

<p style="text-align: center;">UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA</p>  <p style="text-align: center;">UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA</p> <p style="text-align: center;">PROGRAMA ANALÍTICO DE LA UNIDAD DE APRENDIZAJE:</p> <p style="text-align: center;">PROGRAMACION ORIENTADA A OBJETOS</p>	DES:	INGENIERÍA
	Programa Educativo	Ingeniería en Ciencias de la Computación
	Tipo de materia (Obli/Opta):	Obligatoria
	Clave de la materia:	CI475
	Semestre:	4
	Área en plan de estudios (G, E):	Ciencias de la Ingeniería
	Total, de horas por semana:	4
	<i>Teoría: Presencial o Virtual</i>	3
	<i>Laboratorio o Taller:</i>	1
	<i>Prácticas:</i>	0
	<i>Trabajo extra-clase:</i>	0
	Créditos Totales:	4
	Total, de horas semestre (x 16 sem):	64
	Fecha de actualización:	Febrero 2023
	<i>Prerrequisito (s):</i>	Ninguno
<i>Realizado por:</i>	Comité de Rediseño Curricular	

PROPÓSITO DEL CURSO

El curso le proporciona al estudiante los conceptos y técnicas para que sea capaz de escribir programas simples utilizando el paradigma orientado a objetos. El curso lo introduce a la definición y aplicación de los conceptos de herencia, abstracción, encapsulamiento, interfaces, polimorfismo, colecciones, excepciones y concurrencia, generando programas más entendibles, simplificados, de fácil mantenimiento y actualización, con reutilización de código y protección de datos.

COMPETENCIAS (Tipo Y Nombre de la competencias que nutre la materia y a las que contribuye)	DOMINIOS COGNITIVOS (Objetos de estudio, temas y subtemas)	RESULTADOS DE APRENDIZAJE. (Por objeto de estudio).

<p>El curso promueve las siguientes competencias:</p> <p>BÁSICAS:</p> <p>SOLUCIÓN DE PROBLEMAS</p> <ul style="list-style-type: none"> • Aplica las diferentes técnicas de observación para la solución de problemas. • Distingue los diversos tipos de sistemas • Aplica la tecnología a la solución de problemáticas 	<p>I. INTRODUCCION A LA PROGRAMACION ORIENTADA A OBJETOS</p> <ol style="list-style-type: none"> 1.1. Breve historia de los paradigmas de programación 1.2. Comparación entre paradigmas. 1.3. Fundamentos de la programación orientada a objetos. 1.4. Conceptos básicos de la programación orientada a objetos <ol style="list-style-type: none"> 1.4.1.Clase 1.4.2.Objeto 1.4.3.Método 1.4.4.Evento 1.4.5.Programa 1.4.6.Paquete 	<p>Define y analiza los conceptos básicos de la teoría de objetos en el paradigma de programación orientado a objetos.</p>
--	--	--

<ul style="list-style-type: none"> • Aplica el enfoque sistémico en diversos contextos • Crea soluciones innovadoras y utiliza formas no convencionales en la solución de problemas 	<p>1.4.7. Propiedad</p> <p>1.5. Principios fundamentales de la orientación a objetos.</p> <p>1.5.1. Abstracción</p> <p>1.5.2. Encapsulamiento</p> <p>1.5.3. Herencia</p> <p>1.6. Principios extras</p> <p>1.6.1. Polimorfismo</p> <p>1.6.2. Concurrencia</p> <p>1.6.3. Persistencia</p>	
<p>PROFESIONALES:</p> <p>CIENCIAS FUNDAMENTALES DE LA INGENIERÍA</p> <ul style="list-style-type: none"> • Utiliza las matemáticas como herramientas para solución de problemas en ingeniería 	<p>II. HERENCIA, POLIMORFISMO E INTERFACES</p> <p>2.1 Herencia</p> <p>2.1.1. Jerarquía de clases</p> <p>2.1.1.1 Súper Clases</p> <p>2.1.1.2 Sub Clases</p> <p>2.1.2 Derivación de propiedades y constructores de clases</p> <p>2.1.3 Derivación de métodos</p> <p>2.1.4 Herencia simple y múltiple</p> <p>2.2 Polimorfismo</p> <p>2.2.1 Métodos polimórficos</p> <p>2.2.2 Sobrecarga de operadores</p> <p>2.2.2.1 Extensión de clases existentes</p> <p>2.2.2.2 Conversiones de tipos de datos</p> <p>2.3 Clases abstractas</p> <p>2.4 Interfaces</p> <p>2.4.1 Interfaces estándar para añadir Funcionalidad</p> <p>2.4.2 Creación de nuevas interfaces</p>	<p>Enumera y explica los conceptos de herencia, polimorfismo e interfaces en la programación orientada a objetos.</p> <p>Emplea los conocimientos sobre herencia polimorfismo e interfaces para escribir programas simples orientados a objetos.</p>
<p>ESPECIFICAS:</p> <p>FUNDAMENTOS DE CIENCIAS DE LA COMPUTACIÓN</p> <ul style="list-style-type: none"> • Diseña y aplica algoritmos, estructuras y representación de datos para soluciones computacionales • Distingue los fundamentos de un sistema computacional • Aplica las bases de los lenguajes de programación para generar aplicaciones óptimas • Aplica los tópicos de paradigmas de 	<p>III. COLECCIONES</p> <p>3.1 Listas (Lists)</p> <p>3.2 Conjuntos (Sets)</p> <p>3.3 Mapas (Maps)</p> <p>3.4 Métodos de ordenamiento</p> <p>3.4 Implementación de colecciones</p>	<p>Define el concepto de colección en el ámbito de la programación orientada a objetos</p> <p>Desarrolla programas simples empleando el concepto de colección en la programación orientada a objetos</p>

