Programa de Estudios de la Unidad de Aprendizaje:	PROGRAMACIÓN INTERMEDIA

Clave:	4FP-FM1065	Créditos:	4.5

Ramas de Conocimiento					
Ingeniería y Ciencias Físico Matemáticas	X	Ciencias Sociales Administrativas		Ciencias Médico Biológicas	

Área de Formación Curricular					
Institucional		Científica, Humanística y Tecnológica Básica		Profesional	X

Tipo de Espacio							
Aula	X	Taller		Laboratorio	X	Otros ambientes de Aprendizaje	

Modalidad					
Escolarizada	X	No Escolarizada		Mixta	

Vigencia a Partir:	ENERO 2024

Proceso de Diseño y Autorización:			Día	Mes	Año
Elaborado por:	REP. ACAD. NMS	Fecha de Elaboración:	20	02	2023
Revisado por:	DEMS	Fecha de Revisión:	10	10	2023
Aprobado por:	CTCE-NMS	Fecha de Aprobación:	24	10	2023
Autorizado por:	CPA-CGC	Fecha de Autorización:	23	11	2023

Programa Académico:	TÉCNICO EN PROGRAMACIÓN					
Nivel:	1°	2°	3°	4°	5°	6°

Unidades Académicas donde se Imparte:																				
TODAS LAS U.A.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	CET1

Tiempos Asignados:

Global:	72	Hrs/18 semanas/Semestre		
Aula:	1	Hrs/Semana	Total: 18	Hrs/Semestre
Taller:	0	Hrs/Semana	Total: 0	Hrs/Semestre
Laboratorio:	3	Hrs/Semana	Total: 54	Hrs/Semestre
Otros ambientes de aprendizaje:	0	Hrs/Semana	Total: 0	Hrs/Semestre

Organización					
Por Unidad de Aprendizaje:	X	Por Área:		Por Módulo:	

Firma y Sello de Autorización:

M. EN E.N.A. MARÍA ISABEL ROJAS RUÍZ
Directora de Educación Media Superior

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

FUNDAMENTACIÓN

La unidad de aprendizaje **Programación Intermedia** pertenece al área de formación profesional del Bachillerato Tecnológico Bivalente del Nivel Medio Superior del Instituto Politécnico Nacional, se ubica en el tercer nivel del Programa Académico "Técnico en Programación" y se imparte en la modalidad escolarizada, de manera obligatoria en la rama del conocimiento de Ingeniería y Ciencias Físico-Matemáticas.

Esta unidad de aprendizaje coadyuva a comprender **la Programación Intermedia** como una dimensión científica, técnica, tecnológica, social, responsable y metodológica, que incentiva la adquisición, desarrollo y aplicación del pensamiento lógico, analítico, el razonamiento abstracto, la creatividad, la iniciativa, la resolución a problemas y diversas habilidades cognitivas. Introduce al estudiante al campo conceptual, procedimental y actitudinal para crear programas con un lenguaje de programación, resolver problemas de programación intermedia, diseñar, desarrollar y comprender los complejos problemas computacionales para solucionarlos de una manera efectiva. La adquisición de estas destrezas y habilidades relacionadas con el pensamiento analítico favorecerán en el estudiante el desarrollo de una visión crítica y holística, cuya puesta en práctica, en forma autónoma, en el futuro le coadyuvará a responder en forma eficiente y eficaz a los retos que se le presenten al incorporarse a estudios superiores o de igual forma al campo laboral.

La unidad de aprendizaje **Programación Intermedia** contribuye al desarrollo del talento requerido por la Industria 4.0 y para la transformación del país, orientada al logro del desarrollo de software en México. Esto debido a que adquirirá, desarrollará y aplicará conceptos, técnicas y métodos que favorecen el análisis, desarrollo y resolución de problemas computacionales a un nivel intermedio, mediante la escritura de programas en un lenguaje de programación, como instrumento estructural para resolver, organizar y analizar datos en forma lógica, representarlos mediante abstracciones, desde los diagramas de flujo, diseñar y escribir programas para dar solución a problemas cotidianos mediante el pensamiento analítico, crítico y algorítmico, optimizar e innovar soluciones extrapolando en una solución informática en un lenguaje de programación.

Programación Intermedia es una unidad de aprendizaje enfocada al desarrollo de habilidades técnicas, cognitivas y socioemocionales inherentes al estudio, análisis, aplicación y resolución de problemas, a través del desarrollo de aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos de forma responsable, creativa, eficaz, eficiente.

