UNIVERSIDAD DE MURCIA

1. Identificación

1.1. De la asignatura

Curso Académico	2024/2025
Titulación	GRADO EN INGENIERÍA INFORMÁTICA, PROGRAMA ACADÉMICO DE SIMULTANEIDAD DE DOBLE TITULACIÓN CON ITINERARIO ESPECIFICO DE GRADO EN MATEMÁTICAS Y GRADO EN INGENIERÍA INFORMÁTICA, PROGRAMA ACADÉMICO DE SIMULTANEIDAD DE DOBLE TITULACIÓN CON ITINERARIO ESPECÍFICO DE GRADO EN MATEMÁTICAS Y GRADO EN INGENIERÍA INFORMÁTICA
Nombre de la asignatura	TECNOLOGÍAS DE DESARROLLO DE SOFTWARE
Código	1905
Curso	TERCERO CUARTO CUARTO
Carácter	OBLIGATORIA
Número de grupos	5
Créditos ECTS	6.0
Estimación del volumen de trabajo	150.0 150.0 150.0
Organización temporal	1º Cuatrimestre 1º Cuatrimestre 1º Cuatrimestre
Idiomas en que se imparte	Español

1.2. Del profesorado: Equipo docente

GARCIA MOLINA, JESUS JOAQUIN

Docente: **PCEO MATE+INFOR_EXTINCION GRUPO 3,**

Coordinación de los grupos: **GRUPO 3**

Coordinador de la asignatura

Categoría
CATEDRATICOS DE UNIVERSIDAD

Área

LENGUAJES Y SISTEMAS INFORMÁTICOS

Departamento

INFORMÁTICA Y SISTEMAS

Correo electrónico / Página web / Tutoría electrónica

jmolina@um.es Tutoría electrónica: **Sí**

Teléfono, horario y lugar de atención al alumnado

Duración:	Día:	Horario:	Lugar:
C2	Miércoles	10:00-13:00	868884610, Facultad de Informática B1.2.009

Observaciones:
Podrán ser presenciales u online según desee el alumno. Los alumnos dispondrán del número de móvil del profesor. A. Presencial Despacho 2.06, Departamento de Informática y Sistemas, Facultad de Informática, Segunda Planta B. Online A través de Zoom o llamadas telefónicas

Duración:	Día:	Horario:	Lugar:
C1	Miércoles	10:00-13:00	868884610, Facultad de Informática B1.2.009

Observaciones:
Podrán ser presenciales u online según desee el alumno. Los alumnos dispondrán del número de móvil del profesor. A. Presencial Despacho 2.06, Departamento de Informática y Sistemas, Facultad de Informática, Segunda Planta B. Online A través de Zoom o llamadas telefónicas

BERMUDEZ RUIZ, FRANCISCO JAVIER

Docente: **GRUPO 2**

Coordinación de los grupos: **GRUPO 2**

Categoría

PROFESOR COLABORADOR (LICENCIADO)

Área

LENGUAJES Y SISTEMAS INFORMÁTICOS

Departamento

INFORMÁTICA Y SISTEMAS

Correo electrónico / Página web / Tutoría electrónica

fjavier@um.es Tutoría electrónica: **Sí**

Teléfono, horario y lugar de atención al alumnado

Duración:	Día:	Horario:	Lugar:
C1	Martes	12:00-13:30	868888524, Facultad de Informática B1.2.038

Observaciones:
PLAN A: Despacho 2.30 Fac.INFORMÁTICA PLAN B y C: Mediante herramientas online como Zoom y mensajes de AV

Duración:	Día:	Horario:	Lugar:
C1	Miércoles	12:00-13:30	868888524, Facultad de Informática B1.2.038

Observaciones:
PLAN A: Despacho 2.30 Fac.INFORMÁTICA PLAN B y C: Mediante herramientas online como Zoom y mensajes de AV

Duración:	Día:	Horario:	Lugar:
C2	Martes	16:00-18:00	868888524, Facultad de Informática B1.2.038

Observaciones:
PLAN A: Despacho 2.30 Fac.INFORMÁTICA PLAN B y C: Mediante herramientas online como Zoom y mensajes de AV

Duración:	Día:	Horario:	Lugar:
C2	Martes	10:00-11:00	868888524, Facultad de Informática B1.2.038 (DESPACHO BERMUDEZ RUIZ, FRANCISCO JAVIER 2.16)

