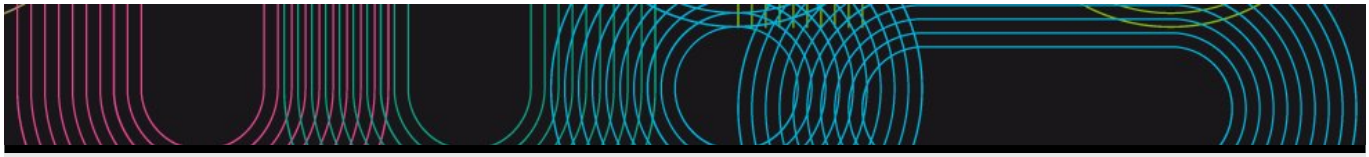

DATOS IDENTIFICATIVOS				
Programación II				
Asignatura	Programación II			
Código	V05G300V01302			
Titulacion	Grado en Ingeniería de Tecnologías de Telecomunicación			
Descriptores	Creditos ECTS	Seleccione	Curso	Cuatrimestre
	6	OB	2	1c
Lengua Impartición	Castellano			
Departamento	Ingeniería telemática			
Coordinador/a	Blanco Fernández, Yolanda			
Profesorado	Blanco Fernández, Yolanda Fernández Masaguer, Francisco			
Correo-e	yolanda@det.uvigo.es			
Web	http://www.faitic.es			
Descripción general	El objetivo general de la asignatura es proporcionar al alumno los fundamentos teóricos y las competencias prácticas que le permitan analizar, diseñar, desarrollar y depurar aplicaciones informáticas siguiendo el paradigma orientado a objetos. Esta es una asignatura eminentemente práctica y en este sentido está orientada al trabajo de los alumnos en la realización de uno o varios proyectos. Para facilitar el desarrollo de los proyectos, en la asignatura se realizará primeramente una muy breve introducción a la disciplina de Ingeniería del Software, conectandola con el paradigma de la programacion orientada a objetos (POO) y limitandola solo a las etapas de análisis, diseño, implementación y depuración. A continuación se analizarán en detalle los elementos de la POO, utilizando elementos y diagramas UML que serán utilizados por los alumnos en sus desarrollos. Para alcanzar este objetivo general los contenidos que se verán en la asignatura se pueden resumir en los siguientes ítems: - Conceptos básicos de Ingeniería del Software. - Conceptos básicos de la orientación a objetos: clases y objetos. - Encapsulación. Principio de ocultación. Conceptos de desacoplamiento y cohesión - Herencia, abstracción, polimorfismo y reutilización - Relaciones entre clases: generalización, asociación y dependencia. - Comunicación entre objetos: métodos, eventos, mensajes. - Persistencia. Almacenamiento en ficheros y en bases de datos. - Generación, captura y procesamiento de excepciones. - Lenguaje de modelado UML.			

Competencias	
Código	
B6	CG6 Facilidad para el manejo de especificaciones, reglamentos y normas de obligado cumplimiento.
B14	CG14 Capacidad para utilizar herramientes informáticas de búsqueda de recursos bibliográficos o de información.
C50	(CE50/T18) Capacidad de desarrollar, interpretar y depurar programas utilizando los conceptos básicos. de la Programación Orientada a Objetos (POO): clases y objetos, encapsulación, relaciones entre clases y objetos, y herencia.

C51	(CE51/T19) Capacidad de aplicación básica de las fases de análisis, diseño, implementación y depuración de programas en la POO.
C52	(CE52/T20) Capacidad de manejo de herramientas CASE (editores, depuradores).
C53	(CE53/T21) Capacidad de desarrollo de programas atendiendo a los principios básicos de calidad de la ingeniería del software, teniendo en cuenta las principales fuentes existentes en normas, estándares y especificaciones.

