UNIVERSIDAD DE MURCIA

1. Identificación

1.1. De la asignatura

Curso Académico	2024/2025
Titulación	MÁSTER UNIVERSITARIO EN INGENIERÍA DEL SOFTWARE / MASTER IN SOFTWARE ENGINEERING
Nombre de la asignatura	CONTROL DE LA CALIDAD Y PRUEBAS DEL SOFTWARE
Código	7804
Curso	PRIMERO
Carácter	OBLIGATORIA
Número de grupos	1
Créditos ECTS	6.0
Estimación del volumen de trabajo	150.0
Organización temporal	1º Cuatrimestre
Idiomas en que se imparte	Español

1.2. Del profesorado: Equipo docente

FERNANDEZ ALEMAN, JOSE LUIS

Docente: **GRUPO 1**

Coordinación de los grupos: **GRUPO 1**

Coordinador de la asignatura

Categoría
CATEDRATICOS DE UNIVERSIDAD

Área
LENGUAJES Y SISTEMAS INFORMÁTICOS

Departamento
INFORMÁTICA Y SISTEMAS

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

Teléfono, horario y lugar de atención al alumnado

Duración:	Día:	Horario:	Lugar:
A	Miércoles	17:30-20:30	868884621, Facultad de Informática B1.2.014 (DESPACHO FERNANDEZ ALEMAN, JOSÉ LUIS 2.11)

Observaciones:
Despacho 2.11. Contactar previamente con el profesor. Las tutorías podrán atenderse por correo electrónico y videoconferencia

Duración:	Día:	Horario:	Lugar:
A	Jueves	09:30-12:30	868884621, Facultad de Informática B1.2.014

Observaciones:
Despacho 2.11. Contactar previamente con el profesor. Las tutorías podrán atenderse por correo electrónico y videoconferencia

MOROS VALLE, BEGOÑA

Docente: **GRUPO 1**

Coordinación de los grupos:

Categoría

PROFESORES TITULARES DE UNIVERSIDAD

Área

LENGUAJES Y SISTEMAS INFORMÁTICOS

Departamento

INFORMÁTICA Y SISTEMAS

Correo electrónico / Página web / Tutoría electrónica

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

Teléfono, horario y lugar de atención al alumnado

Duración:	Día:	Horario:	Lugar:
C1	Martes	10:30-12:00	868884608, Facultad de Informática B1.2.047

Observaciones:
Las tutorías se atenderán por videollamada o de manera presencial (despacho 2.37) bajo demanda.

Duración:	Día:	Horario:	Lugar:
C1	Lunes	12:00-13:30	868884608, Facultad de Informática B1.2.047

Observaciones:
Las tutorías se atenderán por videollamada o de manera presencial (despacho 2.37) bajo demanda.

Duración:	Día:	Horario:	Lugar:
C2	Martes	10:00-13:00	868884608, Facultad de Informática B1.2.047

Observaciones:
Las tutorías se atenderán por videollamada o de manera presencial (despacho 2.37) bajo demanda.

2. Presentación

La calidad del software es un aspecto fundamental en el desarrollo de sistemas y aplicaciones, ya que afecta directamente la satisfacción del usuario final y el éxito del producto Este curso proporciona a los estudiantes las habilidades y conocimientos necesarios para gestionar la calidad en el ámbito del desarrollo de software web y aplicar diversas técnicas y actividades para asegurar los diferentes atributos de calidad

La introducción al curso aborda conceptos clave relacionados con la gestión de la calidad del software Se explorará qué es la calidad del software y cómo se puede gestionar de manera efectiva Se estudiarán temas como la planificación, el control, el aseguramiento y la medición de la calidad del software, proporcionando a los estudiantes las bases necesarias para comprender los procesos involucrados en el aseguramiento de la calidad

Una parte fundamental del control de calidad es la realización de pruebas El curso abarcará una amplia gama de tipos de pruebas según la pirámide de Mike Cohen que sugiere que las pruebas deben estar equilibradas en diferentes niveles, con una base sólida de pruebas unitarias en la parte inferior, seguidas de pruebas de integración en el back-end, pruebas de interfaz de usuario en el front-end y, en la cúspide, pruebas manuales exploratorias Los discentes aprenderán cómo aplicar esta pirámide en el contexto de la programación web utilizando marcos de pruebas como Jest y Cypress Además, se abordarán las pruebas de regresión, que se enfocan en minimizar, seleccionar y priorizar los casos de prueba para garantizar que los cambios en el software no introduzcan errores en funcionalidades previamente probadas Además, se presentarán las pruebas de sistema, que incluyen pruebas de usabilidad, interfaz de usuario, rendimiento y seguridad Estas pruebas avanzadas son cruciales para garantizar que el software cumpla con los estándares de calidad y funcionalidad esperados

