1. Datos Generales de la asignatura

Nombre de la asignatura:	Introducción a la Programación
Clave de la asignatura:	CBD-2424
SATCA[1]:	2-3-5
Carrera:	Ingeniería en Ciberseguridad.

2. Presentación

Caracterización de la asignatura

Aporta el perfil del Ingeniero en Ciberseguridad las siguientes habilidades:

- Utiliza sistemas operativos, lenguajes de programación, redes y entornos tecnológicos para integrar soluciones de seguridad con responsabilidad e inclusión social en las organizaciones.
- Dirige el monitoreo, análisis y control de la información utilizando herramientas y marcos de referencia, con perspectiva ética, de respeto por la persona y de responsabilidad social.
- Evalúa riesgos de seguridad y vulnerabilidad en aplicaciones o instalaciones de tecnologías de la información con apoyo de herramientas de vanguardia automatizadas de acuerdo con metodologías, normas y estándares de excelencia.

La asignatura de Introducción a la Programación pretende proporcionar a los estudiantes los conocimientos, habilidades y metodologías fundamentales para iniciarse en la programación. Se centra en capacitar a los estudiantes en conceptos básicos de programación, así como en la comprensión de las estructuras y herramientas esenciales utilizadas en el desarrollo de software. Esta asignatura es importante debido a que proporcionan a los estudiantes una base sólida en programación, permitiéndoles plantear y resolver problemas computacionales de manera eficiente y utilizando herramientas y técnicas adecuadas.

La asignatura se relaciona con otras materias donde se aplican metodologías de programación, proporcionando una base fundamental para su desarrollo académico y profesional en campos relacionados con la informática y la ingeniería de software.

Intención didáctica

La asignatura de Introducción a la Programación se concibe como un espacio donde los estudiantes son desafiados a resolver problemas computacionales mediante diversas técnicas y herramientas de modelado y codificación, bajo la orientación y monitoreo del docente.
El primer tema, se busca introducir al estudiante a los fundamentos de programación, asegurando una comprensión sólida de los conceptos fundamentales. A través de la exploración de diversos entornos de desarrollo, se fomenta la evaluación de las ventajas y desventajas de cada uno, preparando al estudiante para su futura práctica en el desarrollo de software.

[1] Sistema de Asignación y Transferencia de Créditos Académicos

En el segundo tema, se busca desarrollar la capacidad de análisis del estudiante para identificar problemas y plantear soluciones de manera efectiva a través de la aplicación de estructuras en el desarrollo de software. Se promueve la aplicación gradual de competencias adquiridas en la resolución de problemas concretos.

En el tercer tema, se profundiza en los conceptos de clases y objetos, permitiendo al estudiante modelar propuestas sobre diferentes contextos y destacando la jerarquía de clases y la definición de instancias de objetos. Se busca que el estudiante adquiera habilidades para aplicar estos conceptos en el desarrollo de soluciones prácticas.

En el cuarto tema se abordan las metodologías de desarrollo de software permitiendo que el estudiante pueda planificar, diseñar, implementar y mantener aplicaciones de software de manera eficiente y colaborativa, así como la aplicación de buenas prácticas de programación y la introducción a las pruebas unitarias para asegurar un correcto funcionamiento de las aplicaciones desarrolladas.

Es esencial que el docente trabaje en estrecha colaboración con los estudiantes para identificar problemas y promover la aplicación gradual de las competencias adquiridas en cada tema, con el objetivo de integrar un proyecto de asignatura que permita consolidar y aplicar los conocimientos adquiridos en situaciones reales.

Se fomentará el desarrollo de competencias genéricas para el análisis y resolución de problemas reales, así como las discusiones grupales y exposiciones que fortalezcan la competencia de expresión oral, promoviendo así un aprendizaje integral y significativo para los estudiantes.