La unidad de aprendizaje Programación Intermedia estará fundamentada en el Modelo Educativo Institucional vigente y en la Educación para la industria 4.0, por esto, se empleará la metodologías didáctica activas como el Aprendizaje Basado en Problemas, el Aprendizaje Basado en Retos, entre otras; esto con el propósito de que el estudiante desarrolle competencias de un entorno 4.0, como: el trabajo colaborativo, trabajo en equipo, reto al cambio, autodirección, resolución de problemas cercanos a la realidad, autogestión del aprendizaje y resiliencia. Además, se emplearán herramientas tecnológicas que fomentarán la colaboración e interacción presenciales y virtuales, en forma síncrona o asíncrona, que corresponden a la Educación 4.0. También se emplearán aplicaciones informáticas para el análisis, diseño, codificación y pruebas de problemas computacionales comunes, entre otros.

El rol del docente será de mediador entre el estudiante y los contenidos conceptuales y didácticos a abordar, puesto que se centrará en la creación, organización, supervisión y mediación de los espacios de trabajo, incluidas las aulas virtuales, el aula invertida, los ciberespacios, sin dejar de atender las necesidades técnicas, de conocimientos, apoyo logístico y metodológico en los procesos de aprendizaje individual y grupal, con el objetivo de generar ambientes que favorezcan la educación técnica, inclusiva y flexible.

El estudiante desarrollará un trabajo autónomo en diferentes ambientes de aprendizaje, organizará el trabajo de manera independiente y articulará saberes de diversos campos del conocimiento, que le posibilitarán construir y expresar su propio conocimiento en beneficio de la sociedad; también adquirirá habilidades tanto tecnológicas como personales que promoverán la comunicación asertiva, la creatividad, la negociación, la gestión del tiempo, la motivación, el liderazgo y la responsabilidad social, la erradicación de toda manifestación de violencia de género, la inclusión y la accesibilidad.

La evaluación se efectuará en el marco de la evaluación auténtica, por esto, comprenderá tres momentos: diagnóstica, formativa y sumativa. La evaluación diagnóstica se llevará a cabo mediante una charla informal sobre conocimientos previos con evaluación y retroalimentación durante el mismo momento, con la finalidad de que el docente efectúe los ajustes didácticos pertinentes y de ser necesario, nivele y ajuste los conocimientos previos adquiridos en otras unidades de aprendizaje para que establezca conexiones significativas con las unidades didácticas de la unidad de aprendizaje. Un segundo momento de la evaluación hace referencia a la evaluación formativa, que se desarrollará a

lo largo del proceso de enseñanza-aprendizaje mediante las secuencias didácticas y actividades de aprendizaje formativos que estimulen el aprendizaje activo y significativo del estudiante. Este momento se enriquecerá con diversos tipos de evaluación, como la autoevaluación, la coevaluación y heteroevaluación, puesto que coadyuvarán a dar seguimiento al desarrollo de los saberes y habilidades en contexto. Cabe señalar que estas clases de evaluación serán reforzadas a través de la retroalimentación efectiva y constante.

En el tercer momento de la evaluación, con fines de acreditación, se diseñarán casos con problemas comunes que permitan recuperar el nivel de logro y conducir al estudiante a la meta cognición en la unidad de aprendizaje Programación Intermedia, mediante evidencias de conocimiento, comprensión y diseño de algoritmos, escritura de programas en algún lenguaje de programación, compilación de los programas escritos, entre otras evidencias de aprendizaje, cuyos criterios, aspectos e indicadores serán conocidos por los estudiantes en forma previa. Las evidencias de evaluación formativa e integradora mostrarán el saber hacer de manera reflexiva de los estudiantes, utilizando el conocimiento que van adquiriendo durante el proceso didáctico para luego transferir y aplicar este aprendizaje en contextos escolares, sociales y laborales.