El curso también se adentrará en las pruebas metamórficas y las pruebas de mutación, que son técnicas más avanzadas utilizadas para mejorar la calidad del software Las pruebas metamórficas se centran en verificar el comportamiento del software frente a cambios predefinidos en los datos de entrada, mientras que las pruebas de mutación implican introducir cambios o errores controlados en el código fuente para evaluar la capacidad del conjunto de pruebas para detectar esos cambios

Además de los temas mencionados anteriormente, el curso también introducirá los conceptos clave del Desarrollo Dirigido por Comportamiento (BDD), incluyendo la creación de historias de usuario, el uso de lenguajes de especificación como Gherkin y la implementación de pruebas automatizadas basadas en comportamiento utilizando herramientas populares como Cucumber Los participantes aprenderán cómo utilizar estas técnicas para definir y verificar el comportamiento esperado del software, lo que permite una mayor claridad en los requisitos y una mejor comunicación entre los diferentes roles involucrados en el proyecto

Durante el curso, los estudiantes tendrán la oportunidad de aplicar los conocimientos adquiridos a través de ejercicios prácticos y un proyecto común a todas las asignaturas obligatorias del máster Esto les permitirá desarrollar habilidades prácticas en el diseño y ejecución de pruebas, así como en la interpretación de los resultados obtenidos

3. Condiciones de acceso a la asignatura

3.1. Incompatibilidades

No constan

3.2. Requisitos

No constan

3.3. Recomendaciones

No existen recomendaciones para esta asignatura.

4. Competencias

4.1. Competencias básicas

- CB6: Poseer y comprender conocimientos que aporten una base u oportunidad de ser originales en el desarrollo y/o aplicación de ideas, a menudo en un contexto de investigación
- CB7: Que los estudiantes sepan aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios (o multidisciplinares) relacionados con su área de estudio
- CB10: Que los estudiantes posean las habilidades de aprendizaje que les permitan continuar estudiando de un modo que habrá de ser en gran medida autodirigido o autónomo.

4.2. Competencias de la titulación

- CG1: Capacidad para integrarse en equipos de desarrollo de software que aplican conceptos, métodos, técnicas y tecnologías modernas que actualmente se usan en las principales empresas del mundo.
- CG2: Capacidad de producir software de calidad a través de la aplicación de principios, métodos y técnicas ampliamente aceptadas y usando herramientas extendidas.
- CG4: Capacidad para valorar soluciones software existentes y proponer cambios para su modernización.
- CG5: Habilidades para trabajo en equipo y gestión de equipos en proyectos software.
- CE10: Capacidad para diseñar y ejecutar cada tipo de prueba que debe ser aplicado a un producto software.
- CE11: Capacidad para usar herramientas de prueba y su integración con otras herramientas de automatización en el desarrollo y las operaciones.
- CE12: Capacidad de diseñar un plan de pruebas y validación para un proyecto de desarrollo de una aplicación.
- CE16: Capacidad de desarrollar una aplicación de calidad, abordando desde los requisitos hasta el despliegue en la nube, a través de prácticas ágiles y continuas, y aplicando técnicas que permitan conseguir usabilidad y la mayor productividad posible.

4.3. Competencias transversales y de materia

No constan

5. Contenidos

5.1. Teoría

Bloque 0: Aseguramiento de la calidad del software

Tema 1: Introducción a la calidad del software
Qué es la calidad del software

Gestión de la calidad del software: Planificación, control, aseguramiento y medición de la calidad del software

Fundamentos de las pruebas

Definición y validación del Plan de Pruebas

Introducción a fullstack testing

Bloque 1: Pruebas en programación Web

Tema 1: Técnicas estáticas
Técnicas dinámicas vs. Técnicas estáticas

Coste de las revisiones de software

Tipos de revisiones: informales, walktroughs, revisiones técnicas, inspecciones

Análisis estático de código

Tema 2: Pruebas unitarias
Introducción a las pruebas automáticas de código

La pirámide de la prueba en programación web: pruebas unitarias, pruebas de integración, pruebas end-to-end

Pruebas unitarias

Diseño de casos de prueba. Técnicas de caja blanca

Cobertura de código

Tema 3: Pruebas de caja negra
Introducción

Enfoques de pruebas de caja negra

Particiones de equivalencia

Análisis de valores límite

Tema 4: Mocks
Introducción

¿Cuándo son necesarios los mocks?

