UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

COORDINACIÓN GENERAL DE FORMACIÓN PROFESIONAL

PROGRAMA DE UNIDAD DE APRENDIZAJE

I. DATOS DE IDENTIFICACIÓN

1. Unidad Académica: Facultad de Ingeniería, Arquitectura y Diseño, Ensenada; Facultad Ciencias Químicas e Ingeniería, Tijuana; y Facultad de Ciencias de la Ingeniería y Tecnología, Valle de las Palmas

2. Programa Educativo: Ingeniero en Software y Tecnologías Emergentes

3. Plan de Estudios: 2022-1

4. Nombre de la Unidad de Aprendizaje: Lenguajes de Programación Orientada a Objetos

5. Clave: 40006

6. HC: 02 **HT:** 01 **HL:** 02 **HPC:** 00 **HCL:** 00 **HE:** 02 **CR:** 07

7. Etapa de Formación a la que Pertenece: Disciplinaria

8. Carácter de la Unidad de Aprendizaje: Obligatoria

9. Requisitos para Cursar la Unidad de Aprendizaje: Ninguno

Equipo de diseño de PUA
Sergio Omar Infante Prieto
José Manuel Villegas Izaguirre
J. Reyes Juárez Ramírez

Vo.Bo. de subdirector(es) de Unidad(es) Académica(s)
Humberto Cervantes De Ávila
Daniela Mercedes Martínez Platas
Noemí Hernández Hernández

Fecha: 20 de febrero de 2021

II. PROPÓSITO DE LA UNIDAD DE APRENDIZAJE

La unidad de aprendizaje brinda los conocimientos sobre el paradigma de programación orientada a objetos, lo que permite al profesionista analizar y diseñar modelos de clases para resolver problemas de procesamiento de información, utilizando la sintaxis de lenguajes de programación orientada a objetos de actualidad, que son de carácter universal y permitan desarrollar aplicaciones que corren en cualquier plataforma logrando que estas sean eficientes.

Se ubica en la etapa disciplinaria, con carácter de obligatoria y forma parte del área de Métodos y Tecnologías de Software del programa educativo Ingeniero en Software y Tecnologías Emergentes.

III. COMPETENCIA GENERAL DE LA UNIDAD DE APRENDIZAJE

Aplicar el paradigma orientado a objetos (OO), para dar solución a problemas de procesamiento de información, empleando un lenguaje de modelado de sistemas y lenguajes que permiten la programación orientada a objetos, con actitud analítica y creativa.

IV. EVIDENCIA(S) DE APRENDIZAJE

Prototipo de proyecto de tamaño mediano que incluya el diseño arquitectónico de clases en un lenguaje de modelado de objetos, su código fuente en un lenguaje de programación orientada a objetos y el código ejecutable.

V. DESARROLLO POR UNIDADES
UNIDAD I. Paradigma de orientación a objetos

Competencia:
Distinguir los elementos del paradigma de la programación orientada a objetos, a través del estudio de sus principales conceptos y ventajas, para comprender la utilidad en el proceso de desarrollo de software, con actitud analítica y reflexiva.

Contenido: **Duración:** 4 horas

1.1. El paradigma de OO, sus principales conceptos y ventajas
1.2. El paradigma de OO y las fases de desarrollo de software
1.3. Lenguajes de programación orientada a objetos
1.4. Modelado orientado a objetos

UNIDAD II. Abstracción y encapsulamiento

Competencia:
Diseñar los elementos de los objetos, mediante la representación de los mismos y su sintaxis en los lenguajes de programación orientada a objetos, para dar solución a problemas de procesamiento de información, con actitud analítica, propositiva y metódica.

Contenido: **Duración:** 6 horas

2.1. Clases
- 2.1.1. Definición de clase.
- 2.1.2. Miembros de una clase: variables de clase, métodos
- 2.1.3. Declaración de atributos o datos y su uso
- 2.1.4. Modificadores aplicables en la declaración de clases

2.2. Métodos y operaciones
- 2.2.1. Diferencia entre operación y método
- 2.2.2. Protocolo de un método
- 2.2.3. Tipos de métodos: métodos de instancia y métodos de clase
- 2.2.4. Invocación de métodos y pase de parámetros a métodos
- 2.2.5. Métodos constructores
- 2.2.6. Sobrecarga de métodos