Observaciones:
No consta

HOYOS BARCELO, JOSE RAMON

Docente: **GRUPO 1, GRUPO PCEO MATE+INFORM**

Coordinación de los grupos: **GRUPO 1, PCEO MATE+INFOR_EXTINCION GRUPO PCEO MATE+INFORM,**

Categoría
PROFESOR COLABORADOR (LICENCIADO)

Área
LENGUAJES Y SISTEMAS INFORMÁTICOS

Departamento
INFORMÁTICA Y SISTEMAS

Correo electrónico / Página web / Tutoría electrónica
jose.hoyos@um.es Tutoría electrónica: **Sí**

Teléfono, horario y lugar de atención al alumnado

Duración:	Día:	Horario:	Lugar:
C2	Martes	11:30-13:00	868884398, Facultad de Informática B1.2.024

Observaciones:
despacho 2.18

Duración:	Día:	Horario:	Lugar:
C2	Jueves	11:30-13:00	868884398, Facultad de Informática B1.2.024

Observaciones:
despacho 2.18

Duración:	Día:	Horario:	Lugar:
C1	Martes	12:00-13:30	868884398, Facultad de Informática B1.2.024

Observaciones:
despacho 2.18

Duración:	Día:	Horario:	Lugar:
C1	Miércoles	12:00-13:30	868884398, Facultad de Informática B1.2.024

Observaciones:
despacho 2.18

SANCHEZ CUADRADO, JESUS

Docente: **GRUPO 1, GRUPO 3, GRUPO PCEO MATE+INFORM**

Coordinación de los grupos:

Categoría
PROFESORES TITULARES DE UNIVERSIDAD

Área
LENGUAJES Y SISTEMAS INFORMÁTICOS

Departamento
INFORMÁTICA Y SISTEMAS

Correo electrónico / Página web / Tutoría electrónica
jesusc@um.es Tutoría electrónica: **Sí**

Teléfono, horario y lugar de atención al alumnado

Duración:	Día:	Horario:	Lugar:
A	Lunes	10:00-13:00	, Facultad de Informática B1.2.027 (JESÚS SÁNCHEZ CUADRADO)

Observaciones:
Despacho 2.21.

2. Presentación

Los objetivos principales son:

- Que los alumnos adquieran competencias básicas en la práctica del desarrollo de software destinadas a participar en la construcción de aplicaciones reales. Esta formación continuará en la asignatura "Procesos de Desarrollo de Software" del segundo cuatrimestre y de forma opcional en la Mención y Máster de "Ingeniería del Software.
- El objetivo anterior implica que los alumnos aprendan conceptos, técnicas y herramientas básicas en la construcción de software.
- Se utilizará el lenguaje Java para el desarrollo de un caso de estudio en el que se tendrán que utilizar todos los conocimientos aprendidos en las clases de teoría y prácticas: manejo de requisitos, modelado de software, programación funcional en Java, patrones GRASP, principios de diseño orientado a objetos y patrones de diseño, y arquitectura en capas, y persistencia de objetos.
- En el desarrollo de la aplicación del caso de estudio, los alumnos deberán abordar la creación de interfaces de usuario, utilizarán Git para el control de versiones y Maven como herramienta de construcción del proyecto.

3. Condiciones de acceso a la asignatura

3.1. Incompatibilidades

No constan

3.2. Requisitos

No constan

3.3. Recomendaciones

El alumno debería dominar los conceptos, técnicas y herramientas estudiados en la asignatura de "Programación Orientada a Objetos" impartida en el segundo curso, sería recomendable que tuviese esta asignatura aprobada.

4. Competencias

4.1. Competencias básicas

No constan

4.2. Competencias de la titulación

- CGII1: Capacidad de análisis y síntesis.
- CGII2: Capacidad de organización y planificación.
- CGII6: Capacidad de gestión de la información.
- CGII7: Resolución de problemas.
- CGII8: Toma de decisiones.
- CGII9: Trabajo en equipo.
- CGII10: Trabajo en un equipo de carácter interdisciplinar.
- CGII11: Trabajo en un contexto internacional.
- CGII14: Razonamiento crítico.
- CGII15: Compromiso ético.
- CGII16: Aprendizaje autónomo.
- CGII17: Adaptación a nuevas situaciones.
- CGII18: Creatividad
- CGII19: Liderazgo
- CGII20: Conocimiento de otras culturas y costumbres.
- CGII21: Iniciativa y espíritu emprendedor.
- CGII22: Motivación por la calidad.
- CGUM2: Comprender y expresarse en un idioma extranjero en su ámbito disciplinar, particularmente el inglés.
- CGUM6: Capacidad para trabajar en equipo y para relacionarse con otras personas del mismo o distinto ámbito profesional.
- CEII3: Capacidad para diseñar, desarrollar, evaluar y asegurar la accesibilidad, ergonomía, usabilidad y seguridad de los sistemas, servicios y aplicaciones informáticas, así como de la información que gestionan.
- CEII5: Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad.
- CEII8: Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.

