SILABO DEL CURSO TÉCNICAS DE PROGRAMACIÓN ORIENTADA A OBJETOS

I. INFORMACIÓN GENERAL

Facultad	Ingeniería	**Carrera profesional**	Ingeniería de Sistemas Computacionales	**Ciclo**	5°	**Créditos**		5
Código de curso	SIST1202A	**Requisitos**	Fundamentos de Programación (*)	**Horas**	**HT**	**HP**	**HL**	**PC**
					2	0	4	2
Tipo de curso	Obligatorio	**Modalidad del curso**	Presencial	**Periodo lectivo**	2025-1			
El curso aporta a la(s) competencia(s) general(es):	• Resolución de problemas • Responsabilidad social y ciudadanía							
El curso aporta a la(s) competencia(s) específica(s):	• Análisis de Problemas • Diseño y Desarrollo de Soluciones							
El curso desarrolla el componente:	• Responsabilidad social y formación ciudadana							
ODS (número y nombre):	• No aplica							

II. SUMILLA

El curso es de naturaleza teórico-práctico y tiene como propósito brindar los conocimientos del Paradigma Orientado a Objetos aplicados a la programación, desarrollando en el estudiante la capacidad de implementar modelos de clases en un lenguaje de programación orientado a objetos. Los temas principales son: fundamentos de programación orientado a objetos, representación de clases y relaciones entre clases, programación visual y acceso a base de datos.

III. LOGRO DEL CURSO

Al finalizar el curso, el estudiante desarrolla un proyecto, utilizando el lenguaje Java y aplicando el paradigma de programación orientado a objetos, interfaces gráficas y acceso a datos para resolver problemas específicos; demostrando lógica, habilidad, legibilidad y buenas prácticas en la implementación

IV. METODOLOGÍA DE ENSEÑANZA APRENDIZAJE

Para alcanzar el logro de aprendizaje del curso y de las unidades, el docente integra métodos activos, estrategias y técnicas de manera reflexiva y crítica, buscando motivar, estimular y guiar el aprendizaje del estudiante.

Las estrategias y técnicas didácticas que se utilizan son: resolución de casos en forma de ejercicios propuestos de manera individual y en equipo, para los cuales los estudiantes analizan el contexto, abstraen requerimientos de software, analizan y generan diagramas de clases y generan programas utilizando lenguaje de programación orientado a objetos y documentos básicos de desarrollo de software. Además, se realizan exposiciones individuales y grupales, donde muestran progresivamente avances del proyecto de aplicación, orientado a brindar solución a un problema. Asimismo, se desarrollan actividades grupales de responsabilidad social y analizan el impacto de las mismas.

El docente soporta su práctica pedagógica en un sistema de multiplataformas y recursos multimedia que le permiten el desarrollo de actividades sincrónicas y asincrónicas, así como la gestión de contenidos, videoconferencias y el uso de diversas herramientas tecnológicas para generar experiencias formativas y brindar orientaciones que promuevan el aprendizaje y el desarrollo de competencias generales y específicas en los estudiantes.

HT=Horas de desarrollo teórico
HP= Horas de desarrollo práctico
HL= Horas de desarrollo práctico en laboratorio
PC= Horas de práctica de campo