Resultados de aprendizaje

Resultados previstos en la materia	Resultados de Formación y Aprendizaje	
Comprender los aspectos básicos de la Programación Orientada a Objetos (POO).	B14	C50
Conocer los principales diagramas UML para la documentación de las fases de análisis y diseño de programas de acuerdo a la POO.	B6 B14	C52 C53
Adquirir habilidades sobre el proceso de análisis, diseño, implementación y depuración de aplicaciones de acuerdo a la POO, teniendo en cuenta los estándares principales y normas de calidad.	B6 B14	C51 C53
Adquirir una madurez básica en técnicas de desarrollo y depuración de programas para permitir el aprendizaje autónomo de nuevas capacidades y lenguajes de programación.	B6	C51 C52 C53

Contenidos

Tema	
1. Introducción al paradigma orientado a objetos	a. Breve introducción a la asignatura y su organización b. Nacimiento del paradigma c. Bases: clases y objetos d. Conceptos de encapsulación, herencia (generalización), y polimorfismo e. Breve introducción a UML.
2. Encapsulación	a. Clases, interfaces y paquetes b. Métodos y variables miembro. Visibilidad. Resolución de ámbito. c. Método constructor d. Paso de parámetros: punteros y referencias e. Punteros a objetos
3. Herencia	a. Clases derivadas y tipos de herencia b. Clases abstractas c. Herencia múltiple d. Clase object
4. Diseño orientado a objetos	a. Fundamentos de diseño. b. Conceptos básicos de la Ingeniería del Software. c. Utilización de diagramas UML
5. Polimorfismo	a. Sobrecarga y sobreescritura b. Clases abstractas e interfaces c. Clases genéricas
6. Gestión de excepciones	a. Fundamentos de excepciones b. Manipulación de excepciones en Java

Planificación

	Horas en clase	Horas fuera de clase	Horas totales
Lección magistral	28	42	70
Resolución de problemas	5	8	13
Resolución de problemas de forma autónoma	6	17	23
Estudio de casos	3	9	12
Aprendizaje basado en proyectos	7	18	25
Estudio de casos	1	0	1
Resolución de problemas	3	0	3
Práctica de laboratorio	2	0	2
Proyecto	1	0	1

*Los datos que aparecen en la tabla de planificación son de carácter orientativo, considerando la heterogeneidad de alumnado

Metodologías

	Descripción
Lección magistral	Clases que combinarán la explicación de los conceptos de la POO y la resolución de ejercicios para su aplicación. Estos pueden ser solucionados por el profesor o por los estudiantes, individualmente y/o en grupos. El objetivo es fomentar el debate en clase y fortalecer la adquisición de las competencias. Esta metodología está orientada a la adquisición de las competencias CE50, CE51 and CE53.

Resolución de problemas	En el laboratorio el profesor planteará pequeños programas que serán resueltos para que se puedan entender mejor los principales conceptos de la Programación Orientada a Objetos (POO). Esta metodología está orientada a la adquisición de las competencias CE50, CE51 and CE53.
Resolución de problemas de forma autónoma	Los alumnos resolverán de forma autónoma las prácticas que el profesor les plantee en el laboratorio. Las soluciones y las dudas que surjan al abordar dichos problemas serán discutidas para identificar los errores más comunes cometidos por los alumnos. Esta metodología está orientada a la adquisición por el alumno de las competencias CE50, CE51, CE53, CG6 y CG14.
Estudio de casos	El profesor supervisará y guiará a los alumnos durante el diseño de los diagramas UML, con la intención de identificar los errores cometidos en esta fase del proyecto. Esta metodología está orientada a la adquisición de las competencias CE51 y CE52.
Aprendizaje basado en proyectos	Los alumnos implementarán el sistema software planteado por el profesor. Dispondrán para ello de la segunda parte del curso, combinando trabajo presencial en el laboratorio supervisado por el profesor con trabajo no presencial. Esta metodología está orientada a la adquisición por el alumno de las competencias CE50, CE53, CG6 y CG14.

Atención personalizada

Metodologías	Descripción
Resolución de problemas	Revisión y comentarios de ejercicios resueltos en clase. Glosario de errores frecuentes y recomendaciones de estilo y organización de código. Consejos sobre buenas prácticas de programación.
Aprendizaje basado en proyectos	El profesor supervisará el nivel de entendimiento de los alumnos, asistiéndoles en dudas particulares, posibles errores de diseño y mejoras a nivel de código Java.
Resolución de problemas de forma autónoma	Revisión y comentarios a lo largo del desarrollo del proyecto, ayudando en labores de compilación y ejecución, amén de detectar y corregir errores conceptuales.
Estudio de casos	Análisis, detección de errores y discusión de posibles mejoras en los diseños UML presentados por los alumnos.