2.3. Objetos
- 2.3.1 Definición de objeto
- 2.3.2 Declaración e inicialización de objetos
- 2.3.3 Tipos de objetos

2.4. Niveles de protección o acceso
- 2.4.1. Modificadores de acceso para métodos y atributos
- 2.4.2. Encapsulamiento

2.5. Envío de mensajes

2.6. Agregación y Composición
- 2.6.1. Composición fuerte
- 2.6.2. Composición débil
- 2.6.3. Clases anidadas

UNIDAD III. Jerarquía de clases y herencia

Competencia:
Aplicar los conceptos de la jerarquía de clases, mediante representaciones de herencia, clases abstractas e interfaces, para dar solución a problemas de procesamiento de información, con una actitud analítica, creativa y metódica.

Contenido: **Duración:** 6 horas

3.1. Herencia simple
- 3.1.1. Herencia de los miembros de la clase padre

3.2. Especialización
- 3.2.1. Agregación de comportamiento en la clase derivada
- 3.2.2. Sobre escritura de atributos y métodos
- 3.2.3. Invocación de constructor y métodos sobrescritos de la clase padre

3.3. Clases abstractas
- 3.3.1. Métodos abstractos y clases abstractas
- 3.3.2. Interfaces

3.4. Herencia múltiple
- 3.4.1. Herencia múltiple entre clases
- 3.4.2. Herencia múltiple entre clases e interfaces
- 3.4.3. Herencia múltiple entre interfaces
- 3.4.4. Jerarquía de herencia múltiple

UNIDAD IV. Polimorfismo y reutilización de código

Competencia:
Aplicar el polimorfismo en objetos, mediante el empleo de los principios y estrategias de diseño, para facilitar la reutilización de código y el mantenimiento del software, con una actitud analítica, metódica y propositiva.

Contenido: **Duración:** 8 horas

4.1. Polimorfismo
- 4.1.1. Definición de polimorfismo
- 4.1.2. Polimorfismo estático (Sobrecarga de métodos)
- 4.1.3. Polimorfismo dinámico (Sobre-escritura de métodos y enlace dinámico)

4.2. Cohesión
- 4.2.1. Conceptos de cohesión de clases y entre clases
- 4.2.2. Niveles de cohesión
- 4.2.3. Convergencia y divergencia de métodos
- 4.2.4. Convergencia y divergencia de clases
- 4.2.5. Principio de alta cohesión: Especialización/enfoque
- 4.2.6. Baja cohesión: dificultad para entendimiento, mantenimiento, reutilización

4.3. Acoplamiento
- 4.3.1. Acoplamiento entre clases
- 4.3.2. Niveles de acoplamiento
- 4.3.3. Dependencia entre clases
- 4.3.4. La dependencia de datos y de comportamiento
- 4.3.5. El principio de la separación
- 4.3.6. Principio de bajo acoplamiento: Independencia
- 4.3.7. Alto acoplamiento: dificultad para entendimiento, mantenimiento, reutilización

UNIDAD V. Persistencia y concurrencia

Competencia:
Aplicar la persistencia y concurrencia de objetos, empleando mecanismos de escritura y encriptación para lograr que perduren en el tiempo y a través de espacios de almacenamiento, cuidando el acceso concurrente, manejo de información según el lenguaje de programación empleado con actitud crítica y propositiva.

Contenido: **Duración:** 8 horas

5.1. Persistencia
- 5.1.1. Definición de persistencia
- 5.1.2. Persistencia de en memoria
- 5.1.3. Persistencia en el tiempo
- 5.1.4. Persistencia de objetos

5.2. Diseño de estructuras persistentes
- 5.2.1. Estructuras de datos persistentes
- 5.2.2. Encriptación de datos
- 5.2.3. Persistencia en medios de almacenamiento
- 5.2.4. Bases de datos orientadas a objetos

5.3. Concurrencia
- 5.3.1. Definición de concurrencia
- 5.3.2. Objetos sincrónicos versus asincrónicos
- 5.3.3. Compartición de datos y de bloques de código
- 5.3.4. Sincronización

5.4. Patrones de diseño concurrente
- 5.4.1. Aceptor-Conector
- 5.4.2. Reactor-Proactor
- 5.4.3 .Objetos activos
- 5.4.4. Monitores
- 5.4.5. Líder y seguidores

