UNIVERSIDAD AUTÓNOMA DE CHIAPAS

Facultad de Negocios, Campus IV

Programa descriptivo por unidad de competencia

<table>
<tr><td>Programa educativo</td><td colspan="2">Licenciatura en Ingeniería en Desarrollo y Tecnologías de Software</td><td colspan="2">Modalidad</td><td colspan="2">Presencial</td></tr>
<tr><td>Clave</td><td colspan="2">IS03</td><td colspan="2">H S M</td><td rowspan="2">Horas semestrales</td><td rowspan="2">Créditos</td></tr>
<tr><td rowspan="2">Unidad de competencia</td><td colspan="2" rowspan="2">Programación orientada a objetos</td><td>Teoría</td><td>Práctica</td></tr>
<tr><td>3</td><td>2</td><td>80</td><td>8</td></tr>
<tr><td>Ubicación</td><td colspan="2">Segundo semestre.</td><td colspan="2">Unidades CONAIC</td><td colspan="2">58.67</td></tr>
<tr><td>Prerrequisito</td><td colspan="2">Metodología de la programación.</td><td colspan="2">H S M de cómputo</td><td colspan="2">2</td></tr>
<tr><td>Perfil docente</td><td colspan="6">Contar con título profesional, grado de maestría y preferentemente con grado de doctorado en áreas afines a informática y computación. Demostrar experiencia en docencia en el nivel medio superior o superior mínima de dos años en programación orientada a objetos. Dominar los lenguajes de programación actuales, deseable con una certificación en lenguajes de programación.</td></tr>
<tr><td>Presentación</td><td colspan="6">Proporciona al estudiante las competencias necesarias para abordar el estudio de cualquier lenguaje orientado a objetos, metodología de análisis y diseño orientado a objetos, de los sistemas gestores de bases de datos, y en general de cualquier materia basada en el modelo orientado a objetos. Se ubica en el área de conocimiento de Programación e ingeniería de software y se relaciona con programación avanzada. Atiende al perfil de egreso en el sentido que domine los conocimientos de metodología de análisis y diseño orientado a objetos.</td></tr>
<tr><td>Propósito</td><td colspan="6">Diseñar e implementar objetos de programación que permitan resolver problemas reales y de ingeniería.</td></tr>
<tr><td colspan="7">Competencias genéricas</td></tr>
<tr><td colspan="7">Aplica un pensamiento sistémico y complejo en la construcción de conocimientos y toma de decisiones.
Maneja Tecnologías de la información y comunicación para la gestión y construcción de conocimientos.</td></tr>
<tr><td colspan="7">Competencias disciplinares</td></tr>
<tr><td colspan="7">Aplica un conjunto de metodologías para el desarrollo de productos y servicios de software de base y aplicaciones.
Posee los conocimientos teóricos y prácticos para la construcción conceptual de soluciones de software.</td></tr>
<tr><td colspan="7">Competencias profesionales</td></tr>
<tr><td colspan="7">Aplica metodologías y técnicas de análisis y diseño para el desarrollo de software.</td></tr>
</table>

Mapa de la unidad de competencia

Unidad de competencia	Subcompetencia	Resultado de aprendizaje
Programación orientada a objetos	1. Comprende y describe los conceptos principales del paradigma de programación orientada a objetos.	1.1. Investiga y selecciona en diversas fuentes de información los conceptos principales del paradigma de programación orientado a objetos. 1.2. Identifica ejemplos de la vida real que apliquen o manifiesten dichos conceptos.
	2. Implementa clases y objetos de acuerdo a las reglas de la programación orientada a objetos.	2.1. Programa clases con atributos públicos para exponer y comprender la vulnerabilidad de los datos. 2.2. Instancia objetos para identificar el nacimiento y muerte de los mismos. 2.3. Programa constructores y destructores para las clases, de manera que permitan dar un valor inicial a sus atributos cuando nazcan sus objetos, o liberar recursos cuando nazcan sus objetos, o liberar recursos cuando mueran los mismos.

	3. Implementa la herencia en clases derivadas.	3.1. Analiza analogías taxonómicas de los seres vivos que compartan rasgos comunes por estar relacionados mediante una herencia genética e identificar la especie a la que pertenecen. 3.2. Identifica los atributos y comportamientos propios de una especie que comparten los animales pertenecientes a ella. 3.3. Programa una clase base para una especie de animales con los atributos y comportamientos comunes a todos los animales pertenecientes a ella.
	4. Implementa interfaces y clases polimórficas.	3.1. Analiza clases base que no requieran ser instanciadas, o que carezcan de sentido para ello por ser abstractas. 3.2. Implementa clases abstractas en clases base que no requieran ser instanciadas con al menos un método abstracto para que sea implementado por sus clases derivadas en múltiples formas. 3.3. Programa inter faces para definirlos comportamientos que una clase deberá detener al implementarla.