- CEII9: Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero Técnico en Informática.
- CR6: Conocimiento y aplicación de los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos.
- CR7: Conocimiento, diseño y utilización de forma eficiente los tipos y estructuras de datos más adecuados a la resolución de un problema.
- CR8: Capacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.
- CR16: Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software.
- CR17: Capacidad para diseñar y evaluar interfaces persona computador que garanticen la accesibilidad y usabilidad a los sistemas, servicios y aplicaciones informáticas.

4.3. Competencias transversales y de materia

- El alumno aprenderá conceptos, técnicas y herramientas que le ayudarán a construir aplicaciones de un modo sistemático
- Aprender los patrones de diseño básicos para construcción de software OO: Creación, Estructurales y Comportamiento
- Aplicar patrones de diseño en el diseño e implementación de una aplicación
- Conocer un conjunto de criterios básicos para identificar defectos en el diseño del software
- Conocer los niveles de prueba del software a partir de la prueba unitaria y de integración: prueba de validación, de sistema y de aceptación
- Conocer los aspectos específicos de las pruebas de software OO
- Desarrollar componentes de interfaz grafica
- Valorar el papel de los sistemas de control de versiones de código fuente en el desarrollo de software

5. Contenidos

5.1. Teoría

Tema 1: Introducción al Desarrollo de Software

- Ingeniería del software: Una visión general
- Ciclo de vida
- Modelado del software: UML
- Diagramas de clases UML
- Diagramas de interacción UML
- Proceso de desarrollo software

Tema 2: Principios básicos en construcción de software orientado a objetos

- Patrones GRASP: Experto, Creador y Controlador
- Principio Separación Modelo-Vista
- Arquitectura en capas
- Principio abierto-cerrado y Ocultación de información
- Programación dirigida a la interfaz
- Favorecer la composición frente a la herencia
- Buenas prácticas básicas del Desarrollo dirigido por el dominio (DDD)

Tema 3: Conceptos avanzados de programación orientada a objetos

- Clases anidadas en lenguajes orientados a objetos. Caso Java: estáticas e internas
- Parametrizar métodos con funciones.Caso Java: Expresiones lambda e interfaces funcionales
- Patrones de diseño Estrategia y Command
- Iteradores en lenguajes de programación. Patrón Iterador
- Streams en Java
- Programación funcional en Java

Tema 4: Persistencia de Objetos

- Problema de la persistencia de objetos
- Mapping de objetos a datos almacenados en bases de datos
- Sistema de Persistencia de la asignatura
- Mapping de objetos Java a datos PersistenciaTDS
- Aplicaciones independientes del servicio de persistencia: Patrón DAO: Adaptador + Factoría Abstracta + Singleton

Tema 5: Patrones de Diseño

- Patrones de **creación:** Builder, Método Factoría
- Patrones **estructurales:** Composite, Fachada
- Patrones de **comportamiento:**Observer, Estado, Método Plantilla

5.2. Prácticas

Práctica 1: Interfaces de usuario. Estructura.

Objetivos:

El alumno aprenderá los componentes básicos que forman una ventana a través de la librería Swing de Java. También coonocerá como los builders de ventanas facilitan su creación y creará su primera ventana con WindowBuilder en Eclipse.

Práctica 2: Interfaces de usuario. Layouts.

Objetivos:

El alumno seguirá adquiriendo conocimientos y destrezas para **ser capaz de construir una interfaz gráfica de usuario** con un API Java como Swing. En la segunda sesión se familiarizará con algunos de los manejadores de layouts más extendidos como BorderLayout, BoxLayout y GridBagLayout. Creará su segunda ventana con Window Builder Pro.

-

Práctica 3: Interfaces de usuario. Creación de una ventana de aplicación

Objetivo: Se dedicará la clase a crear la ventana principal del caso de estudio de la asignatura.

Práctica 4: Taller de modelado de clases

Objetivo: Se realizarán dos ejercicios previamente entregados de modelado de las clases del dominio de una aplicación con diagramas de clase UML.