En base a la flexibilidad curricular y en el reconocimiento de aprendizajes múltiples, también podrá aplicarse una evaluación general para verificar que los conocimientos adquiridos por el estudiante demuestren que domina los saberes, objetivos y alcances de la Programación Intermedia, al finalizar el periodo ordinario. De esa forma, el programa de estudios de esta unidad de aprendizaje establece estándares para el desarrollo de conocimientos, habilidades socioemocionales, actitudes y valores.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

DESCRIPCIÓN DE LA UNIDAD DE APRENDIZAJE

Unidad de Aprendizaje: Programación Intermedia		
Propósito de la Unidad de Aprendizaje Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		
Unidad 1: Arquitectura del software y propiedades de Programación Orientada a Objetos		
Unidad de competencia	Aprendizajes esperados	Contenidos de aprendizaje
1. Usa los elementos del software para programar con base en su estructura, comportamiento e interacción, tomando en cuenta las propiedades de herencia, encapsulamiento, polimorfismo y abstracción, respetando la simbología y buenas prácticas de programación.	1. Clasifica escenarios para aplicar los conceptos de clase, método y objeto utilizando las propiedades de abstracción, encapsulamiento, herencia y polimorfismo de forma ordenada y funcional.	**Conceptual:** 1.1 Conceptos básicos de programación orientada a objetos 1.1.1 Clase 1.1.2 Objeto 1.1.3 Abstracción 1.1.4 Encapsulamiento 1.1.5 Herencia 1.1.6 Polimorfismo **Procedimental:** Reconoce las herramientas de programación orientada a objetos disponibles en los sistemas de cómputo. **Práctica 1**. Hola mundo (Se realiza después del subtema 1.1.2 Objeto) **Actitudinal:** Muestra una actitud respetuosa hacia sus compañeros y docentes, el desarrollo de sus actividades lo realiza de una forma analítica, reflexiva, con una comunicación efectiva y el trabajo se cumple de forma colaborativa.
	2. Usa los principios del modelado de clases en UML en la solución de problemas en un entorno real, utilizando diagramas de clases, diagramas de actividades, entre otros. De forma ordenada y funcional.	**Conceptual:** 1.2 UML 1.2.1 Definición 1.2.2 Elementos 1.2.2.1 Diagrama de casos de uso 1.2.2.2 Diagrama de clases 1.2.2.3 Diagrama de secuencias 1.2.2.4 Diagrama de colaboración 1.2.2.5 Diagrama de estado 1.2.2.6 Diagrama de actividades 1.2.2.7 Diagrama de componentes **Procedimental:** Emplea los principios de UML (Unified Modeling Language) de forma ordenada y funcional en la solución de problemas de un entorno real, por medio de diagramas de clase, diagramas de actividades, entre otros para la elaboración de los programas informáticos.

		Práctica 2. Modelado de software. (se realiza después del subtema 1.2.2.7 Diagrama de componentes) **Actitudinal:** Muestra una actitud respetuosa hacia sus compañeros y docentes, el desarrollo de sus actividades lo realiza de una forma analítica, reflexiva, con una comunicación efectiva y el trabajo se cumple de forma colaborativa.
Unidad 2: Persistencia de datos y manejo de archivos		
Unidad de competencia	Aprendizajes esperados	Contenidos de aprendizaje
2. Demuestra software para resguardar información usando elementos de persistencia de datos y manejo de archivos de forma ordenada, analítica y responsable.	1. Practica programas para almacenar, consultar y quitar información de forma temporal utilizando clases, objetos, métodos, arreglos, listas y vectores de forma persistente y ordenada.	**Conceptual:** 2.1 Listas genéricas de matrices 2.1.2 Tipos de listas, Composición y características de las listas genéricas 2.1.2 Declaración de características de clases con métodos get() y set(). 2.1.3 Acceso e iteración 2.1.5 Algoritmos para almacenar, consultar y quitar elementos de las listas. **Procedimental:** Demuestra programas de listas genéricas con algoritmos para almacenar, consultar y quitar información de las listas genéricas. **Práctica 3**. Listas genéricas. (Se realiza después del subtema 2.1.5 Algoritmos para almacenar, consultar y quitar elementos de las listas) **Actitudinal:** Hace uso del pensamiento criticó y reflexivo Capacidad para analizar y resolver casos
	2. Pone a prueba software para guardar, buscar y eliminar información de manera definitiva implementando archivos de forma analítica y responsable.	**Conceptual** 2.2. Archivos 2.2.1 Características de un archivo 2.2.2 Lectura y escritura de bytes 2.2.3 Secuencias y flujo de datos 2.2.4 Manejo de ficheros **Procedimental** Realiza software capaz de resguardar, eliminar y editar información en archivos. **Práctica 4.** Manejo de archivos. (Se realiza al final del subtema 2.2.4 Manejo de ficheros) **Actitudinal** Usa el pensamiento crítico para identificar y reconocer errores. Demuestra habilidad de resolución de casos.
Unidad 3: Interfaces gráficas		
Unidad de competencia	Aprendizajes esperados	Contenidos de aprendizaje
3. Crea software con interfaces gráficas para mejorar la experiencia de la interacción con el	1. Ordena una interfaz gráfica para permitir al usuario ingresar y visualizar información empleando componentes de entrada, salida y controles, fácil de operar y usar.	**Conceptuales** 3.1 Interfaz gráfica de usuario 3.1.1 Arquitectura Modelo-Vista-Controlador 3.1.2 Contenedores

