MINISTERIO
DEDEFENSA

GUÍA DOCENTE

Fundamentos de Lenguaje de Programación Orientada a Objetos

CURSO PARA LA OBTENCIÓN DEL DIPLOMA DE INFORMÁTICA MILITAR

CÓDIGO DE SIPERDEF DEL CURSO: 59130 2023 001

ACADEMIA DE INGENIEROS

DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN Y CIBERDEFENSA

FECHA DE ELABORACIÓN: 2 de septiembre de 2024

ÍNDICE

1. FICHA DE LA ASIGNATURA, MATERIA, MÓDULO O CURSO.

Nombre de la asignatura: Fundamentos de Lenguaje de Programación Orientado a Objetos (POO)
Materia: Programación
Curso: Curso para la Obtención del Diploma de Informática Militar para Oficiales
Módulo: Desarrollo de Aplicaciones
Tipo de curso: Perfeccionamiento.
Código del curso: 59130
Centro: ACADEMIA DE INGENIEROS
Departamento: Sistemas de Información y Ciberdefensa (SIyCD)
Créditos ECTS: 7 **Horas/ETCS:** 25 **Carga total de trabajo**: 175 horas
Duración[1]: Cuatrimestral
Carácter[2]: Obligatoria
Profesorado: CTE. INF DIM D. Ismael Lanchas Díaz
Horario de Tutoría: L y J de 14:00 a 15:00
Idioma en el que se imparte: Español

[1] Anual / Semestral / Cuatrimestral
[2] Obligatoria / Optativa / Prácticas

2. PRESENTACIÓN.

Las asignaturas de programación del curso para la obtención del Diploma de Informática Militar son impartidas por el Departamento de Sistemas de Información y Ciberdefensa.

El papel de estas asignaturas en el plan de estudios es muy importante ya que proporcionan los conocimientos necesarios para poder implementar posteriormente las aplicaciones.

Junto con los conocimientos adquiridos en las asignaturas “Introducción a la programación” y “Algoritmos y estructuras de datos”, esta asignatura constituye la base fundamental de los conocimientos de programación, en general y de orientación a objetos en especial, adquiridos durante el curso. Se emplea el lenguaje de programación Java.

Fundamentos de Lenguaje de POO es la base fundamental para el resto de las asignaturas del Módulo de Desarrollo de Aplicaciones, aportando los conocimientos necesarios para implementar la parte lógica y de negocio.

El desarrollo de la materia es presencial, con 80 horas de clase presencial, salvo que la enseñanza híbrida recomiende lo contrario.

3. COMPETENCIAS.

- **Competencias generales.**

- Haber adquirido conocimientos avanzados y demostrado, en un contexto altamente especializado, una comprensión detallada y fundamentada de los aspectos teóricos y prácticos y de la metodología de trabajo en el campo de los sistemas de información. (C.G.1)
- Ser capaces de asumir la responsabilidad de su propio desarrollo profesional y de su especialización en uno o más campos de estudio. (C.G.2)

- **Competencias específicas.**

- Concebir, planificar, dirigir, implementar y mantener proyectos de desarrollo software. (C.E. 3)
- Asesorar al mando sobre la mejor solución y sus alternativas con criterios objetivos en función de los requisitos determinados. (C.E. 4)
- Asumir y perseguir el valor de soluciones altamente reutilizables, flexibles, abiertas, interoperables, escalables y las ventajas del trabajo formando parte de comunidades. (C.E. 5)

- **Unidades de Competencia.**

- Describir los conceptos de programación orientada a objetos y aplicarlos para

realizar proyectos de programación. (U.C. 3.12)
- Aplicar librerías básicas y APIs de acceso a datos. Aprovechar APIs para crear sistemas altamente flexibles utilizando distintas librerías. (U.C. 5.4)
- Capacidad de aprendizaje autónomo de nuevas herramientas, frameworks, APIs y lenguajes de programación. (U.C. 5.5)

Por tanto, se debe ser capaz de desarrollar código robusto y seguro, aplicando el paradigma de la programación orientada a objetos, para programas sencillos que, tras el modelado de problemas reales, resuelvan estos con soluciones flexibles, de fácil mantenimiento y extensión e interoperables mediante la observación de estándares (como la guía de estilo).