3. Participantes en el diseño y seguimiento curricular del programa

Lugar y fecha de elaboración o revisión	Participantes	Observaciones
Tecnológico Nacional de México del 4 al 6 de marzo del 2024.	Representantes de los Institutos Tecnológicos de: Aguascalientes, Cerro Azul, Ciudad Juárez, La Paz, Jiquilpan, Mérida, Morelia, Tuxtla Gutiérrez, Villahermosa. Institutos Tecnológicos Superiores de La Región Carbonífera, Las Choapas	Propuesta sintética de la carrera de Ingeniería en Ciberseguridad.
Tecnológico Nacional de México del 22 al 26 de abril del 2024.	Representantes de los Institutos Tecnológicos de: Aguascalientes, Cerro Azul, Ciudad Juárez, La Paz, Jiquilpan, Mérida, Morelia, Tuxtla Gutiérrez, Villahermosa. Institutos Tecnológicos Superiores de La Región Carbonífera, Las Choapas. Representante de Ciencias Básica de los Institutos de: Celaya, Morelia CENIDET y CIIDET.	Diseño y/o desarrollo curricular de la carrera de Ingeniería en Ciberseguridad
Tecnológico Nacional de México del 27 al 31 de mayo del 2024.	Representantes de los Institutos Tecnológicos de: Aguascalientes, Cerro Azul, Jiquilpan, Mérida, Villahermosa. Institutos Tecnológicos Superiores de La Región Carbonífera, Las Choapas	Consolidación curricular de la carrera de Ingeniería en Ciberseguridad.

4. Competencia(s) a desarrollar

Competencia(s) específica(s) de la asignatura
• Diseña, desarrollar y mantiene aplicaciones de software de manera efectiva, aplicando los principios y técnicas fundamentales de la programación y el desarrollo de software.

5. Competencias previas

• Identifica las estructuras básicas de las matemáticas discretas para aplicarlas en el manejo, tratamiento y seguridad de la información.

6. Temario

No.	Temas	Subtemas
1	Introducción a la programación	1.1. Conceptos fundamentales de programación. 1.2. Historia y evolución de los lenguajes de programación. 1.3. Ciclo de vida del software. 1.4. Estructuras de datos y algoritmos básicos. 1.5. Herramientas y entornos de desarrollo.
2	Fundamentos de la programación estructurada	2.1. Algoritmos y su representación. 2.2. Tipos de datos y variables. 2.3. Operadores y expresiones. 2.4. Arreglos. 2.5. Estructuras de control: condicionales y bucles. 2.6. Funciones y procedimientos.
3	Programación Orientada a Objetos (POO)	3.1. Principios de la programación orientada a objetos. 3.2. Clases y objetos. 3.3. Encapsulamiento, herencia y polimorfismo. 3.4. Gestión de errores y excepciones.
4	Técnicas de comprobación y validación	4.1. Metodologías de desarrollo de software. 4.2. Diseño de algoritmos y solución de problemas. 4.3. Documentación y buenas prácticas de codificación. 4.4. Pruebas de escritorio. 4.5. Pruebas unitarias y debugging.

7. Actividades de aprendizaje de los temas

1. Introducción a la Programación	
Competencias	Actividades de aprendizaje
Específica(s): Comprende los conceptos básicos de la programación desarrollo de software y adquiere habilidades básicas en el uso de herramientas y entornos de desarrollo. *Genérica(s):* ● Habilidades de gestión de información ● Capacidad de análisis y síntesis. ● Conocimientos básicos de la carrera. ● Capacidad de comunicación oral y escrita. ● Habilidad para trabajar en forma autónoma. ● Habilidades en el uso de las tecnologías de la información y de la comunicación. *Transversal(es):* ● Aplica los conocimientos en la práctica, identificando aquellos que incorporen el compromiso con la responsabilidad social. ● Usa comunicación oral y escrita atendiendo los principios de no discriminación, Inclusión y equidad social. ● Diseña e implementa soluciones a problemas propios de ámbito de su área de aplicación integrando aprendizajes, rasgos y capacidades de excelencia, vanguardia e innovación social que fortalezcan el desarrollo humano.	● Investigar la historia y evolución de los lenguajes de programación, así como las diferentes estructuras de datos y algoritmos básicos y las fases de su ciclo de vida, identificando aquellos aspectos que incorporen el compromiso con la responsabilidad social, como el impacto de los lenguajes y tecnologías de programación en la sociedad y el desarrollo sostenible. ● Investigar conceptos fundamentales de la programación. ● Realizar ejercicios prácticos para aplicar los conceptos fundamentales de programación aprendidos con ejemplos sencillos. ● Explorar diferentes herramientas y entornos de desarrollo.