VI. ESTRUCTURA DE LAS PRÁCTICAS DE TALLER				
No.	**Nombre de la Práctica**	**Procedimiento**	**Recursos de Apoyo**	**Duración**
UNIDAD I				
1	**Esquematización de los conceptos de POO**	1. Atiende las orientaciones del profesor para clasificar conceptos mediante esquemas gráficos. 2. Investiga los principales conceptos del paradigma de orientación a objetos. 3. Elabora un mapa conceptual con los conceptos investigados. 4. Investiga los principales lenguajes que soportan el paradigma de orientación a objetos y sus características. 5. Elabora un cuadro sinóptico con la información obtenida. Entrega cuadro sinóptico al profesor para obtener retroalimentación	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet	1 hora
2	**Esquematización de los elementos de modelado OO**	1. Atiende las orientaciones del profesor para clasificar conceptos mediante esquemas gráficos. 2. Investiga los principales elementos de un lenguaje de modelado que soporte el paradigma de orientación a objetos. 3. Elabora un cuadro sinóptico con la información obtenida. 4. Entrega cuadro sinóptico al profesor para obtener retroalimentación.	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet	1 hora

UNIDAD II				
3	**Esquematización de los elementos de una clase**	1. Atiende las orientaciones del profesor para clasificar conceptos mediante esquemas gráficos. 2. Investiga los elementos de los objetos de una clase. 3. Elabora un mapa conceptual de los elementos y miembros de una clase. 4. Entrega el resultado del mapa conceptual al profesor para obtener retroalimentación.	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	2 horas
4	**Modelado de clases**	1. Atiende las orientaciones del profesor para modelar clases y su explicación del concepto de abstracción, encapsulamiento, modelado de clases, su representación en un lenguaje de modelado y su implementación en diferentes lenguajes de programación orientado a objetos. 2. Analiza el concepto o problema. 3. Desarrolla el diagrama de clases que representa la abstracción del problema, utilizando el modelo 4. Entrega el resultado del ejercicio al profesor para obtener retroalimentación.	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	1 hora
UNIDAD III				
5	**Modelado de herencia simple entre varias entidades**	1. Atiende las orientaciones del profesor para modelar clases y su explicación sobre cómo	• Apuntes del curso • Literatura a consultar • Computadora con conexión a	2 horas

		reproducir las relaciones entre varias entidades a través de la herencia. 2. Analiza un problema y lo representa en un diagrama de clases considerando la herencia para resolver el ejercicio. 3. Elabora el modelo para reproducir las relaciones entre varias entidades del mundo real. 4. Entrega el resultado del modelo al profesor para obtener retroalimentación.	Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	
6	**Modelado de herencia múltiple entre varias entidades**	1. Atiende las orientaciones del profesor para modelar clases y su explicación sobre cómo reproducir las relaciones entre varias entidades a través de la herencia múltiple. 2. Analiza un problema y lo representa en un diagrama de clases considerando herencia múltiple. 3. Hace una tabla comparativa de los lenguajes de programación que permiten la herencia múltiple. 4. Elabora un documento donde escriba el modelo de clase que analizo y colocando la tabla comparativa de los lenguajes de programación que permiten la herencia múltiple. 5. Entrega el documento realizado al profesor para obtener retroalimentación.	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	2 horas

UNIDAD IV				
7	**Análisis de polimorfismo**	1. Atiende las orientaciones del profesor para modelar clases y su explicación sobre cómo facilitar la reutilización de código y el mantenimiento del software a través del polimorfismo. 2. Analiza un problema para resolverlo y lo representa en un diagrama de clases considerando la relación polimórfica entre las diferentes entidades relacionadas con el problema. 3. Elabora un documento donde escriba la solución al problema analizado. 4. Elabora una tabla comparativa sobre cómo se maneja el polimorfismo en diferentes lenguajes. 5. Entrega el documento realizado junto con la tabla comparativa sobre cómo se maneja el polimorfismo en diferentes lenguajes al profesor para obtener retroalimentación.	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	2 horas
8	**Modelado de la cohesión y el acoplamiento**	1. Atiende las orientaciones del profesor para modelar clases y su explicación sobre los conceptos y principios de la cohesión y el acoplamiento entre las clases. 2. Analiza un problema para resolverlo y representarlo en un diagrama de clases,	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	2 horas