	5. Identifica, maneja, gestiona y crea las condiciones de error.	5.1. Crea un programa que deliberadamente genere excepciones comunes para identificar: sus nombres, sus causas, su comportamiento, y reporte de error. 5.2. Programa una clase con varios métodos invocándose en cadena, donde el último método genere una excepción para estudiar y comprender la propagación de las mismas.

Cuadro descriptivo por subcompetencia

Subcompetencia	**Comprende y describe los conceptos principales del paradigma de programación orientada a objetos.**	**Número**	**1**
Propósito de la subcompetencia	Comprende, describe y modela los conceptos principales del paradigma de programación orientado a objetos y aplicarlos a situaciones de la vida real.	**Total de horas**	10
Resultado de aprendizaje	1.1. Investiga y selecciona en diversas fuentes de información los conceptos principales del paradigma de programación orientado a objetos.	**Horas asignadas**	4

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual y un cuadro sinóptico del paradigma de POO.	1. Documento con mapa conceptual. 2. Documento con cuadro sinóptico.	5%	1 Clases. 2 Objetos. 3 Abstracción. 4 Modularidad. 5 Encapsulamiento. 6 Herencia. 7 Polimorfismo.

Resultado de aprendizaje	1.2. Identifica ejemplos de la vida real que apliquen o manifiesten dichos conceptos.	**Horas asignadas**	6

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual y un cuadro sinóptico con ejemplos de POO.	1. Documento con mapa conceptual. 2. Documento con cuadro sinóptico. 3. Examen escrito de la unidad de aprendizaje.	5%	1. Lenguaje de modelado unificado: diagrama de clases.

Cuadro descriptivo por subcompetencia

Subcompetencia	**Implementa clases y objetos de acuerdo a las reglas de la programación orientada a objetos.**	**Número**	**2**
Propósito de la subcompetencia	Implementar clases y objetos cumpliendo las reglas de la programación orientada a objetos e implementar constructores y destructores para inicializar atributos y liberar recursos.	**Total de horas**	15
Resultado de aprendizaje	2.1. Programa clases con atributos públicos para exponer y comprender la vulnerabilidad de los datos.	**Horas asignadas**	5

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual sobre las clases. 2. Programa una clase con atributos públicos.	1. Documento con mapa conceptual. 2. Documento o programa de una clase.	10%	1. Declaración de clases. 2. Instanciación de una clase.

Resultado de aprendizaje	2.2. Instancia objetos para identificar el nacimiento y muerte de los mismos.	**Horas asignadas**	5

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual sobre la instancia de objetos 2. Instancia objeto.	1. Documento con Mapa conceptual. 2. Documento o programa con la instancia de objetos.	10%	1. Referencia al objeto actual. 2. Métodos de declaración.

Resultado de aprendizaje	2.3. Programa constructores y destructores para las clases, de manera que permitan dar un valor inicial a sus atributos cuando nazcan sus objetos, o liberar recursos cuando mueran los mismos.	**Horas asignadas**	5

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual sobre constructores y destructores de clases.	1. Documento con mapa conceptual. 2. Documento o programa sobre constructores y destructores de clases. 3. Examen escrito de la subcompetencia.	10%	1. Constructores y destructores: declaración, uso y aplicaciones. 2. Sobrecarga de métodos. 3. Sobrecarga de operadores: Concepto y utilidad, operadores unarios y binarios.

Cuadro descriptivo por subcompetencia

Subcompetencia	**Implementa la herencia en clases derivadas.**	**Número**	**3**
Propósito de la subcompetencia	Implementar la herencia en clases derivadas para reutilizar los miembros de una clase base.	**Total de horas**	18
Resultado de aprendizaje	3.1. Analiza analogías taxonómicas de los seres vivos que compartan rasgos comunes por estar relacionados mediante una herencia genética e identificar la especie a la que pertenecen.	**Horas asignadas**	6

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual y un cuadro sinóptico de analogías taxonómicas.	1. Documento con mapa conceptual. 2. Documento con cuadro sinóptico.	5%	1. Definición: clase base, clase derivada. 2. Clasificación. herencia simple, herencia múltiple. 3. Reutilización de miembros heredados.