V. ORGANIZACIÓN DE UNIDADES DE APRENDIZAJE

UN	NOMBRE / LOGRO DE UNIDAD	SEM	SABERES ESENCIALES
I	**Fundamentos de programación orientada a objetos** Al finalizar la unidad, el estudiante desarrolla casos aplicando correctamente los principios de programación orientada a objetos, demostrando buenas prácticas de programación y utilizando sistemas de control de versiones para gestionar eficazmente su código.	1	Presentación del sílabo. Paradigma Orientado a Objetos. Fundamentos de la programación Orientada a Objetos. Java como herramienta de programación orientada a objetos. Principales sentencias en Java. Introducción a la programación orientada a objetos. Clases y controladores.
		2	Estructura de programas: escáner y sentencias condicionales y de control. Clases de biblioteca (String, Random, Math) y métodos Java. Clases, objetos y métodos. Modificadores de acceso.
		3	Parámetros y métodos de sobrecarga. Modificador estático y clases anidadas. Variables, métodos, clases estáticos. Manejo de errores.
		4	Colecciones. Taller de análisis de problemas.
		5	Requerimientos de sistemas. Historias de usuario. Diagramas UML
		6	**Evaluación T1** Identificación de restricciones realistas para la solución de problemas y establecer alternativas de solución.
II	**Clases y relaciones entre clases** Al finalizar la unidad, el estudiante desarrolla casos, haciendo uso correcto de diagramas y buenas prácticas de programación orientado a objetos, demostrando creatividad y responsabilidad en su trabajo, utilizando sistemas de control de versiones para gestionar eficazmente su código.	7	Herencia. Diagrama de clases y desarrollo de casos. Polimorfismo.
		8	Clases y métodos abstractos. Modificador final. Interfaz. Herencia Múltiple. Diagrama de clases.
		9	Relaciones entre clases: Binaria. Reflexiva. Diagrama UML. Desarrollo de casos
		10	**Evaluación T2**
		11	Relaciones entre clases: Agregación. Composición Diagrama UML. Desarrollo de casos
III	**Programación visual y acceso a base de datos** Al finalizar la unidad, el estudiante será capaz de desarrollar aplicaciones de programación visual y con acceso a datos que, para almacenamiento de información, demostrando creatividad y buenas prácticas de programación utilizando sistemas de control de versiones para gestionar eficazmente su código.	12	**Swing** Programación de componentes gráficos.
		13	**Java FX**. Componentes del proyecto. Eventos. **Evaluación T3**
		14	Gráficos, audio. JDBC: Java DataBase Connectivity: JDBC. Base de Datos. Conectividad JDBC. Conectando con SQL.
		15	JDBC: Puente JDBC-ODBC y Lenguaje SQL. Consultas SQL. Manipulación de Base de Datos. CRUD a tablas.
		16	**Evaluación Final**
		(-)	**No Aplica evaluación sustitutoria**

HT=Horas de desarrollo teórico
HP= Horas de desarrollo práctico
HL= Horas de desarrollo práctico en laboratorio
PC= Horas de práctica de campo

VI. SISTEMA DE EVALUACIÓN

EVALUACIÓN	PESOS	SEM	DESCRIPCIÓN DE LA EVALUACIÓN (Acción + Producto de la evidencia que debe presentar el estudiante)
T1 (a)	10%	6	Desarrollo de casos
T2 (a)	20%	10	Desarrollo de casos
T3 (a)	30%	13	Evaluación de práctica de campo
Evaluación final (EF) (a)	40%	16	Presentación y sustentación de proyecto final

(a) Los calificativos deben ser publicados en el sistema de acuerdo con el Calendario Académico establecido para el presente Semestre.

VII. BIBLIOGRAFÍA BÁSICA

N°	AUTOR	TÍTULO	AÑO	ENLACE URL
1	Blasco, Francisco	Programación orientada a objetos en Java	2019	https://elibro.bibliotecaupn.elogim.com/es/lc/upnorte/titulos/127125

a) BIBLIOGRAFÌA COMPLEMENTARIA

N°	AUTOR	TÍTULO	AÑO	ENLACE URL
1	Jiménez de Parga, Carlos - Autor	UML: arquitectura de aplicaciones en Java, C++ y Python	2021	https://elibro.bibliotecaupn.elogim.com/es/lc/upnorte/titulos/222720
2	Blasco, Francisco	Programación en Java	2020	https://digitalia.bibliotecaupn.elogim.com/a/110219

VIII. INFORMACIÓN COMPLEMENTARIA

REFERENCIA	ENLACE URL
Biblioteca Virtual UPN	https://biblioteca.upn.edu.pe/
Agile Mindset	https://conectaempleo-formacion.fundaciontelefonica.com/web/pe-agile-mindset-ed-1
Presentaciones en Público y Digitales	https://conectaempleo-formacion.fundaciontelefonica.com/web/pe-presentaciones-en-publico-y-digitales-ft-ed-6
JF Java Fundamentals Alumno - español	https://myacademy.oracle.com/