Dobles de prueba: stub, mock y spy

Ejemplo

Mocking Frameworks

Conclusiones

Tema 5: Pruebas del front-end
Pruebas de interfaz vs. Pruebas end-to-end

Pruebas de interfaz de usuario

Entorno de pruebas para JavaScript: JSDOM y React Testing Library

Tema 6: Pruebas de integración
Pruebas de back-end

Pruebas de integración

Pruebas de contratos

Pruebas de servicios: pruebas de endpoints y pruebas de datos.

Tema 7: Pruebas end-to-end
Introducción

Pruebas end-to-end

Pruebas end-to-end vs. Pruebas de aceptación

Cuándo escribir pruebas end-to-end

Herramientas para pruebas end-to-end: Selenium vs. Cypress

Tema 8: Pruebas de sistema: experiencia de usuario
Tipos de pruebas de sistema:

Pruebas de experiencia de usuario

Pruebas de usabilidad

Pruebas de accesibilidad

Tema 9: Prueba de sistema: rendimiento y seguridad
Pruebas de rendimiento

Pruebas de estrés

Pruebas de seguridad

Tema 10: Pruebas de desarrollo dirigido por comportamiento

Bloque 2: Pruebas avanzadas

Tema 1: Pruebas de mutación
Hipótesis de la prueba de mutación

Conceptos básicos: operador de mutación, puntuación de mutación, mutante equivalente, mutantes vivos/eliminados

Ventajas e inconvenientes de la prueba mutación

El proceso de la prueba de mutación

Reducción del tiempo de ejecución: weak mutation, strong mutation, firm mutation

Técnicas para reducir el coste de la prueba de mutación: reducción del número de mutantes y del tiempo de compilación y ejecución

Técnicas para reducir el número de mutantes equivalentes: higher order mutation, criterios para eliminar mutantes, técnicas heurísticas

Tema 2: Pruebas metamórficas
Introducción a las pruebas metamórficas

Ventajas e inconvenientes de las pruebas metamórficas

El proceso de las pruebas metamórficas

Casos de éxito con pruebas metamórficas

Tema 3: Pruebas de regresión
Introducción a las pruebas de regresión

5.2. Prácticas

- ### Práctica 1: Definición de un plan de pruebas

  Se definirá y validará un plan de pruebas para asegurarnos que es completo, efectivo y está alineado con los objetivos del proyecto.

  Relacionado con:
  - Bloque 0: Aseguramiento de la calidad del software
  - Tema 1: Introducción a la calidad del software

- ### Práctica 2: Análisis estático de código

  Instalación y prueba de la funcionalidad de ESLint y Prettier para control estático del código en VS Code.

  Relacionado con:
  - Tema 1: Técnicas estáticas

- ### Práctica 3: Pruebas de unidad con Jest

  Instalación del paquete Jest

  Definición de casos de prueba siguiendo las técnicas de caja blanca

  Agrupación de pruebas con Jest (test suites)

  Generación de informes de cobertura de código

  Relacionado con:
  - Bloque 1: Pruebas en programación Web
  - Tema 2: Pruebas unitarias

- ### Práctica 4: SonarQube

  Descarga e instalación de SonarQube Community Edition

  Proyectos y métricas en SonarQube

  Tipos de problemas detectados: errores, hedores, vulnerabilidades, puntos conflictivos de seguridad

  Consulta, modificación y creación de reglas SonarQube

  Consulta y definición de perfiles de calidad en SonarQube

  Relacionado con:
  - Bloque 1: Pruebas en programación Web
  - Tema 1: Técnicas estáticas
  - Tema 2: Pruebas unitarias

Práctica 5: Definición de mocks con Jest

Introducción

Mockear una función: jest.fn()