4. RESULTADOS DE APRENDIZAJE.

Los resultados de aprendizaje hacen referencia a los logros que el alumno debe alcanzar al finalizar la asignatura:

- Realizar programas sencillos que empleen diferentes tipos de datos y operadores
- Realizar programas sencillos que empleen diferentes sentencias de control de flujo
- Realizar programas sencillos que empleen métodos
- Realizar programas sencillos que empleen clases
- Realizar programas sencillos que gestionen los errores mediante excepciones
- Realizar programas sencillos que empleen herencia y polimorfismo
- Realizar programas sencillos que empleen interfaces, genéricos y colecciones
- ~~Realizar operaciones avanzadas con datos~~

5. CONTENIDOS.

La asignatura se articula en Unidades didácticas.

Unidad Didáctica 1: Fundamentos

UA1:	Hola Mundo
UA2:	Mostrando más mensajes
UA3:	Ejercicios en *"Think Java"* y resto de sitios
UA4:	Variables
UA5:	Operadores
UA6:	Métodos
UA7:	Condiciones
UA8:	Bucles
UA9:	Arrays

UA10:	Contenido básico extra

Unidad Didáctica 2: Pensando en Objetos

UA1:	Visión General
UA2:	Clasificación de Tipos
UA3:	Clases
UA4:	Objetos
UA5:	Herencia
UA6:	Polimorfismo
UA7:	Interfaces
UA8:	Colecciones
UA9:	Listas
UA10:	Comparable

Unidad Didáctica 3: Avanzando hacia la abstracción

UA1:	Comparable vs Comparator
UA2:	Integrando código externo
UA3:	Genéricos

~~**Unidad Didáctica 4: Entrada y Salida**~~

~~**UA1:**~~	~~Escribir y leer datos~~
~~**UA2:**~~	~~Serialización~~

6. METODOLOGÍAS DE ENSEÑANZA-APRENDIZAJE Y ACTIVIDADES FORMATIVAS.

La metodología enseñanza-aprendizaje a emplear será expositiva, indagatoria y participativa con las siguientes características:

- Clase prioritariamente presencial con posibilidad de reconvertirse a enseñanza online en función de las circunstancias formando parte de la enseñanza híbrida.
- Aprendizaje basado en prácticas.
- Promover una enseñanza basada en una actitud de mejora profesional y personal.
- Promover la participación de los alumnos compartiendo las prácticas planteadas en la asignatura.
- Tutoría
- Evaluación.

6.1. Distribución de carga de trabajo en horas.

UNIDAD DIDÁCTICA	HORAS TEÓRICAS	HORAS PRÁCTICAS	HORAS ESTUDIO/ TRABAJO INDIVIDUAL	HORAS EXAMENES	TOTAL
1	**10**	**8**	**28**	**1**	**46**
2	**15**	**11**	**40**	**1**	**71**
3	**10**	**10**	**40**	**1**	**58**
~~**4**~~	~~**2**~~	~~**5**~~	~~**5**~~	~~**0**~~	~~**12**~~
TOTAL	**35**	**29**	**108**	**3**	**175**

	ECTS / horas
Trabajo presencial	**2.6 / 67**
Trabajo no-presencial	**4.4 / 108**
Total ECTS/horas	**7 / 175**

6.2. Estrategias metodológicas y actividades.

La estrategia metodológica que se seguirá en la asignatura tendrá las siguientes pautas y actividades.

- Enseñar los conocimientos necesarios para alcanzar las competencias exigidas.
- Motivar al alumno a través de implementación de diferentes prácticas en el aula de Sistemas de Información y Ciberdefensa de la Academia de Ingenieros.
- Explicar los objetivos que se pretenden alcanzar a lo largo de las diferentes Unidades Didácticas, trasmitiendo la utilidad de la asignatura para las otras del módulo al que pertenece, así como para el curso en general.
- Solicitar la participación de los alumnos, compartiendo los problemas y soluciones que conlleva la programación orientada a objetos.
- Observar las buenas prácticas mientras se programa, incluyendo la guía de estilo marcada.
- Fomentar aprendizaje activo e interactivo mediante la resolución de ejercicios. Es fundamental el rol activo del alumno para que sea partícipe en la construcción de su propio conocimiento, incentivando la búsqueda proactiva de soluciones más eficientes. Esta actitud será clave en el caso de tener que migrar a una enseñanza online o mediante seminarios web (webminars).