		considerando principios para lograr una alta cohesión y un bajo acoplamiento relacionadas con el problema. 3. Elabora un documento donde escriba la solución al problema analizado y describiendo la forma en la que se logró alta cohesión y bajo acoplamiento. 4. Entrega el documento realizado al profesor para obtener retroalimentación.		
UNIDAD V				
9	**Implementación de la persistencia en varios lenguajes**	1. Atiende las orientaciones del profesor con respecto a la persistencia en objetos. 2. Investiga cómo se maneja la persistencia en varios lenguajes de programación orientada a objetos. 3. Elabora una infografía para aplicar la persistencia en varios lenguajes de programación. 4. Entrega la infografía para aplicar la persistencia en varios lenguajes de programación al profesor para obtener retroalimentación.	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	1 hora
10	**Implementación de la concurrencia en varios lenguajes**	1. Atiende las orientaciones del profesor con respecto a la concurrencia y aspectos de diseño de aplicaciones en las que se utilice concurrencia. 2. Investiga cómo se maneja la concurrencia en varios lenguajes de programación orientada a objetos.	• Apuntes del curso • Literatura a consultar • Computadora con conexión a Internet • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Compilador	2 horas

		3. Elabora una infografía con la información sobre el manejo de concurrencia. 4. Analiza un problema en el que se requiera manejo de concurrencia, para resolverlo y representarlo en un diagrama de clases. 5. Elabora un documento donde escriba la solución al problema analizado y describiendo la forma en la que se manejara la concurrencia. 6. Entrega la infografía con la información sobre el manejo de concurrencia y el documento donde escriba la solución al problema analizado al profesor para obtener retroalimentación.		

VI. ESTRUCTURA DE LAS PRÁCTICAS DE LABORATORIO				
No.	**Nombre de la Práctica**	**Procedimiento**	**Recursos de Apoyo**	**Duración**
UNIDAD I				
1	**Instalación de compiladores y configuración de entorno de desarrollo**	1. Atiende las orientaciones del profesor con respecto a la instalación de compiladores y configuración de los entornos de desarrollo. 2. Descarga los compiladores de los lenguajes de programación seleccionados por el profesor. 3. Descarga los entornos de	• Computadora con conexión a internet • Procesador de textos	2 horas

		desarrollo para los lenguajes de programación seleccionados por el profesor. 4. Instala y configura los compiladores y entornos. 5. Hace pruebas de los entornos y compiladores con diversos programas ejemplo. 6. Elabora un reporte de práctica de la configuración de compiladores en los entornos de desarrollo. 7. Entrega el reporte de práctica al profesor para obtener retroalimentación.		
2	**Instalación y configuración de entornos de modelado OO**	1. Atiende las orientaciones del profesor para la selección, instalación y configuración de entornos de modelado OO. 2. Descarga software para crear los principales diagramas utilizados para el modelado orientado a objetos. 3. Crea los diagramas de clases para los ejemplos proporcionados por el profesor. 4. Elabora un reporte de práctica con respecto al uso de los programas para diagramación. Entrega los diagramas desarrollados y el reporte de práctica al profesor para obtener retroalimentación.	• Computadora con conexión a internet • Procesador de textos	2 horas
UNIDAD II				
3	**Implementación básica de clases y objetos**	1. Atiende las orientaciones del profesor para la implementación básica de	• Computadora con conexión a Internet. • Compiladores de Lenguajes de	4 horas

		clases. 2. Investiga la sintaxis para la codificación de clases y creación de objetos en diferentes lenguajes. 3. Codifica, en diversos lenguajes de programación, clases a partir de diagramas proporcionados por el profesor. 4. Crea y prueba objetos a partir de las clases codificadas. 5. Elabora un reporte de práctica. Entrega los programas y el reporte de práctica al profesor para obtener retroalimentación.	programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación • Procesador de palabras	
4	**Modelado de clases para un problema del mundo real**	1. Atiende las orientaciones del profesor para realizar abstracción de entidades del mundo real. 2. Analiza un problema indicado por el profesor. 3. Desarrolla los diagramas necesarios para representar la solución al problema. 4. Codifica los diagramas desarrollados en diversos lenguajes de programación. 5. Compila y prueba el código generado. 6. Elabora un reporte de práctica. 7. Entrega los diagramas, programas y el reporte de práctica al profesor para obtener retroalimentación.	• Computadora con conexión a Internet. • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación • Procesador de palabras	2 horas
UNIDAD III				