Evaluación

	Descripción	Calificación	Resultados de Formación y Aprendizaje	
Aprendizaje basado en proyectos	El proyecto consiste en el diseño final (diagramas UML), el código Java y la documentación correspondiente. El código necesariamente tiene que compilar y poder ser ejecutado en los ordenadores del laboratorio. El proyecto puede ser llevado a cabo individualmente o en grupos de 2 personas, según el mecanismo de evaluación elegido por cada estudiante. El profesor entrevistará a los alumnos para asegurar la autoría del código entregado y para ejecutar diferente pruebas de funcionalidad. En caso de grupos, ambos miembros tienen que asistir a la entrevista. Las cuestiones planteadas por el profesor deberán ser contestadas individualmente por cada estudiante para corroborar así su nivel de entendimiento e implicación durante el desarrollo del proyecto. Cada estudiante tiene que identificar la parte del proyecto de software que ha implementado. Los estudiantes que, a juicio del profesor, no consigan demostrar la autoría del código suspenderán la asignatura en la primera oportunidad. En otro caso, la nota de cada alumno dependerá de (i) sus respuestas durante la entrevista, (ii) la cantidad de pruebas de funcionalidad superadas, y (iii) la calidad del código Java en lo tocante a la adopción de las técnicas pilares de la POO.	35	B6 B14	C50 C53
Estudio de casos	Los estudiantes diseñarán el proyecto software planteado por el profesor mediante el lenguaje UML, incluyendo los diagramas solicitados y la documentación necesaria para entender las decisiones de diseño tomadas. El diseño UML puede ser desarrollado individualmente o en grupos de 2 personas, según el mecanismo de evaluación elegido. En caso de grupos de 2 alumnos, la nota individual de cada estudiante dependerá de la calidad de los diagramas UML presentados.	5		C51 C52
Resolución de problemas	Cada estudiante realizará --individualmente y sin material de apoyo-- un examen en la fecha oficial aprobada por la Junta de Escuela. Este examen combinará problemas, cuestiones de respuesta corta, elección múltiple y pruebas de verdadero/falso, orientadas a evaluar el nivel de entendimiento de los estudiantes en relación a los conceptos teóricos expuestos en las sesiones teóricas de la asignatura.	50		C50 C51 C53

Práctica de laboratorio	Esta prueba consiste en un conjunto de prácticas de iniciación Java que ayudarán a los alumnos a familiarizarse con un lenguaje de programación orientado a objetos. Estas prácticas serán entregadas únicamente por los alumnos que sigan la evaluación continua. Serán desarrolladas en grupos de 2 alumnos y su nota individual dependerá de: (i) las respuestas de cada alumnos a las cuestiones planteadas por el profesor en una entrevista, y (ii) la calidad y correcto funcionamiento del código Java entregado.	10	C50 C51 C52 C53

Otros comentarios sobre la Evaluación

Existen dos mecanismos de evaluación: evaluación continua (EC) y evaluación única (EU), que deberán ser elegidos por los alumnos considerando las siguientes condiciones:

- La EC incluye las 4 pruebas descritas en la sección anterior de este documento (examen de teoría, prácticas inicialicion de Java, diseño UML e implementación Java del proyecto).
- Los alumnos que opten por EU deben implementar el proyecto software (cuyas especificaciones se publicarán en faiTIC) individualmente.
- Mediante la entrega del diseño UML del proyecto, los estudiantes se comprometen a ser evaluados mediante EC, renunciando así al mecanismos de EU. En virtud de dicho compromiso, estos alumnos no podrán figurar como "No presentados".
- Los estudiantes que no entreguen el diseño UML renuncian al mecanismo de EC, siendo necesariamente evaluados mediante EU. No será posible unirse a la EC en las siguientes pruebas.
- El calendario de todas las pruebas de evaluación será aprobado por la CAG y puesto a disposición de los alumnos al principio del cuatrimestre de impartición de la asignatura.
- Las pruebas de EC sólo se llevarán a cabo en las fechas estipuladas por los profesores. No podrán repetirse más tarde.
- Las notas de EC y de otros exámenes y proyectos prácticos no serán válidas más allá del año académico actual.
- En caso de plagio, el estudiante recibirá la nota "suspenso (0)" y este hecho se notificará a la dirección del Centro a los efectos oportunos.

Procedimiento de evaluación en primera oportunidad para alumnos que opten por EC:

Parte teórica: Examen (50%). Examen individual sin ningún tipo de material de apoyo. Es la tercera prueba descrita en la sección de evaluación. La nota de este examen sólo podrá ser guardada para la segunda oportunidad en caso de obtener 4.5 o más puntos (sobre 5).