Mockear un módulo: jest.mock()

Resetear los mocks

Relacionado con:

- Tema 4: Mocks

Práctica 6: Pruebas front-end con JSDOM

Introducción a JSDOM

Definición de asertos sobre el DOM

Búsqueda de nodos del DOM con fom-testing-library

Mejora la legibilidad de los asertos con jest-dom

Gestión de eventos en las pruebas

Relacionado con:

- Tema 5: Pruebas del front-end

Práctica 7: Pruebas en el front-end: React Testing Library

Introducción

Descripción de los componentes de la aplicación

Primer test en React

Renderizado de componentes y localización de elementos

Interacción con componentes

Esperando a que termine la ejecución de un evento

Relacionado con:

- Tema 5: Pruebas del front-end

Práctica 8: Pruebas back-end: Insomnia y Rest Client

Pruebas manuales de endpoints: Insomnia y Rest Client

Swagger para documentación de endpoints

Relacionado con:

- Bloque 1: Pruebas en programación Web
- Tema 6: Pruebas de integración

Práctica 9: Pruebas back-end: Jest y supertest

Pruebas automáticas de endpoints

Simular el comportamiento de endpoints con mocks

Relacionado con:

- Tema 6: Pruebas de integración

Práctica 10: Pruebas de backend: bases de datos

Creación de un cluster de base de datos en MongoDB Atlas

Intalación del driver MongoDB para Node.js (Mongoose)

Definición del esquema de base de datos

Conectando Mongoose con el backend de la aplicación

Configuración de variables de entornos para establecer la base de datos de pruebas

Pruebas automáticas con la base de datos de pruebas

Relacionado con:

- Bloque 1: Pruebas en programación Web
- Tema 6: Pruebas de integración

Práctica 11: Pruebas end-to-end con Cypress

Instalación y ejecución de Cypress

Primer test en Cypress

Nuevo endpoint de la API para testing

Definición de asertos

Definición de comandos

Buenas prácticas con Cypress

Ejecución de Cypress en modo headless

Relacionado con:

- Bloque 1: Pruebas en programación Web
- Tema 7: Pruebas end-to-end

Práctica 12: Pruebas de mutación con Stryker

Herramientas de mutación

Instalación y uso de la herramienta Stryker mutator en un proyecto JavaScript simple proporcionado como recurso, empleando el test runner de Jest

Uso de la herramienta Striker mutator en el proyecto de programación web en desarrollo del máster

Relacionado con:

- Bloque 2: Pruebas avanzadas
- Tema 1: Pruebas de mutación

Práctica 13: Pruebas metamórficas

Desarrollo de un ejemplo de pruebas metamórficas (pruebas end-to-end)

Automatización de pruebas metamórficas con una herramienta

Diseño de pruebas metamórficas y uso de una herramienta automática

Relacionado con:

- Bloque 2: Pruebas avanzadas
- Tema 2: Pruebas metamórficas

Práctica 14: Pruebas de regresión

Desarrollo de un ejemplo de pruebas de regresión

Pruebas de regresión funcional, visual, de rendimiento y de seguridad

Diseño de pruebas regresión en el proyecto en desarrollo del máster

Relacionado con:

- Bloque 2: Pruebas avanzadas
- Tema 1: Pruebas de mutación

Práctica 15: Pruebas con usabilidad y accesibilidad

Herramientas de revisión y auditoría de usabilidad y accesibilidad: Accessibility Insights for Web, Axe, pa11y, AChecker, WAVE y Google Lighthouse

Relacionado con:

- Bloque 1: Pruebas en programación Web
- Tema 8: Pruebas de sistema: experiencia de usuario

Práctica 16: Pruebas de carga, rendimiento y seguridad

Prueba de carga con JMeter

Prueba de rendimiento y seguridad con Google Lighthouse

Relacionado con:

- Bloque 1: Pruebas en programación Web
- Tema 9: Prueba de sistema: rendimiento y seguridad

Práctica 17: Pruebas en BDD

Descripción de la sintaxis del lenguaje Gherkin

Definición de las palabras clave: Feature, Scenario, Given, When, Then, And, But

Escribiendo escenarios BDD utilizando Gherkin

Preparación en Gherkin de un ejemplo con historias de usuario, features y ejecución de pruebas