Relacionado con:

- Tema 1: Introducción al Desarrollo de Software

Práctica 5: Taller de modelado de interacciones

Objetivo: Se realizarán varios ejercicios de aplicación de patrones GRASP previamente entregados.

Relacionado con:

- Tema 1: Introducción al Desarrollo de Software

Práctica 6: Control de versiones con Git

Objetivos:

- Conocer los distintos conceptos que subyacen a las herramientas de control de versiones: revisión, copia local, tag, branch, commit, checkout, merge, etc, y cómo se manejan en Git
- Estudiar commands básicos de Git: clone, push, fetch, commit, push, merge, etc
- Instalar un repositorio Git en Github.

Práctica 7: Interfaces de Usuario. Manejadores de Eventos.

Objetivos: En la tercera sesión sobre construcción de interfaces de usuario, el alumno aprenderá los principales eventos asociados a botones, campos de entrada, listas, tablas, combobox, etc. y las técnicas para su manejo en Swing. Agregará manejadores a la ventana creada en la clase anterior.

Práctica 8: Construcción de proyectos con Maven

Objetivos:

- Aprender a **gestionar un proyecto con Maven,** la herramienta más utilizada para ese propósito.
- Comprender las distintas **etapas del ciclo de vida de un proyecto Maven**
- Aprender a manejar las **dependencias a librerías y software externo al proyecto.**

- ## Práctica 9: Manejo de Maven y Git en un proyecto

  Objetivo: Se trabajará sobre el proyecto del caso de estudio en Eclipse: creación del proyecto maven, se almacenará en el repositorio, se realizarán cambios, se instalarán las dependencias, se creará un jar entre otras actividfades.

- ## Práctica 10: Capas del Negocio y Almacenamiento

  Objetivos:

  - Aplicar la arquitectura en 3 capas a la aplicación del caso práctico de la asignatura: Controlador y Repositorios
  - Aplicar mapping de objetos del caso práctico al servicio de persistencia de la asignatura: Implementar algún DAO.
  - Analizar la aplicación ejemplo "Tienda"

  Relacionado con:
  - Tema 2: Principios básicos en construcción de software orientado a objetos
  - Tema 4: Persistencia de Objetos

- ## Práctica 11: Taller de ejercicios de programación funcional en Java

  Objetivo: Realizar ejercicios previamente entregados de programación basada en Streams de Java.

  Relacionado con:
  - Tema 3: Conceptos avanzados de programación orientada a objetos

- ## Práctica 12: Taller de ejercicios de patrones de diseño

  Objetivo: Realizar ejercicios previamente entregados de patrones de diseño en Java

  Relacionado con:
  - Tema 3: Conceptos avanzados de programación orientada a objetos
  - Tema 5: Patrones de Diseño

6. Actividades Formativas

Actividad Formativa	Metodología	Horas	Presencialidad
A1: Actividades con grupo grande de alumnos entre las que se encuentran la presentación en el aula de los conceptos propios de la materia mediante metodología expositiva con lecciones magistrales participativas y medios audiovisuales. También se contemplan en este grupo las actividades de evaluación teórico prácticas.		28.0	43.1
A2: Actividades con grupo mediano en el aula de resolución de problemas, seminarios, charlas, ejercicios basados en el aprendizaje orientado a proyectos, estudios de casos, exposición y discusión de trabajos relativas al seguimiento		2.0	12.5

individual y/o grupal de adquisición de las competencias.		
A3: Actividades con grupo pequeño en el laboratorio relacionadas con la componente práctica de las asignaturas, desarrollo de trabajos con equipo técnico especializado, desarrollo de programas, etc.	21.67	34.4
A5: Estudio y trabajo autónomo orientado a la asimilación de contenidos, realización de problemas, ejercicios o redacción de informes técnicos o memorias descriptivas, desarrollo de proyectos o prácticas individuales o en	98.33	0.0
Totales	150,00	

