Nombre de la asignatura									Programación II	Clave de asignatura C0108064
Área de formación	Docencia frente a grupo según SATCA				Trabajo de Campo Supervisado según SATCA				Carácter de la asignatura	
	HCS	HPS	TH	C	HTCS	TH	C	TC		
General	2	2	4	4	0	0	0	4	(X) Obligatoria	( ) Optativa

SERIACIÓN		
Explícita		Implícita
Asignaturas antecedentes	Asignaturas subsecuentes	Conocimientos previos
Programación I	Programación III	

PROPÓSITO DE LA ASIGNATURA
Desarrollar programas aplicando técnicas y paradigmas de la programación orientada a objetos para la solución de problemas.

COMPETENCIAS A DESARROLLAR	
Genéricas	**Específicas**
Capacidad de análisis y síntesis. Uso de las TIC. Capacidad de trabajar en equipos interdisciplinarios. Compromiso ético. Capacidad de aplicar los conocimientos en la práctica. Resolución de problemas. Habilidades de investigación. Trabajo autónomo. Diseño y gestión de proyectos.	Desarrollar sistemas de software integrando tecnologías para la solución de problemas automatizando los procesos operativos, flujo de información y toma de decisiones en las organizaciones con un enfoque sistémico bajo estándares internacionales. Instrumentar proyectos tecnológicos bajo metodologías innovadoras para el desarrollo de software para la eficiencia de los procesos en las organizaciones tomando en cuenta los requerimientos de los diversos sectores productivos.

UNIDAD No. 1	Conceptos básicos de programación orientada a objetos	Horas estimadas para cada unidad
		12

CONTENIDOS		
Conceptuales	**Aprendizajes esperados**	**Evidencias de aprendizaje**
1.1. Introducción a la POO. 1.2. Diseño de diagramas de clase en UML. 1.3. Clases y objetos. 1.4. Atributos y métodos. 1.5. Encapsulación. 1.6. Modificadores de acceso. 1.7. Constructores.	Elabora diagramas de clases utilizando UML. Realiza programas aplicando los elementos básicos de la POO para resolver problemas.	Diagrama UML de clases de la solución de problemas. Programas codificados utilizando un lenguaje de programación orientado a objetos.

UNIDAD No. 2	Herencia	Horas estimadas para cada unidad
		8

CONTENIDOS		
Conceptuales	**Aprendizajes esperados**	**Evidencias de aprendizaje**
2.1. Herencia. 2.2. Diseño del diagrama de clases en UML usando herencia. 2.3. Implementación de programas usando herencia.	Desarrolla diagramas de clases UML que modelen problemas empleando herencia. Realiza programas aplicando la herencia de la POO para resolver problemas.	Diagramas UML de clases usando la herencia. Programas codificados usando la herencia.

UNIDAD No.3	Polimorfismo	Horas estimadas para cada unidad
		10

CONTENIDOS		
Conceptuales	**Aprendizajes esperados**	**Evidencias de aprendizaje**
3.1. Polimorfismo. 3.2. Diseño del diagrama de clases en UML usando polimorfismo. 3.3. Implementación de programas usando polimorfismo.	Desarrolla diagramas de clases UML que modelen problemas empleando el polimorfismo. Desarrolla programas aplicando el polimorfismo de la POO para resolver problemas.	Diagramas UML de clases usando el polimorfismo. Programas codificados usando el polimorfismo.

UNIDAD No. 4	Clases abstractas e interfaces.	Horas estimadas para cada unidad
		14

CONTENIDOS		
Conceptuales	**Aprendizajes esperados**	**Evidencias de aprendizaje**
4.1. Clases abstractas. 4.2. Interfaces. 4.3. Principio de diseño *Program-to-an-interface.* 4.4. Diseño del diagrama de clases en UML usando clases abstractas e interfaces. 4.5. Implementación de programas usando el principio de diseño *Program-to-an-interface.*	Elabora diagramas de clases UML que modelen problemas con el uso de las clases abstractas y las interfaces. Desarrolla programas que implementen las clases abstractas y las interfaces para la solución de problemas.	Diagramas UML usando las clases abstractas y las interfaces. Programas codificados usando las clases abstractas y las interfaces.

UNIDAD No. 5	Conceptos avanzados de programación orientada a objetos	Horas estimadas para cada unidad
		20

CONTENIDOS

Conceptuales	Aprendizajes esperados	Evidencias de aprendizaje
5.1. Clases genéricas. 5.2. Relaciones entre clases. 5.3. Manejo de excepciones. 5.4. Colecciones.	Desarrolla programas utilizando las clases genéricas en el manejo de datos, y que además validen la captura y los procesos por medio del manejo de las excepciones. Desarrolla programas utilizando las Colecciones para resolver problemas.	Programas codificados usando las clases genéricas, relaciones entre clases, manejo de las excepciones y colecciones. Diagrama UML del diseño de clases de un proyecto final. Programa codificado del proyecto final.