Parte práctica. Se incluyen las siguientes pruebas:

- **Prácticas de iniciación Java** (10%). Serán desarrolladas en grupos de 2 alumnos. Es la cuarta prueba descrita en la sección de evaluación.
- **Proyecto** (40%). Será desarrollado en grupo de 2 alumnos. El proyecto consta de dos partes:
  - **Diseño UML** (5%). Es la segunda prueba descrita en la sección de evaluación.
  - **Implementación Java** (35%). Es la primera prueba descrita en la sección de evaluación, en la que se incluye el código Java, la documentación Javadoc correspondiente y la entrevista de autoría con el profesor.

Para superar la asignatura mediante el mecanismo de EC los alumnos deben reunir los siguientes requisitos:

- Conseguir al menos 1/3 de la nota máxima de la parte teórica.
- Conseguir al menos 1/3 de la nota máxima de la implementación Java del proyecto de la parte práctica.
- Conseguir una nota final (parte teórica + parte práctica) igual o superior a 5 puntos (sobre 10).
- Si la nota final es igual o mayor que 5 pero el alumno no alcanza las notas mínimas mencionadas anteriormente, su nota final será suspenso (4.5).

Procedimiento de evaluación en primera oportunidad para alumnos que opten por EU:

Parte teórica: **Examen** (50%). Examen individual sin ningún tipo de material de apoyo. Es la tercera prueba descrita en la

sección de evaluación. Los alumnos que hayan obtenido en primera oportunidad una nota en esta parte de 4.5 puntos o más (sobre 5), podrán rescatar su calificación sin necesidad de repetir el examen de teoría.

Parte práctica: **Proyecto** (50%). Debe ser desarrollado individualmente. Es la primera prueba descrita en el apartado de evaluación, en el que se incluye código Java, diseño UML, documentación Javadoc y entrevista de autoría con el profesor.

Para superar la asignatura mediante el mecanismo de EU los alumnos deben reunir los siguientes requisitos:

- Conseguir al menos 1/3 de la nota máxima de la parte teórica.
- Conseguir al menos 1/3 de la nota máxima de la parte práctica.
- Conseguir una nota final (parte teórica + parte práctica) igual o superior a 5 puntos (sobre 10).
- Si la nota final es igual o mayor que 5 pero el alumno no alcanza las notas mínimas mencionadas anteriormente, su nota final será suspenso (4.5).

Procedimiento de evaluación en la segunda oportunidad:

Parte teórica: **Examen** (50%). Examen individual sin ningún tipo de material de apoyo. Es la tercera prueba descrita en la sección de evaluación. La nota de este examen no se guardará nunca.

Parte práctica: Proyecto (50%). Es la primera prueba descrita en la sección de evaluación. En la evaluación de esta parte se contemplan tres posibles escenarios, en función del mecanismo de evaluación elegido por el alumno (EC, EU) y las notas obtenidas en la parte práctica durante la primera oportunidad. Con independencia del escenario concreto, aquellos alumnos que hayan optado por EU en la primera oportunidad deberán necesariamente realizar el proyecto individualmente en la segunda oportunidad.

[Scenario #1] La nota del proyecto puede ser guardada para la segunda oportunidad. Aunque los alumnos pueden rescatar las notas obtenidas en la primera oportunidad (en los supuestos descritos a continuación), siempre tienen la opción de mejorar su calificación previa entregando una versión del proyecto planteado en la segunda oportunidad (cuyas especificaciones se publicarán en faiTIC). En este caso, los estudiantes deberán proporcionar un documento en el que se describan los cambios que hayan hecho en el diseño de la primera versión del proyecto para poder acomodar las nuevas funcionalidades exigidas.

Este escenario es aplicable a estudiantes que optaron por EC en la primera oportunidad, siempre que la nota de su proyecto haya sido igual o mayor que 1.5 (sobre 4 puntos), y que su nota del diseño UML haya sido igual o mayor que 0.3 (sobre 0.5). En caso de que el alumno no rescate su nota de primera oportunidad y opte por entregar las funcionalidades exigidas en la segunda oportunidad, la implementación del proyecto se evaluará con hasta 3.5 puntos (sobre 10) y el diseño UML con hasta 0.5 puntos (sobre 10). La nota de las prácticas de iniciación Java (hasta 1 punto sobre 10) será la obtenida en la primera oportunidad.

Este escenario también es válido para los alumnos que hayan optado por EU, habiendo obtenido una nota en el proyecto igual o superior a 2.5 puntos (sobre 5). En caso de optar por una nueva entrega en segunda oportunidad, el proyecto será evaluado con hasta 5 puntos (sobre 10).