<p>programación: estructurado, orientado a objetos. Lógico, funcional, entre otros para la creación de aplicaciones óptimas</p>		
	<p>IV. MANEJO DE EXCEPCIONES</p> <p>4.1 Tipos de excepciones 4.2 "Lanzando" Excepciones (Throwing Exceptions) 4.3 "Atrapando" Excepciones (Catching Exceptions) 4.4 Extensión de la Clase "Exception"</p>	<p>Ilustra los tipos de excepciones de un programa en el paradigma orientado a objetos.</p> <p>Utiliza el concepto de excepción para escribir programas robustos orientados a objetos</p>
	<p>V. CONCURRENCIA</p> <p>5.1 Procesos e hilos de control 5.1.1 Hilos 5.1.2 Juntas de hilos 5.2 Sincronización 5.2.1 Locks y bloques sincronizados</p>	<p>Define el concepto de concurrencia en la programación orientada a objetos.</p> <p>Desarrolla programas de cómputo concurrente orientados a objetos.</p>

OBJETO DE ESTUDIO	METODOLOGIA (Estrategias, secuencias, recursos didácticos)	EVIDENCIAS DE APRENDIZAJE.
<p>I. INTRODUCCION A LA PROGRAMACION ORIENTADA A OBJETOS</p> <p>II. HERENCIA</p> <p>III. POLIMORFISMO E INTERFACES</p> <p>IV. COLECCIONES</p> <p>V. MANEJO DE EXCEPCIONES</p> <p>V. CONCURRENCIA</p>	<p>Aprendizaje interactivo (exposición del profesor)</p> <p>Grupo de discusión.</p> <p>Auto aprendizaje (búsqueda y análisis de información)</p> <p>Inductivo</p> <ul style="list-style-type: none"> • Observación • Comparación <p>Deductivo</p> <ul style="list-style-type: none"> • Aplicación • Comprobación • Demostración <p>Sintético</p> <ul style="list-style-type: none"> • Recapitulación • Definición • Resumen • Esquemas • Modelos matemáticos • Conclusión. 	<ul style="list-style-type: none"> • Tareas y ejercicios. • Escritura de programas de computo. • Exámenes escritos.

	<p>Material de Apoyo didáctico: Recursos</p> <ul style="list-style-type: none"> • Manual de Instrucción • Materiales gráficos: artículos, libros, diccionarios, etc. • Cañón • Pintarrones • Equipo de computo 	
--	--	--

FUENTES DE INFORMACIÓN (Bibliografía, Direcciones electrónicas)	EVALUACIÓN DE LOS APRENDIZAJES (Criterios e instrumentos)
<ol style="list-style-type: none"> 1. Concepts of Programming Languages. Robert W. Sebesta. Cualquier edición. 2. Como programar en Java. Deitel. Pearson - Prentice Hall. 3. Como programar en C++. Deitel. Pearson - Prentice Hall. 	<p>INSTRUMENTOS:</p> <ul style="list-style-type: none"> • Exámenes • Solución de ejercicios (aplicación de conocimientos) • Lista de cotejo (Respeto y participación al trabajo dentro del salón de clase, interés por la asignatura) <p>CRITERIOS DE DESEMPEÑO:</p> <ul style="list-style-type: none"> • Los programas desarrollados deberán estar completos. • Las tareas se deberán entregar en tiempo y forma <ul style="list-style-type: none"> • Exámenes escritos: <p>Se realizan 3 exámenes escritos durante el semestre y las fechas se establecen por la secretaría académica</p> <p>Se toma en cuenta para integrar calificaciones parciales:</p> <p>Exámenes 60% Tareas y practicas 40%</p> <p>Fecha de exámenes parciales: 1º. Parcial: 2º. Parcial: 3º. Parcial:</p> <p>LAS ACTIVIDADES NO REALIZADAS EN TIEMPO Y FORMA SE CALIFICAN CON CERO.</p> <p>Nota: para acreditar el curso se deberá tener calificación aprobatoria tanto en la teoría como en las prácticas.</p>

Cronograma del Avance Programático

S e m a n a s

Unidades de aprendizaje	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
I. Introducción a la programación orientada a objetos	■	■														
II. Herencia, polimorfismo e interfaces			■	■	■	■	■	■								
III. Colecciones									■	■	■					
IV. Manejo de excepciones												■	■			
V. Concurrencia														■	■	■