7. ORIENTACIONES PARA EL ESTUDIO Y REALIZACIÓN DE ACTIVIDADES.

7.1. Orientaciones generales para abordar el estudio.

A continuación, se reflejan algunas orientaciones para abordar el estudio:

- Leer compresivamente la Guía Docente, tomando nota de la estructura de la asignatura y práctica/s a desarrollar durante el desarrollo de esta.
- Como todo lenguaje en el ámbito de los sistemas de información es aconsejable utilizar un aprendizaje por descubrimiento, evitando un aprendizaje repetitivo.
- Programar un calendario de trabajo: Se recomienda revisar los contenidos de cada sesión previamente a la misma con el objeto de identificar los contenidos más relevantes, así como posibles dudas que sean necesarias resolver en sesiones posteriores. Así mismo, es conveniente revisar el contenido de las unidades didácticas con anterioridad a la realización de los exámenes.
- Al ser una asignatura eminentemente práctica es aconsejable dedicarle tiempo diariamente para coger soltura en la implementación del lenguaje y mantener un ritmo a la altura de las sesiones siguientes.
- El tiempo dedicado es mejor programarlo a la misma hora todos los días y en caso de dificultades no excederlo, siendo mejor cambiar a otro ejercicio o tarea y preguntar al profesor.
- Evitar caer en el error de emplear un tiempo excesivo en un asunto y desmoralizarse cuando habitualmente el profesor puede solucionarlo en muy poco tiempo y el alumno aprende dónde está su problema particular igualmente, evitando por otra parte una solución que implique una mala práctica.
- En Internet es habitual encontrar malas prácticas en foros o videos creados por no profesionales. Es importante diferenciar sitios de referencia seguros y huir de sumideros de tráfico como los que se anuncian con frases llamativas del tipo “Aprende a programar en 1 día”. En la bibliografía se pueden encontrar enlaces a sitios seguros.
- El tiempo dedicado debe ser principalmente a implementar funcionalidades con código. Un estudio pasivo centrado en la lectura o la visualización de videos no es eficaz.
- Lo más eficaz es marcarse una meta clara, que esté fuera de la zona de confort y seguir trabajando hasta alcanzarla sin conformarse con una solución parcial intermedia.

7.2. Cronograma general de la asignatura.

La asignatura se desarrolla durante toda la subfase básica, siendo el examen ordinario en los últimos días de ella. El programa oficial se encuentra en https://web.institutomilitar.com/semanal.html.

La evaluación extraordinaria de recuperación se realizará, en su caso, al inicio de la subfase de aplicación.

Dicho cronograma será igualmente válido para el desarrollo de la enseñanza a distancia, modificándose en la parte práctica según las circunstancias. La evaluación será siempre presencial.

8. TUTORIZACIÓN Y SEGUIMIENTO.

Durante el desarrollo de la asignatura los alumnos podrán realizar las consultas que consideren oportunas a los profesores. Para las consultas presenciales solicitarán al profesor correspondiente una tutoría que será de 14:00h a 15:00h de lunes y jueves en el Dpto. Sistemas de la Información y Ciberdefensa.
Con objeto de la asignatura existe un canal en el chat de la plataforma del Dpto. SIyCD (#java), poniendo en común las dudas de interés para todo el curso y que los profesores del departamento aportarán su solución.

También se ofrecen tutorías online, en horario a convenir con el profesor, utilizando la plataforma oficial del curso.

Además, los alumnos disponen del correo oficial y teléfono de cada profesor titular para un contacto directo, según la siguiente relación:

Cte. Lanchas email: ilandia@et.mde.es telf: 819 5025

9. EVALUACIÓN.

9.1. Procedimiento general de evaluación.

Se llevará a cabo una evaluación final presencial, para demostrar las competencias y resultados de aprendizaje descritos anteriormente, mediante una prueba práctica acerca de la implementación con código fuente en Java sobre las partes que se soliciten para un supuesto de problema real.