2. Fundamentos de la programación estructurada	
Competencias	Actividades de aprendizaje
Específica(s): Capacidad para comprender los conceptos fundamentales de la programación estructurada, así como de diseñar e implementar programas de software en un lenguaje de programación específico. *Genérica(s):* ● Habilidad para trabajar en forma autónoma. ● Capacidad de aplicar los conocimientos en la práctica. ● Capacidad para actuar en nuevas situaciones. ● Capacidad para identificar, plantear y resolver problemas. ● Habilidades en el uso de las tecnologías de la información y de la comunicación. *Transversal(es):* ● Aplica los conocimientos en la práctica, identificando aquellos que incorporen el compromiso con la responsabilidad social. ● Usa comunicación oral y escrita atendiendo los principios de no discriminación, Inclusión y equidad social. ● Diseña e implementa soluciones a problemas propios de ámbito de su área de aplicación integrando aprendizajes, rasgos y capacidades de excelencia, vanguardia e innovación social que fortalezcan el desarrollo humano.	● Realizar ejercicios prácticos de codificación que impliquen usar datos, variables, operadores y expresiones para familiarizarse con la sintaxis y la lógica de programación. ● Resolver una variedad de problemas que requieran el uso de estructuras de control condicionales y bucles para controlar el flujo de ejecución del programa. ● Realizar pruebas en los programas desarrollados para identificar y corregir posibles errores (bugs), utilizando técnicas de depuración para entender y solucionar problemas en el código. ● Participar en sesiones de revisión de código en grupos diversos e inclusivos, donde los estudiantes puedan compartir y analizar el código que han escrito, proporcionando retroalimentación constructiva para mejorar la calidad y eficiencia de este.

3. Programación Orientada a Objetos (POO)	
Competencias	Actividades de aprendizaje
Específica(s): Capacidad diseñar, implementar y depurar programas utilizando los principios de la programación orientada a objetos. *Genérica(s):* ● Capacidad creativa. ● Capacidad para identificar, plantear y resolver problemas. ● Capacidad de análisis y síntesis. ● Capacidad de aplicar los conocimientos en la práctica. ● Habilidades en el uso de las tecnologías de la información y de la comunicación. *Transversal(es):* ● Aplica los conocimientos en la práctica, identificando aquellos que incorporen el compromiso con la responsabilidad social. ● Usa comunicación oral y escrita atendiendo los principios de no discriminación, Inclusión y equidad social. ● Diseña e implementa soluciones a problemas propios de ámbito de su área de aplicación integrando aprendizajes, rasgos y capacidades de excelencia, vanguardia e innovación social que fortalezcan el desarrollo humano.	● Realizar ejercicios prácticos de codificación que impliquen la creación y el uso de clases y objetos para representar entidades del mundo real, así como para modelar y resolver problemas específicos, aplicando los conocimientos en la práctica y diseñando soluciones que integren aprendizajes, rasgos y capacidades de excelencia, vanguardia e innovación social que fortalezcan el desarrollo humano. ● Resolver problemas de programación que requieran el uso de herencia para establecer relaciones entre clases y el polimorfismo para permitir que objetos de diferentes clases respondan de manera uniforme a mensajes comunes. ● Trabajar en programas que involucren la aplicación de los principios de la programación orientada a objetos para diseñar e implementar sistemas de software más complejos y modulares, fomentando el trabajo en equipo y la colaboración en un ambiente inclusivo y diverso, donde se valoren las diferentes perspectivas y se promuevan oportunidades equitativas para todas las personas, independientemente de su género, origen o capacidades.