usuario, mediante componentes de entrada, salida, controles y eventos considerando las cualidades de usabilidad.		3.1.3 Componentes de entrada 3.1.4 Controles y sus propiedades 3.1.5 Jerarquía de componentes en una aplicación 3.1.6 Gráficos, iconos e imágenes 3.1.7 Mensajes de dialogo **Procedimental** Ejemplifica componentes y controles para presentar interfaces gráficas de usuario. Implementa formularios para clasificar datos de entrada. **Práctica 5.** Interfaz gráfica. (Se realiza después del subtema 3.1.7 Mensajes de dialogo) **Actitudinal** Aplica la creatividad para dar solución a problemas. Actúa de manera ética y responsable
	2. Demuestra software con funcionalidades para manipular información implementando eventos para crear interactividad entre la interfaz gráfica y el usuario, intuitivo y fácil de usar.	**Conceptual** 3.2 Programación orientada a eventos 3.2.1 Eventos con el mouse 3.2.2 Eventos con el teclado 3.2.3 Menús interactivos 3.2.4 Cajas de dialogo 3.2.5 Métodos para guardar, registrar, editar y mostrar información. 3.2.1 Tablas y Arboles **Procedimental** Implementa una interfaz gráfica para introducir, buscar y mostrar información. Integra elementos interactivos en la interfaz gráfica. **Práctica 6.** Eventos (Se realiza después del subtema 3.2.4 Cajas de dialogo) **Actitudinal** Aplica la creatividad para dar solución a problemas. Actúa de manera ética y responsable.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

MATRIZ DE VINCULACIÓN

	Unidad de Competencia 1		Unidad de Competencia 2		Unidad de Competencia 3	
COMPETENCIAS PARA EL SIGLO XXI HABILIDADES BLANDAS Y SOCIOEMOCIONALES	**AE 1**	**AE 2**	**AE 1**	**AE 2**	**AE 1**	**AE 2**
Trabajo colaborativo	X	X			X	X
Pensamiento reflexivo	X	X			X	X
Resolución de problemas.	X	X	X	X	X	X
Pensamiento crítico y analítico	X	X	X	X	X	X
Comunicación efectiva					X	X
Creatividad	X	X	X	X	X	X
Responsabilidad			X	X	X	X
Ética			X	X	X	X
Respeto			X	X	X	X

INSTITUTO POLITÉCNICO NACIONAL
Dirección de Educación Media Superior

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

PERFIL DOCENTE

El docente que imparta la Unidad de Aprendizaje **Programación Intermedia** del programa Académico de Técnico en Programación, contará con las habilidades en el manejo de los saberes disciplinares y/o profesionales, así como su disposición, autoridad y tolerancia en el manejo de grupos de aprendizaje. Por lo tanto, debe poseer las habilidades que favorezcan el desarrollo del talento 4.0.

Habilidades docentes en el desarrollo del Talento

En el campo de su especialización:

- Habilidades y conocimientos profesionales que se requiere para la impartición de la Unidad de Aprendizaje Programación Intermedia.
- Actualiza las habilidades digitales para desarrollarlas e implementarlas en el aula.
- Experiencia deseable en el campo laboral.

En el campo pedagógico:

- Fomentar procesos de enseñanza que le permitan interpretar y resolver las necesidades de aprendizaje de los estudiantes, tomando en cuenta sus capacidades, habilidades, vocación e intereses.
- Desarrollar procesos de enseñanza aprendizaje, utilizando métodos basados en administración de proyectos reales, aprovechando espacios educativos distintos a las aulas, para mejorar la calidad y pertinencia de la enseñanza.

