Práctica 2

ACTIVIDADES FORMATIVAS Y METODOLOGÍAS DOCENTES

Actividades formativas

Actividad Formativa	Horas totales	Horas presenciales
<i>Clases teóricas / Expositivas</i>	18	18
<i>Clases Prácticas</i>	9	9
<i>Tutorías</i>	2	1
<i>Estudio independiente y trabajo autónomo del alumno</i>	26	0
<i>Elaboración de trabajos (en grupo o individuales)</i>	17	0
<i>Actividades de Evaluación</i>	3	3

Metodologías docentes

Método expositivo o lección magistral

Aprendizaje de casos

Aprendizaje basado en la resolución de problemas

Aprendizaje cooperativo o colaborativo

Aprendizaje por indagación

Metodología Flipped classroom o aula invertida

Gamificación

Just in time Teaching (JITT) o aula a tiempo

Método expositivo o lección magistral

Método del caso

Aprendizaje basado en la resolución de problemas

Aprendizaje cooperativo o colaborativo

Aprendizaje por indagación

Metodología flipped classroom o aula invertida

Gamificación

DESARROLLO TEMPORAL

UNIDADES DIDÁCTICAS / TEMAS	PERÍODO TEMPORAL
1. Introducción a los paradigmas de programación	Semana 1 y 2
2. Programación Funcional	Semanas 3, 4, 5, y 6
3. Programación Reactiva	Semanas 6, 7, 8, 9, 10, 11, 12, 13, 14 y 15

SISTEMA DE EVALUACIÓN

ACTIVIDAD DE EVALUACIÓN	VALORACIÓN MÍNIMA RESPECTO A LA CALIFICACIÓN FINAL (%)	VALORACIÓN MÁXIMA RESPECTO A LA CALIFICACIÓN FINAL (%)
<i>Evaluación de la participación en clase, en prácticas o en proyectos de la asignatura</i>	0	30
<i>Evaluación de trabajos, proyectos, informes, memorias</i>	30	80
<i>Prueba Objetiva</i>	10	60

CRITERIOS ESPECÍFICOS DE EVALUACIÓN

ACTIVIDAD DE EVALUACIÓN	CONVOCATORIA ORDINARIA	CONVOCATORIA EXTRAORDINARIA
-------------------------	---------------------------	--------------------------------

<i>Evaluación de trabajos, proyectos, informes, memorias</i>	80	80
<i>Prueba Objetiva</i>	20	20

Consideraciones generales acerca de la evaluación

- The extraordinary assessment will be passed by taking an exam containing the parts failed in the ordinary exam:

BIBLIOGRAFÍA / WEBGRAFÍA

Paradigmas de Programación. (2013) Fernando Alonso Amo, Juan Alfonso Lara Torralbo, David Lizcano Casas y Loïc Martínez Normand

Programming Languages: Principles and Paradigms (2010) Maurizio Gabbrielli , Simone Martini

MATERIALES, SOFTWARE Y HERRAMIENTAS NECESARIAS

Tipología del aula

Aula teórica

Equipo de proyección y pizarra

Materiales:

Ordenador personal

Software:

-