Introducción a las herramientas de automatización de pruebas BDD: Cucumber, Behave, SpecFlow, JBehave

Relacionado con:

- Bloque 1: Pruebas en programación Web
- Tema 10: Pruebas de desarrollo dirigido por comportamiento

6. Actividades Formativas

Actividad Formativa	Metodología	Horas	Presencialidad
AF1: Sesiones virtuales y síncronas.	Actividades de clase expositiva: exposición teórica, clase magistral, proyección, dirigida al gran grupo, con independencia de que su contenido sea teórico o práctico. La modalidad de impartición estará adaptada a las posibilidades que ofrece la docencia virtual. Junto a la exposición de conocimientos, en las clases se plantean cuestiones, en ocasiones utilizando herramientas de respuesta de audiencia, se aclaran dudas, se realizan ejemplificaciones, se establecen relaciones con las diferentes actividades prácticas que se realizan y se orienta la búsqueda de información.	20.0	0.0
AF2: Seminarios especializados virtuales y síncronos.	Seminarios: trabajo de los alumnos de profundización en una temática concreta, que puede integrar contenidos teóricos y prácticos, realizado en grupos reducidos y supervisado por el profesor, concluyendo con la elaboración y presentación escrita de un informe que, en algunos casos, puede hacerse público mediante exposición oral por parte de los alumnos y debate.	4.0	0.0
AF3: Prácticas de laboratorio virtuales y síncronas.	Actividades de clase práctica de aula: actividades prácticas de ejercicios y resolución de problemas, estudio de casos, actividades presenciales programadas en una docencia de aula invertida, aprendizaje basado en roles, contrato didáctico o de aprendizaje, exposición y análisis de trabajos, debates, simulaciones, etc. Suponen la realización de tareas por parte de los alumnos, dirigidas y supervisadas por el profesor, con independencia de que en el aula se realicen individualmente o en grupos reducidos. Aprendizaje orientado a proyectos. Los estudiantes, organizados en grupos, llevan a cabo de forma colaborativa la realización de un proyecto en un tiempo determinado abordando una tarea mediante la planificación, diseño y realización de una serie de actividades, en las cuales se puede utilizar también un aprendizaje basado en roles.	16.0	0.0
AF4: Exposición y discusión	Seminarios: trabajo de los alumnos de profundización en una temática concreta, que puede integrar contenidos teóricos y prácticos, realizado en grupos	4.0	0.0

virtual de trabajos.	reducidos y supervisado por el profesor, concluyendo con la elaboración y presentación escrita de un informe que, en algunos casos, puede hacerse público mediante exposición oral por parte de los alumnos y debate.		
AF5: Trabajo autónomo del alumno.	Trabajo autónomo del estudiante para repasar y estudiar los conceptos vistos en clase, realizar las tareas asignadas y preparación de las sesiones de la asignatura.	102.0	0.0
AF6: Tutorías formativas virtuales, síncronas y asíncronas, individualizadas y en grupo.	Tutorías en grupo: sesiones programadas de orientación, revisión o apoyo a los alumnos por parte del profesor, realizadas en pequeños grupos, con independencia de que los contenidos sean teóricos o prácticos, pudiendo ser presenciales o virtuales. Tutorías individualizadas: sesiones de intercambio individual con el estudiante, previstas en el desarrollo de la materia, pudiendo ser presenciales o virtuales.	4.0	0.0
	Totales	150,00	

7. Horario de la asignatura

https://www.um.es/web/master-software/2024-25#horarios

8. Sistemas de Evaluación

Identificador	Denominación del instrumento de evaluación	Criterios de Valoración	Ponderación
SE1	Entrevistas virtuales de seguimiento de prácticas: reuniones virtuales con los alumnos, de forma individual o en grupo, para que obtengan retroalimentación sobre sus trabajos o para comprobar sus habilidades en la ejecución de tareas, mostrando los conocimientos teóricos y prácticos adquiridos con la realización de las prácticas.	Este instrumento está relacionado con las sesiones prácticas de la asignatura. Se valorará: -Respuestas de los estudiantes a las preguntas que haga el profesor durante las entrevistas para controlar el proceso de aprendizaje - Asistencia y participación - Grado de avance en la tarea asignada sometida a seguimiento No se establece una nota mínima para este instrumento de evaluación	20.0