4. Técnicas de comprobación y validación	
Competencias	Actividades de aprendizaje
Específica(s): Capacidad de diseñar, desarrollar y mantener aplicaciones de software de manera efectiva, aplicando las metodologías y prácticas de desarrollo de software. *Genéricas:* ● Capacidad creativa. ● Capacidad para identificar, plantear y resolver problemas. ● Capacidad de análisis y síntesis. ● Capacidad de aplicar los conocimientos en la práctica. ● Habilidades en el uso de las tecnologías de la información y de la comunicación. ● Capacidad de trabajo en equipo. *Transversal(es):* ● Aplica los conocimientos en la práctica, identificando aquellos que incorporen el compromiso con la responsabilidad social. ● Usa comunicación oral y escrita atendiendo los principios de no discriminación, Inclusión y equidad social. ● Diseña e implementa soluciones a problemas propios de ámbito de su área de aplicación integrando aprendizajes, rasgos y capacidades de excelencia, vanguardia e innovación social que fortalezcan el desarrollo humano.	● Investigar las distintas metodologías de desarrollo de software actuales y realizar un cuadro comparativo, aplicando los conocimientos en la práctica e identificando aquellos aspectos que incorporen el impacto de las metodologías en la calidad, eficiencia y sostenibilidad del desarrollo de software. ● Participar en sesiones de planificación y diseño de proyectos de software, donde los estudiantes puedan definir los objetivos, requisitos y funcionalidades de la aplicación a desarrollar, así como establecer un plan de trabajo y asignar tareas, utilizando una comunicación oral y escrita que atienda los principios de no discriminación, inclusión y equidad social, y fomentando un ambiente de trabajo en equipo colaborativo y respetuoso de la diversidad. ● Documentar el código de las aplicaciones desarrolladas, incluyendo comentarios claros y descriptivos, así como explicaciones de la lógica y el propósito de las funciones y componentes del software, utilizando una comunicación oral y escrita que atienda los principios de no discriminación, inclusión y equidad social. ● Diseñar y ejecutar pruebas unitarias para verificar el correcto funcionamiento de las funciones y componentes del software, así como identificar y corregir errores (bugs) utilizando técnicas de debugging y herramientas de depuración, aplicando los conocimientos en la práctica de manera rigurosa y ética, y promoviendo la calidad y la transparencia en el proceso de desarrollo de software.

8. Práctica(s)

- Realizar ejercicios de resolución de problemas a través de seudocódigos
- Realizar diversos programas que ejemplifiquen el suso de variables, constantes, estructuras cíclicas, estructuras condicionales, arreglos y funciones.
- Realizar programas que resuelvan problemas de programación haciendo uso del paradigma de programación de orientada a objetos (clases, objetos, polimorfismo, herencia y encapsulamiento).
- Aplicar diversas metodologías de desarrollo de software en la resolución de problemas de programación sencillos, aplicando técnicas de validación y pruebas de calidad del software.

9. Proyecto de asignatura

Desarrollo de un Sistema de Gestión de Inventario

El presente proyecto tiene como objetivo principal demostrar el desarrollo y alcance de los logros formativos de la asignatura de Introducción a la Programación. Para ello, se propone la creación de un sistema de gestión de inventario que permita a una empresa u organización realizar un seguimiento de sus productos almacenados.

El objetivo del proyecto que planteé el docente que imparta esta asignatura, es demostrar el desarrollo y alcance del(los) logro(s) formativo(s) de la asignatura, considerando las siguientes fases:

- **Fundamentación:** marco referencial (teórico, conceptual, contextual, legal) en el cual se fundamenta el proyecto de acuerdo con un diagnóstico realizado, mismo que permite a los estudiantes lograr la comprensión de la realidad o situación objeto de estudio para definir un proceso de intervención o hacer el diseño de un modelo.
- **Planeación:** con base en el diagnóstico en esta fase se realiza el diseño del proyecto por parte de los estudiantes con asesoría del docente; implica planificar un proceso: de intervención empresarial, social o comunitario, el diseño de un modelo, entre otros, según el tipo de proyecto, las actividades a realizar los recursos requeridos y el cronograma de trabajo.
- **Ejecución:** consiste en el desarrollo de la planeación del proyecto realizada por parte de los estudiantes con asesoría del docente, es decir en la intervención (social, empresarial), o construcción del modelo propuesto según el tipo de proyecto, es la fase de mayor duración que implica el desempeño de los saberes, habilidades y destrezas a desarrollar.
- **Evaluación:** es la fase final que aplica un juicio de valor en el contexto laboral-profesión, social e investigativo, ésta se debe realizar a través del reconocimiento de logros y aspectos a mejorar se estará promoviendo el concepto de “evaluación para la mejora continua”, el desarrollo del pensamiento crítico y reflexivo en los estudiantes.

10. Evaluación de saberes, habilidades y destrezas

- Rúbrica.
- Listas de cotejo.
- listas de verificación.
- Manual de prácticas.

11. Fuentes de Información

1. Sipser, M. (2012). Introduction to the Theory of Computation. Cengage Learning. [DOI: 10.1007/978-3-319-06653-7]
2. Martin, R. C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall. ISBN: 978-0132350884
3. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional. ISBN: 978-0201633610
4. Martin, R. C. (2006). Agile Principles, Patterns, and Practices in C#. Prentice Hall. ISBN: 978-0131857254
5. Asociación Nacional de Instituciones de Educación en Tecnologías de Información A.C. (2024). Modelo curricular por competencias. ANIEI.