Resultado de aprendizaje	3.2. Identifica los atributos y comportamientos propios de una especie que comparten los animales pertenecientes a ella.	**Horas asignadas**	6

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual y un cuadro sinóptico con los atributos y comportamientos de una especie.	1. Documento con mapa conceptual. 2. Documento con cuadro sinóptico.	5%	1. Referencia al objeto de la clase base. 2. Constructores y destructores en clases derivadas. 3. Redefinición de métodos en clases derivadas.

Resultado de aprendizaje	3.3. Programa una clase base para una especie de animales con los atributos y comportamientos comunes a todos los animales pertenecientes a ella.	**Horas asignadas**	6

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Implementa la herencia en una clase.	1. Programa con la implementación de una clase base para una especie de animales. 2. Examen escrito de la subcompetencia.	10%	1. Herencia en una clase.

Cuadro descriptivo por subcompetencia

Subcompetencia	**Implementa interfaces y clases polimórficas.**	**Número**	**4**
Propósito de la subcompetencia	Implementar interfaces y clases polimórficas.	**Total de horas**	18
Resultado de aprendizaje	4.1. Analiza clases base que no requieran ser instanciadas, o que carezcan de sentido para ello por ser abstractas.	**Horas asignadas**	6

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Construye un mapa conceptual sobre las clases abstractas.	1. Documento con mapa conceptual.	5%	1. Definición. 2. Clases abstractas: definición, métodos abstractos, implementación de clases abstractas, modelado de clases abstractas.

Resultado de aprendizaje	4.2 Implementa clases abstractas en clases base que no requieran ser instanciadas con al menos un método abstracto para que sea implementado por sus clases derivadas en múltiples formas.	**Horas asignadas**	6

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Implementa clases abstractas en clases base.	1. Programa con la implementación de clase.	10%	1. Interfaces: definición, implementación de interfaces, herencia de interfaces. 2. Variables polimórficas (plantillas): definición, uso y aplicaciones.

Resultado de aprendizaje	4.3. Programa interfaces para definir los comportamientos que una clase deberá de tener al implementarla.	**Horas asignadas**	6

Actividades de evaluación	Evidencias a recopilar	%	Contenido
1. Programa interfaces para los comportamientos de una clase.	1. Programa con las interfaces de comportamientos de una clase. 2. Examen escrito de la subcompetencia.	5%	1. Reutilización de código.

Cuadro descriptivo por subcompetencia

Subcompetencia	**Identifica, maneja, gestiona y crea las condiciones de error.**			**Número**	**5**
Propósito de la subcompetencia	Identificar, manejar, gestionar y crear las condiciones de error que interrumpan el flujo normal de ejecución de un programa.			**Total de horas**	19
Resultado de aprendizaje	5.1. Crea un programa que deliberadamente genere excepciones comunes para identificar: sus nombres, sus causas, su comportamiento, y reporte de error.			**Horas asignadas**	8
Actividades de evaluación	**Evidencias a recopilar**	**%**	**Contenido**		
1. Crea un programa para generar excepciones.	1. Programa que genera excepciones.	10%	1. Excepciones. 2. Tipos de excepciones.		
Resultado de aprendizaje	5.2 Programa una clase con varios métodos invocándose en cadena, donde el último método genere una excepción para estudiar y comprender la propagación de las mismas.			**Horas asignadas**	11
Actividades de evaluación	**Evidencias a recopilar**	**%**	**Contenido**		
1. Programa una clase con métodos invocados en cadena.	1. Programa con una clase con métodos invocados en cadena. 2. Examen escrito de la subcompetencia.	10%	1. Propagación de excepciones. 2. Gestión de excepciones: manejo de excepciones, lanzamiento de excepciones. 3. Creación y manejo de excepciones definidas por el usuario.		

Actitudes y valores	Analítico. Ordenado. Coherente. Proactivo. Asertivo.

Recursos, materiales y equipo didáctico	
Recursos didácticos	**Equipo de apoyo didáctico**
Antologías. Diapositivas. Videos.	Proyector de video. Software especializado

Fuentes de información

Bibliografía básica:

Barnes, D. J. (2012). *Programación orientada a objetos con java*. México:Prentice-Hall.
Deitel (2012). *Java como programar* (9a. ed.). México: Pearson.
Thomas, W. (2008). *Programación en Java*. México: McGraw Hill.

Bibliografía complementaria:

López R., L. (2011). *Programación estructurada y orientada a objetos* (3a. ed.). México: Alfaomega.
Bell, D. (2011). *Java para estudiantes* (6a. ed.). México: Pearson.

Recursos digitales:

Ninguno.