7. Horario de la asignatura

https://www.um.es/web/estudios/grados/informatica/2024-25#horarios

8. Sistemas de Evaluación

Identificador	Denominación del instrumento de evaluación	Criterios de Valoración	Ponderación
IE1	Examen teórico-práctico. En este instrumento incluimos desde el tradicional examen escrito o tipo test hasta los exámenes basados en resolución de problemas, pasando por los de tipo mixto que incluyen cuestiones cortas o de desarrollo teórico junto con pequeños problemas. También se incluye aquí la consideración de la participación activa del alumno en clase, la entrega de ejercicios o realización de pequeños trabajos escritos y presentaciones.	Examen de los contenidos del programa de teoría formado por cuestiones teórico-prácticas similares a las incluidas en los boletines de ejercicios que se entregan a los alumnos y se resuelven en clase. El examen consta de unas 8 cuestiones En el enunciado del examen se indica la puntuación de cada cuestión Se debe obtener un 5 para aprobar la parte de teoría La nota se mantiene para las siguientes convocatorias del curso en el caso de no aprobar la parte práctica (aplicable a convocatorias de Enero y Junio) Los alumnos podrán disponer durante el examen del material entregado en la asignatura y de los textos que consideren útiles. Podrán disponer de un portátil o iPad no conectado a Internet	50.0
IE2	Informe técnico. En este instrumento incluimos los resultados de actividades prácticas, o de laboratorio, junto con sus memorias descriptivas. Los resúmenes del estado del arte o memorias de investigación sobre temas concretos. Y la	Entrega del caso práctico propuesto a través de una especificación entregada en la segunda semana del curso. Este proyecto supondrá el desarrollo de una aplicación de tamaño pequeño (unas 2500 líneas de código) en la que los	50.0

	posibilidad de realizar entrevistas personales o presentaciones de los trabajos realizados también entran en esta categoría.	alumnos formarán grupos de dos y deberán mostrar la comprensión de los conocimientos y herramientas explicadas durante el curso Se realizará una entrega del modelo de clases antes de empezar con la implementación y entonces el profesorado publicará un modelo de clases de referencia Se mantendrá una entrevista de seguimiento sobre la primera semana de diciembre y una entrevista final tras la entrega de la práctica en período de exámenes Se creará una tarea en el aula virtual para la entrega final que incluirá el archivo zip obtenido al exportar el **proyecto Eclipse**y un **documento pdf** que deberá incluir lo siguiente: a) Diagrama UML de clases del dominio b) Diagrama de colaboración UML para una operación indicada c) Descripción breve de la arquitectura MVC de la práctica y otras cuestiones de diseño d) Descripción de la aplicación de los patrones de diseño utilizados e) Explicación breve del componente creado f) Código de al menos una clase de prueba Junit g Â (Opcional) Pequeño manual de usuario con las ventanas principales h) Otras cuestiones (tiempo dedicado, sugerencias, etc) No es necesaria una documentación del código usando javadoc, sino que deben incluirse comentarios en el código, como mí-nimo para explicar lo que hace cada método cuyo comportamiento no sea obvio y las partes más complejas del código En la **estructura del proyecto** deber-á existir una carpeta "lib" con las librerías usadas (configurar el build path con archivos locales en vez de externos) con el fin de facilitar su ejecución

La **valoración** de la práctica sería:

- Modelo de clases del dominio (primera entrega) (15%)

- Diseño de la interfaz (20%)

- Uso del componente (10%),

- Aplicación de patrones (15%),

- Código (separación en tres capas, legibilidad, uso de principios básicos de programación OO, implementación de patrones de diseño, uso de swing, comentarios) (30%),

- Uso de JUnit (5%),

- Documentación entregada (5%)

- La no utilización de Git supondrá una penalización en la nota final

Se debe obtener un 5 para aprobar la parte práctica

La nota se mantiene para las siguientes convocatorias del curso en el caso de no aprobar la parte práctica (aplicable a convocatorias de Enero y Junio)

9. Fechas de exámenes

https://www.um.es/web/estudios/grados/informatica/2024-25#examenes

10. Resultados del Aprendizaje

Objetivos Formativos

· Valorar el papel que juegan los patrones de diseño como forma de reutilización de la experiencia

· Aprender los patrones de diseño básicos para construcción de software orientado a objetos

· Adquirir destreza en la identificación de los patrones aplicables a un determinado problema

· Aplicar patrones de diseño en el diseño e implementación de una aplicación

· Comprender el concepto de refactorización como técnica de mejora de la calidad del software

· Conocer un conjunto de criterios básicos para identificar defectos en el diseño del software

· Aprender un catálogo de técnicas de refactorización para software orientado a objetos y aplicarlas en el desarrollo de un proyecto software

· Comprender la importancia de las pruebas en el proceso de refactorización

· Distinguir los conceptos de validación y verificación (vertical y horizontal)

· Conocer los niveles de prueba del software a partir de la prueba unitaria y de integración: prueba de validación, de sistema y de aceptación