La fecha de evaluación será conforme la cronología establecida anteriormente.

Además, habrá una evaluación continua y aleatoria que ajuste la calificación final a la participación del alumno durante las sesiones de clase.

9.2. Criterios de evaluación.

Los criterios de evaluación de la asignatura serán los siguientes:

- Se realizará una prueba objetiva presencial escrita de evaluación de conocimientos prácticos sobre la asignatura completa. Se exige alcanzar un 5 como nota para aprobar la asignatura en un baremo de 0 a 10.
- Sobre ese baremo se podrá aumentar o disminuir en un punto la participación en clase. Durante las sesiones de clase se realizarán preguntas sobre alumnos aleatoriamente y se registrará el éxito en la participación:

- Las respuestas correctas aumentarán la calificación.
- Las respuestas incorrectas o la falta de ellas la disminuirán.
- La suma de los registros positivos y negativos determinará la modificación de la calificación final según la fórmula del siguiente punto.

– Sobre el mismo baremo, se puede penalizar, hasta con dos puntos en la calificación total de la evaluación, la falta de respeto a las buenas prácticas en el código como:
  - No observar la guía de estilo marcada
  - No utilizar convenios propios del lenguaje
  - No tener bien organizado el código en los paquetes y archivos
  - No utilizar nombres descriptivos de tipos, variables o métodos
  - Utilizar múltiples retornos en un bloque
  - Encontrar código repetido evitable
  - Utilizar de forma ineficaz o ineficiente las distintas posibilidades que ofrece el lenguaje (por ejemplo, no respetar SOLID)
  - Código muerto (por ejemplo, sin acceso o comentado sin necesidad)
  - Salidas no necesarias o pedidas por consola
  - No funcionamiento en general del código implementado
  - Entregar archivos que deben evitarse (por ejemplo, incluir en un commit archivos binarios del tipo .class)

9.3. Calificación final.

La calificación de la asignatura se hará conforme a la siguiente fórmula.

Nasig = Ev + NParticip - Penal

Nasig = Nota asignatura
Ev = Calificación en prueba de evaluación
NParticip = Sumatorio normalizado a +1 / -1
Penal = Penalización por no respetar buenas prácticas

Para resultar apto **Nasig** y **Ev** deberá ser mayor o igual a cinco (5)

Los alumnos que hayan sido no aptos en la asignatura en la convocatoria ordinaria deberán realizar una nueva prueba de evaluación para la recuperación. En la recuperación no se tendrá en cuenta el valor NParticip. La calificación final de esta nueva prueba no podrá ser en ningún caso superior a 5, siendo necesario un 5 para resultar apto.

La no superación de la asignatura supondrá la no superación del módulo, y por extensión del curso conforme a lo especificado en el punto 11.1 del currículo.

9.4. Recuperación

Esta prueba de evaluación se realizará en similares condiciones y en fecha a determinar por el profesor para esta convocatoria extraordinaria salvo indicación contraria que lo recomiende.

Con carácter general, la recuperación se realizará sin perjuicio de las sesiones programadas inicialmente para el resto de los alumnos.

10. BIBLIOGRAFÍA

- **Bibliografía básica**
  - Apuntes para el alumno: https://hijosdelspectrum.blogspot.com/p/curso-java-de-zero-zen-jazz-descripcion.html
  - Libro de referencia (salvo genéricos) en inglés: https://greenteapress.com/wp/think-java/
  - Repositorio con ejercicios, autoevaluaciones y exámenes anteriores: https://github.com/Awes0meM4n/codigoHijosDelSpectrum/tree/master/Java/practicas#pr%C3%A1cticas
  - Exámenes de años anteriores en GitEIE: https://git.institutomilitar.com

- **Bibliografía complementaria**
  - Repositorio con el código hecho en clase: https://github.com/DptoSIC/CodigoClase
  - Sitio web con ejercicios variados: https://www.codewars.com/users/Awes0meM4n/authored_collections
  - Foro de referencia de preguntas y respuestas: https://stackoverflow.com/