ANEXO: Ficha de Horas de Práctica de campo (Técnicas de programación orientada a objetos)

SEMANA	NOMBRE DE LA UNIDAD	ACTIVIDADES PRÁCTICAS DE CAMPO	EVIDENCIA DE PRÁCTICA DE CAMPO (Acción + Producto)	ENTREGA DE EVIDENCIA
1	I Fundamentos de programación orientada a objetos	**Desarrollar guía de primeros pasos con Git.** Creación de repositorio local, clonar un repositorio de GitHub a la máquina local. Navegación básica por el repositorio. Primer commit. Realizar cambios en el código, Utilizar comandos git add y git commit. Visualizar historial de commits.	Repositorio compartido. Guía documentada de Git y GitHub. Desarrollo de casos y publicarlos en el repositorio compartido. Informe de proyecto.	SEMANA 6
2		**Desarrollar guía de primeros pasos con Git. Manejo de branches y merges.** Crear y cambiar entre ramas (branches) utilizando comandos got Branch y git checkout. Fusión de branches. Realizar cambios en una rama y fusionarlos con la rama principal. Uso de git merge para combianr ramas. Resolución de conflictos.		

HT=Horas de desarrollo teórico
HP= Horas de desarrollo práctico
HL= Horas de desarrollo práctico en laboratorio
PC= Horas de práctica de campo

SEMANA	NOMBRE DE LA UNIDAD	ACTIVIDADES PRÁCTICAS DE CAMPO	EVIDENCIA DE PRÁCTICA DE CAMPO (Acción + Producto)	ENTREGA DE EVIDENCIA
3		**Colaboración en GitHub.** Invitar a compañeros a colaborar en el repositorio. Clonar un repositorio compartido. Pull request: realizar cambios en una rama y crear un Pull Request en GitHub. Evidenciar revisión de código.		
4		**Mejores prácticas y comandos avanzados.** Convención de nombres, mensajes de commit claros, importancia de branches. Estrategias de ramificación (Git Flow, por ejemplo) Comandos avanzados: git stash, git revert y git Cherry-pick		
5		Desarrollo de práctica de sobrecarga de métodos, manejo de errores y colecciones evidencia aporte individual en el repositorio compartido.		
6		En grupos de Proyecto: Identificar y formular el problema (diagrama de Ishikawa u otro), documentar antecedentes adecuadamente (Cita APA)		
7	II Clases y relaciones entre clases	Identificar restricciones realistas que afectan al proyecto y propone alternativas de solución en base a ellas. Establecer objetivos del proyecto. Alcance de la solución y objetivos.	Presentar informe y software que incluya los requerimientos y formatos de PC descritos + software en repositorio.	Semana 11
8		Documentar al menos 40 requerimientos funcionales. Elaborar historias de usuario.		
9		Elaborar diagrama de clases. Investigar y realizar práctica de manejo de archivos en Java.		
10		Documentar criterios de aceptación de cada requerimiento. Implementar haciendo uso de Java los requerimientos declarados y personalizados en historias de usuario (15 requerimientos)		
11		Desarrollar actividad de responsabilidad social.		
12	III Programación visual y acceso a base de datos	Implementar haciendo uso de Java los requerimientos declarados y personalizados en historias de usuario (15 requerimientos)	Presentar informe y software que incluya los requerimientos y formatos de PC descritos + software en repositorio.	Semana 14
13		Actualización estado de requerimientos y criterios de aceptación. Implementar 10 requerimientos del proyecto y evidenciar el cumplimiento de los criterios de aceptación.		
14		Implementar proyecto final. Actualizar historias de usuario. Actualizar informe final de proyecto.		
15	Retroalimentación y validación del registro de evidencias en Blackboard			
16	Reflexión sobre las actividades de práctica de campo realizadas en el ciclo			

HT=Horas de desarrollo teórico
HP= Horas de desarrollo práctico
HL= Horas de desarrollo práctico en laboratorio
PC= Horas de práctica de campo