· Ser capaz de desarrollar un plan de pruebas

· Ser capaz de revisar un segmento de código o diseño de tamaño medio mediante una inspección o walkthrough

· Conocer los aspectos específicos de las pruebas de software orientado a objetos

· Ser capaz de implementar pruebas unitarias y de integración en un proyecto de tamaño medio

· Ser capaz de aplicar un proceso básico de desarrollo de software dirigido por las pruebas

· Comprender el concepto de componente y su papel en el proceso de desarrollo de software

· Valorar el desarrollo basado en componentes como una técnica de reutilización de software

· Conocer los tipos de modelos de componentes y sus implementaciones tecnológicas

· Ser capaz de aplicar el desarrollo software basado en componentes para la construcción de interfaces gráficas de usuario

· Desarrollar componentes de interfaz gráfica

· Valorar el papel de los sistemas de control de versiones de código fuente en el desarrollo de software

· Conocer los modelos de gestión de código fuente y especialmente el modelo centralizado de mezcla de versiones

· Organizar un repositorio de código fuente y utilizarlo en un proyecto de desarrollo de software en equipo

· Conocer y utilizar herramientas de automatización de la construcción de software

· Utilizar un sistema de gestión de incidencias en un proceso de desarrollo

11. Bibliografía

Bibliografía básica

- (adicional) Eric T. Freeman et al. "Head First Design Patterns", O'Reilly, 2004.
- Desarrollo de software basado en pruebas. Apuntes de la asignatura. 2011
- Horstmann, C., Cornell G.; Core Java (Vol. 1). Fundamentos. Pearson. 2006.
- Mark Grand, Patterns in Java, vol. 1, John Wiley, Segunda Edición, 2002.
- Martin Fowler et al., "Refactoring. Improving the design of existing code", Addison-Wesley, 2001.
- Richard Warburton, "Java 8 Lambdas,", O'Reilly, 2014.
- -Craig Larman, UML y Patrones, 2ª edición, Prentice-Hall, 2002
- -Erich Gamma et al., Patrones de Diseño, Addison-Wesley, 2002.

- MAVEN: (Quick Reference Card)
- Tutorial Git

Bibliografía complementaria

- Documentación Git
- Tutorial Java Swing

12. Observaciones

Los criterios para establecer la nota que aparecerá en el acta en cada convocatoria son los siguientes:

- Si un alumno no se presenta al examen de teoría ni realiza ninguna entrega de prácticas, su calificación será "No Presentado"

- Si un alumno aprueba una parte y no se presenta a la otra, su calificación será "No presentado"

- Si un alumno suspende una parte y no se presenta o supera la otra, su calificación será "Suspenso" con la nota de la parte suspensa

- Cuando un alumno aprueba o suspende las dos partes, su calificación será resultado de aplicar los pesos establecidos para cada parte

Sí un alumno sólo supera una parte después de las 3 convocatorias del curso, la nota no se mantendrá para el próximo curso

Esta asignatura no se encuentra vinculada de forma directa con los Objetivos de Desarrollo Sostenible

NECESIDADES EDUCATIVAS ESPECIALES

Aquellos estudiantes con discapacidad o necesidades educativas especiales podrán dirigirse al Servicio de Atención a la Diversidad y Voluntariado (ADYV - https://www.um.es/adyv) para recibir orientación sobre un mejor aprovechamiento de su proceso formativo y, en su caso, la adopción de medidas de equiparación y de mejora para la inclusión, en virtud de la Resolución Rectoral R-358/2016. El tratamiento de la información sobre este alumnado, en cumplimiento con la LOPD, es de estricta confidencialidad.

REGLAMENTO DE EVALUACIÓN DE ESTUDIANTES

El artículo 8.6 del Reglamento de Evaluación de Estudiantes (REVA) prevé que "salvo en el caso de actividades definidas como obligatorias en la guía docente, si el o la estudiante no puede seguir el proceso de evaluación continua por circunstancias sobrevenidas debidamente justificadas, tendrá derecho a realizar una prueba global".

Se recuerda asimismo que el artículo 22.1 del Reglamento de Evaluación de Estudiantes (REVA) estipula que "el o la estudiante que se valga de conductas fraudulentas, incluida la indebida atribución de identidad o autoría, o esté en posesión de medios o instrumentos que faciliten dichas conductas, obtendrá la calificación de cero en el procedimiento de evaluación y, en su caso, podrá ser objeto de sanción, previa apertura de expediente disciplinario".