5	**Modelado e implementación de relaciones de herencia simple**	1. Atiende las orientaciones del profesor para modelar clases y su explicación sobre cómo reproducir las relaciones entre varias entidades a través de la herencia simple. 2. Analiza un problema considerando la herencia para resolver la práctica. 3. Codifica la solución del problema en los lenguajes de programación orientada a objetos indicados por el profesor. 4. Elabora un reporte de práctica de la solución del problema mediante los lenguajes de programación orientada a objetos. 5. Entrega reporte de práctica y los programas al profesor para obtener retroalimentación.	• Computadora con conexión a Internet. • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación • Procesador de palabras	4 horas
6	**Modelado e implementación de relaciones de herencia múltiple**	1. Atiende las orientaciones del profesor para modelar clases y su explicación sobre cómo reproducir las relaciones entre varias entidades a través de la herencia múltiple. 2. El alumno analiza un problema considerando herencia múltiple. 3. Codifica la solución del problema en los lenguajes de programación orientada a objetos que la permitan. 4. Elabora un reporte de práctica de la solución del problema mediante los lenguajes de	• Computadora con conexión a Internet. • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación • Procesador de palabras	4 horas

		programación orientada a objetos. 5. Entrega reporte de práctica y los programas al profesor para obtener retroalimentación.		
UNIDAD IV				
7	**Diseño e implementación de polimorfismo**	1. Atiende las orientaciones del profesor para resolver un problema y facilitar la reutilización de código y el mantenimiento del software a través del polimorfismo. 2. Analiza un problema para resolverlo y poder reutilizar código y facilitar el mantenimiento en el futuro. 3. Codifica la solución del problema en los lenguajes de programación orientada aplicando polimorfismo. 4. Elabora un documento donde escriba la solución al problema analizado. 5. Elabora una tabla comparativa sobre cómo se maneja el polimorfismo en diferentes lenguajes. 6. Entrega reporte de práctica y los programas al profesor para obtener retroalimentación.	• Computadora con conexión a Internet. • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación • Procesador de palabras	4 horas
8	**Diseño e implementación de cohesión y acoplamiento**	1. Atiende las orientaciones del profesor para resolver un problema aplicando los principios de la cohesión y el acoplamiento entre las clases. 2. Analiza un problema para resolverlo considerando	• Computadora con conexión a Internet. • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación	4 horas

		principios para lograr una alta cohesión y un bajo acoplamiento relacionadas con el problema. 3. Codifica la solución del problema en los lenguajes de programación orientada aplicando principios para lograr una alta cohesión y un bajo acoplamiento relacionadas con el problema. 4. Elabora un documento donde escriba la solución al problema analizado. 5. Entrega reporte de práctica y los programas al profesor para obtener retroalimentación.	• Procesador de palabras	
UNIDAD V				
9	**Diseño e implementación de persistencia de objetos**	1. Atiende las orientaciones del profesor para diseñar e implementar la persistencia de objetos. 2. Analiza un problema que involucre el manejo de persistencia en objetos, y representa la solución en los diagramas pertinentes 3. Codifica los diagramas desarrollados en diversos lenguajes de programación, implementando la persistencia de objetos correspondiente. 4. Compila y prueba el código generado. 5. Elabora un reporte de práctica. Entrega los diagramas, programas y el reporte de	• Computadora con conexión a Internet. • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación • Procesador de palabras	2 horas

		práctica al profesor para obtener retroalimentación.		
10	**Diseño e implementación de concurrencia de objetos**	1. Atiende las orientaciones del profesor para diseñar e implementar la concurrencia de objetos. 2. Analiza un problema en el que se requiera manejo de concurrencia y representa la solución en los diagramas correspondientes. 3. Elabora un documento donde escriba la solución al problema analizado y describiendo la forma en la que se maneja la concurrencia. 4. Codifica, en diversos lenguajes de programación, los diagramas desarrollados, implementando los mecanismos para manejo de concurrencia apropiados. 5. Compila y prueba el código generado. 6. Elabora un reporte de práctica. 7. Entrega los diagramas, programas y el reporte de práctica al profesor para obtener retroalimentación.	• Computadora con conexión a Internet. • Compiladores de Lenguajes de programación orientados a objetos • Editor de texto/ IDEs • Software de diagramación • Procesador de palabras	4 horas

VII. MÉTODO DE TRABAJO

Encuadre: El primer día de clase el docente debe establecer la forma de trabajo, criterios de evaluación, calidad de los trabajos académicos, derechos y obligaciones docente-alumno.