SE2	Evaluación de informes escritos, trabajos y proyectos: trabajos escritos realizados durante la ejecución de las prácticas y portafolios, con independencia de que se realicen individual o grupalmente.	Este instrumento está relacionado con las prácticas entregables y puntuables. Se valorará: - La presentación correcta del trabajo, según las pautas marcadas por el profesorado - Justificación de las decisiones tomadas - Corrección en respuestas - Grado de integración de los conocimientos teóricos y prácticos de la asignatura, desde un conocimiento reflexivo y analítico - Entrega puntual y en formato adecuado No se establece una nota mínima para este instrumento de evaluación	60.0
SE3	Evaluación de la presentación pública de trabajos: exposición de los resultados obtenidos y procedimientos necesarios para la realización de un trabajo, así como respuestas razonadas a las posibles cuestiones que se plantee sobre el mismo.		20.0

9. Fechas de exámenes

https://www.um.es/web/master-software/2024-25#examenes

10. Resultados del Aprendizaje

R1 Conocer y aplicar las diferentes actividades de prueba y aseguramiento de la calidad aplicables en los proyectos de ingeniería del software

R2 Capacidad de diseñar y aplicar un plan de pruebas y aseguramiento de la calidad

R3 Conocer el funcionamiento de las herramientas de prueba para automatizar actividades de prueba y aseguramiento de la calidad en diferentes tipologías de proyectos y plataformas

R4 Capacidad de retroalimentación continua sobre la calidad de la aplicación en desarrollo, usando las herramientas apropiadas

R5 Capacidad de gestionar la mejora continua de la calidad en las organizaciones software, a través de la implantación de actividades y técnicas para abordar el aseguramiento de los diferentes atributos de calidad

11. Bibliografía

Grupo: GRUPO 1

Bibliografía básica

- G. Mohan, "Full Stack Testing: A Practical Guide for Delivering High Quality Software," O'Reilly, 2022.
- Lucas Da Costa, "Testing JavaScript Applications," Manning Publications, 2021.
- P. Ammann and J. Offutt, "Introduction to Software Testing," Cambridge University Press, 2016.
- Jonathan Ramusson. The Way of the Web Tester. A Beginner´s Guide to Automating Tests. Pragmatic Bookshelf. 2016

Bibliografía complementaria

- "Software Testing: An ISTQB-BCS Certified Tester Foundation guide," BCS, The Chartered Institute for IT, 4th ed., 2019.
- C.M. Barnum, "Usability Testing Essentials: Ready, Set--Test!," Morgan Kaufmann, 2011.
- IEEE Std 829-2008, "IEEE Standard for Software and System Test Documentation," IEEE Computer Society, 2008.
- N. de Voil, "User Experience Foundations," BCS, The Chartered Institute for IT, 2020.
- R. Bierig, S. Brown, E. Galván, and J. Timoney, "Essentials of Software Testing," Cambridge University Press, 2022.
- R. Pressman and B. Maxim, "Software Engineering: A Practitioner's Approach," 9th ed., McGraw-Hill Education, 2020.
- S. Segura, G. Fraser, A. B. Sanchez and A. Ruiz-Cortés, "A Survey on Metamorphic Testing," in IEEE Transactions on Software Engineering, vol. 42, no. 9, pp. 805-824, 1 Sept. 2016, doi: 10.1109/TSE.2016.2532875.
- W. Mwaura, "End-to-End Web Testing with Cypress: Explore Techniques for Automated Front End Web Testing with Cypress and JavaScript," Packt Publishing, 2021.
- Y. Jia and M. Harman, "An Analysis and Survey of the Development of Mutation Testing," in IEEE Transactions on Software Engineering, vol. 37, no. 5, pp. 649-678, Sept.-Oct. 2011, doi: 10.1109/TSE.2010.62.

12. Observaciones

El Máster de Ingeniería del Software sigue una metodología de "aprendizaje basado en proyecto", de modo que el trabajo práctico de la asignatura estará relacionado con el caso de estudio del proyecto definido para el conjunto del Máster

RELACIÓN CON OBJETIVOS DE DESARROLLO SOSTENIBLE

Esta asignatura se encuentra vinculada de forma directa con los Objetivos de Desarrollo Sostenible 4 "Educacíón de calidad", 9 "Industria, innovación e infraestructura" y 12 "Producción y Consumo responsable"

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".