Contenidos procedimentales	Contenidos actitudinales
Analiza los requerimientos. Diseña diagramas UML de clases. Codifica en un lenguaje de programación orientado a objetos.	Responsabilidad en la entrega de sus trabajos. Disposición para trabajar en equipo. Rigor en el manejo de la información. Respeto a las propuestas de sus compañeros.

Metodología para la construcción del conocimiento

Actividades de aprendizaje con el docente	Actividades de aprendizaje autónomo
Exposición de temas. Elaboración de diagramas de clases utilizando una herramienta UML. Prácticas guiadas y supervisadas en el laboratorio cómputo. Instalación y uso de un entorno de desarrollo integrado.	Investigación documental. Uso de una herramienta UML. Uso del entorno de desarrollo integrado. Desarrollo de programas. Prácticas autónomas. Reproducción video tutoriales.

Evidencias de desempeño

Acreditación	Evaluación	Calificación
Cumplir con lo establecido en el Reglamento Escolar vigente. Entrega de evidencias de aprendizaje.	Al final de cada unidad. Al final del curso.	15% Diagramas UML de clases. 45% Programas codificados utilizando un lenguaje de programación orientado a objetos. 10% Diagrama UML del diseño de clases de un proyecto final. 30% Programa codificado del proyecto final.

FUENTES DE APOYO Y CONSULTA
BÁSICA

1. Flores, A. (2012). Programación Orientada a Objetos Usando Java. Colombia: Ecoe Ediciones
2. Garcìa,L, Luis Fernando. (2010). Todo lo básico que debería saber: sobre programación orientada a objetos en Java.Colombia**:** Ediciones de la U*
3. Liguori, R., Finegan, E. (2010). Sun Certified Java Associate. USA:McGraw Hill *
4. López, L. (2013). Metodología de Programación Orientada a Objetos. México: Alfaomega.
5. Object Management Group (2010). UML resource page. [en línea] URL http://www.uml.org. *
6. Savitch, Walter. Absolute Java with Student Resource Disk (2nd Edition), 2005 *
7. Savitch, Walter. Java: an introduction to computer science and programming.
8. Dale, N.,T. Joyce., Weems, Chip (2012) Object-Oriented Data Structures Using Java, Third Editionjk
9. DSisa, A. (2002). Estructuras de datos y algoritmos: con énfasis en programación orientada a objetos. Bogotá: Prentice Hall. *
10. Goodrich, M.T., Tamassia, R. y Goldwasser, M. H. (2014). Data Structures & Algorithms in Java. Sixth edition. E.U.: Wiley.
11. Joyanes, A.L., Zahonero, M.I. (2014). Programación en C, C++, Java y UML. México: McGraw Hill.
12. Weiss, M. (2002). Programming and Problem Solving with C++: Comprehensive, Sixth Edition en java. Madrid: Addison Wesley. *

COMPLEMENTARIAS

1. Cetus Team (2010). Architecture and Design: Unified Modeling Language (UML). [en línea] URL http://www.cetuslinks. org/oo_uml.html. *
2. López, L. (2011). Programación Estructurada y Orientada A Objetos. México: Alfaomega.*
3. Martín, A. (2010). Programador Certificado Java 2. México: Alfaomega Ra-ma *
4. Sierra, K., Bates, B. (2009) Head First Java. 2nd ed. USA: O'Reilly Media.*
5. Bronson, Gary J., (2007) C++ para ingeniería y ciencias, 2a edición. Cengage Learning Editores, S.A. de C.V.*
6. Dale, N., Chip, Weems (2014) Programming and Problem Solving with C++: Comprehensive, Sixth Edition.
7. Fuenlabrada, V. S, Miranda E (2015). Manejo de técnicas de programación (eBook). Enfoque por competencias. Lenguajes y programación. México: Pearson.
8. López, Goytia J. L., Gutierrez, González A. (2014). Programación orientada a objetos con C++ y Java. Grupo Editorial Patria S.A. de C.V.

*La bibliografía con antigüedad mayor de cinco años contiene información relevante para el desarrollo de esta asignatura. Cabe destacar que son textos clásicos con ejemplos didácticos de fácil comprensión para el estudiante. Son difíciles de conseguir en el mercado, pero se encuentran en los catálogos de varias bibliotecas**.**

RESPONSABLE DEL DISEÑO	
Elaborado por:	Jorge Alberto Gómez de Dios, Julián Alejandro González Arellano.
Fecha de elaboración	20 de diciembre de 2016.