[Scenario #2] Se conservan las notas obtenidas en el diseño UML y las prácticas de iniciación Java, pero es preciso entregar un nuevo proyecto en la segunda oportunidad que será evaluado sobre 3.5 puntos.

Este escenario es válido para estudiantes que hayan optado por EC en la primera oportunidad obteniendo una nota en el proyecto igual o mayor que 1 punto pero menor que 1.5 puntos (sobre 4), y una nota en el diseño UML mayor que 0.3 puntos (sobre 0.5).

[Scenario #3] No es posible rescatar ninguna nota de la primera oportunidad. En este caso el alumno deberá entrega un nuevo proyecto que será evaluado con hasta 5 puntos (sobre 10).

Este escenario es aplicable a estudiantes que optaron por EC en la primera oportunidad cuya nota en el proyecto no alcanzó 1 punto (sobre 4) y cuya calificación en el diseño UML fue menor que 0.3 (sobre 0.5).

Este escenario también es válido para estudiantes que no entregaron el proyecto durante la primera oportunidad, y para alumnos que no consiguieron superar la entrevista de autoría realizada con el profesor.

Para superar la asignatura en la segunda oportunidad los alumnos deben reunir los siguientes requisitos:

- Conseguir al menos 1/3 de la nota máxima de la parte teórica.

- Conseguir al menos 1/3 de la nota máxima de la parte práctica (3.5 puntos en EC, y 5 puntos en EU).
- Conseguir una nota final (parte teórica + parte práctica) igual o superior a 5 puntos (sobre 10).
- Si la nota final es igual o mayor que 5 pero el alumno no alcanza las notas mínimas mencionadas anteriormente, su nota final será suspenso (4.5).

Procedimiento de evaluación en la convocatoria extraordinaria (fin de carrera):

Parte teórica: **Examen** (50%). Examen individual sin ningún tipo de material de apoyo. Es la tercera prueba descrita en la sección de evaluación. La nota de esta parte no podrá ser rescatada de un examen de teoría previo en ningún supuesto.

Parte práctica: **Proyecto** (50%). Deberá ser desarrollado individualmente. Es la primera prueba descrita en el apartado de evaluación, en la que se incluye código Java, diseño UML, documentación Javadoc y entrevista de autoría con el profesor.

Para superar la asignatura en la convocatoria extraordinaria (fin de carrera) los alumnos deben reunir los siguientes requisitos:

- Conseguir al menos 1/3 de la nota máxima de la parte teórica.
- Conseguir al menos 1/3 de la nota máxima de la parte práctica.
- Conseguir una nota final (parte teórica + parte práctica) igual o superior a 5 puntos (sobre 10).
- Si la nota final es igual o mayor que 5 pero el alumno no alcanza las notas mínimas mencionadas anteriormente, su nota final será suspenso (4.5).

Fuentes de información

Bibliografía Básica

W. Savitch, **Absolute Java**, 4ª edición, Pearson, 2010

Y. D. Liang, **Introduction to Java programming**, 8ª, Pearson, 2010

P. Deitel, H. Deitel, **Java: How to program**, 9ª, Pearson, 2011

Bibliografía Complementaria

B. Eckel, **Thinking in Java**, 4ª edición, Prentice-Hall, 2006

P. Niemeyer, D. Leuck, **Learning Java**, 4ª edición, O'Reilly., 2013

Oracle, **Java SE. Oracle**,

Oracle, **Java API Specifications**, 2016

G. Booch, J. Rumbaugh, I. Jacobson, **The Unified Modeling Language User Guide**, 2, Addison-Wesley., 2005

S. Zakhour, S. Hommel, J. Royal, I. Rabinovitch, T. Risser, M. Hoeber, **The Java Tutorial. A short course on the basics**, 4ª edición, Prentice-Hall, 2006

A. Eberhart, S. Fischer, **Java Tools**, Wiley, 2002

M. Page-Jones, **Fundamentals of object-oriented design in UML**, Addison-Wesley, 2002

M. Fowler, **UML Distilled: A Brief Guide to the Standard Object Modeling Language**, 3ª edicion, Addison-Wesley., 2003

Jean-Michel DOUDOUX, **Développons en Java 2.10**, 2016

Recomendaciones

Asignaturas que se recomienda haber cursado previamente

Programación I/V05G300V01205