Estrategia de enseñanza (docente):

- Presenta información sobre los conceptos básicos
- Presenta ejercicios prácticos relacionados con las temáticas
- Dirige, supervisa y retroalimenta las prácticas de taller
- Propicia la participación activa de los estudiantes
- Revisa y evalúa reportes de prácticas y actividades
- Elabora y aplica evaluaciones

Estrategia de aprendizaje (alumno):

- Investiga y analiza información sobre conceptos básicos
- Resuelve ejercicios prácticos proporcionados por el profesor
- Realiza las prácticas de taller y de laboratorio
- Participa activamente en clase
- Elabora y entrega reportes de prácticas
- Trabaja en equipo
- Elabora y entrega actividades y prácticas en tiempo y forma
- Presenta evaluaciones

VIII. CRITERIOS DE EVALUACIÓN

La evaluación será llevada a cabo de forma permanente durante el desarrollo de la unidad de aprendizaje de la siguiente manera:

Criterios de acreditación

- Para tener derecho a examen ordinario y extraordinario, el estudiante debe cumplir con los porcentajes de asistencia que establece el Estatuto Escolar vigente.
- Calificación en escala del 0 al 100, con un mínimo aprobatorio de 60.

Criterios de evaluación

- Evaluaciones parciales……. 20%
- Prácticas de laboratorio…… 25%
- Prácticas de taller……15%
- Prototipo de proyecto……...…..40%

Total……...…….100%

IX. REFERENCIAS

Básicas

Forouzan, B.A. y Gilberg, R.F. (2020). *C++ Programming: An Object Oriented Approach*, Estados Unidos: McGraw-Hill Education.

Jeya-Mala, D. y Geetha, S. (2013). *Object Oriented Analysis andDesign Using UML*. Estados Unidos: McGraw-Hill Education. [clásica]

Price, M.J.(2020). *C# 9 and .NET 5 – Modern Cross-Platform Development*, Reino Unido: Packt Publishing

Schildt, H. (2019). *Java: The Complete Reference, Eleventh Edition*, Estados Unidos: McGraw-Hill Education.

Weisfeld, M (2019). *The Object-Oriented Thought Process*, Estados Unidos: Pearson Education.

Complementarias

Booch, G., Maksimchuk, R.A., Engle, M.W., Young, B.J, Conallen, J., y Houston, K. (2007). *Object-Oriented Analysis and Design with Applications (3ª ed.)*. Estados Unidos: Addison Wesley.[clásica]

Baesens, B., Backiel, A. y vanden Broucke, S. (2015). *Beginning Java Programming: The Object-Oriented Approach*. Reino Unido: Wrox press

Clark, D. (Marzo, 2013). Beginning C# Object-Oriented Programming (Expert's Voice in .NET) (2ª ed.). Apress.

Deitel, P., y Deitel, H. (2017). *Java 9 For Programmers (4ª ed.)*, Estados Unidos: Pearson Education

Dusty, P. (2015). *Python 3 Object Oriented Programming. (2a ed.)*. Reino Unido: Packt Publishing Ltd.

Gervais L. (2019), *Aprender la Programación Orientada a Objetos con el lenguaje Java,* España: Eni.

Lafore, R., Object-Oriented Programming in C++ (4th Ed). Estados Unidos: Sams Publishing Company

Oracle Corporation (2021), Java Platform, Standard Edition & Java Development Kit, Version 11 API Specification, Estados Unidos: Oracle Corporation, Recuperado de: https://docs.oracle.com/en/java/javase/11/docs/api/index.html

X. PERFIL DEL DOCENTE

El docente que imparta la unidad de aprendizaje de Lenguajes de Programación Orientada a Objetos debe contar con título de Ingeniero de Software, Licenciado en Ciencias de la Computación o afín, preferentemente con estudios de posgrado en ciencias de la computación. Contar con dos años de experiencia docente. Ser proactivo, analítico y que fomente el trabajo en equipo.

UNIVERSIDAD AUTÓNOMA
DE BAJA CALIFORNIA
REGISTRADO
07 OCT 2021
REGISTRADO
COORDINACIÓN GENERAL
DE FORMACIÓN PROFESIONAL