En el campo de la investigación:

- Fortalecer el trabajo académico a partir del aprovechamiento de los resultados y productos de los proyectos de investigación.

Perfil Profesional

- Titulado en Ing. en Sistemas Computacionales, Inteligencia Artificial, Ciencia de Datos, y/o Licenciado en Informática, Comunicaciones y Electrónica, Telemática, Computación o Maestría en Ciencias de la Computación, maestría o especialización en docencia o afín con experiencia de dos años en el área docente.
- Experiencia comprobable de dos años en la iniciativa pública o privada aplicando los conocimientos de la unidad de aprendizaje.

La unidad de aprendizaje Programación Intermedia provee al estudiante de competencias profesionales que deben ser supervisadas y evaluadas de forma individual, debido a la complejidad de la unidad de aprendizaje y al tamaño de los grupos, el docente titular requiere del apoyo de dos docentes auxiliares con las horas frente a grupo y las mismas competencias, capaces de diseñar estrategias y recursos didácticos así como evaluar y realimentar a los estudiantes al interior del aula y laboratorio, de esta forma cada docente podrá atender a varios estudiantes del mismo grupo permitiéndole resolver las dudas, orientar su aprendizaje y brindarle la realimentación pertinente y de calidad, esto evitará retrasos y la falta de atención, factores que impactan en la trayectoria académica de los estudiantes.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

ESTRUCTURA DIDÁCTICA

Unidad Didáctica 1:	Arquitectura del software y propiedades de Programación Orientada a Objetos	**Nivel:**	4to
Propósito General:	Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		
Unidad de Competencia No 1:	Usa los elementos del software para programar con base en su estructura, comportamiento e interacción, tomando en cuenta las propiedades de herencia, encapsulamiento, polimorfismo y abstracción, respetando la simbología y buenas prácticas de programación.		
Aprendizaje Esperado No 1:	Clasifica escenarios para aplicar los conceptos de clase, método y objeto utilizando las propiedades de abstracción, encapsulamiento, herencia y polimorfismo de forma ordenada y funcional.	**Tiempo estimado para obtener el Aprendizaje Esperado:**	12

Contenidos de Aprendizaje

Conceptuales	Procedimentales	Actitudinales
1.1 Conceptos básicos de programación orientada a objetos 1.1.1 Clase 1.1.2 Objeto 1.1.3 Abstracción 1.1.4 Encapsulamiento 1.1.5 Herencia 1.1.6 Polimorfismo	Reconoce las herramientas de programación orientada a objetos disponibles en los sistemas de cómputo. ✓ **Práctica 1**. "Hola mundo"	Muestra una actitud respetuosa hacia sus compañeros y docentes, el desarrollo de sus actividades lo realiza de una forma analítica, reflexiva, con una comunicación efectiva y el trabajo se cumple de forma colaborativa.

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje basado en problemas.

Apertura: Los estudiantes conocen los principios de la Programación Orientada a Objetos, sus conceptos y sus características, así como la estructura de una clase y de un objeto, esto por medio del material que el docente les exponga.

Desarrollo: El docente expondrá a los estudiantes los elementos principales de la Programación Orientada a Objetos, formulará una serie de preguntas para reafirmar los conocimientos adquiridos por el estudiante.

Con el objetivo de verificar el correcto funcionamiento y configuración de los equipos de computo en el laboratorio, el docente solicita la elaboración de la Práctica 1 "Hola mundo", de manera individual, al igual que la evidencia de aprendizaje formativa, la cual será evaluada en ese momento.

Posteriormente, el docente planteará problemas a resolver con los conceptos de clase, objeto, abstracción, encapsulamiento, herencia y polimorfismo.

Cierre: Al concluir el tiempo destinado a la actividad planteada, el docente solicitará un informe con el proceso de la solución.

Ambiente de Aprendizaje: Aula y Laboratorio de Programación.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Instrumento y Criterios de Evaluación
Herramientas Tecnológicas: Equipo de cómputo Plataforma digital Software orientado a la unidad de competencia Internet Proyector **Recursos Didácticos:** Video tutoriales Repositorio digital con material de consulta Infografías Libros	Informe con el procedimiento de la solución de los problemas.	**Instrumento de Evaluación:** • Lista de cotejo. **Criterios de evaluación de forma:** • El informe cuenta con una portada que incluye los datos de identificación del estudiante. • Los textos están creados con fuente Arial 12, interlineado intermedio. **Criterios de evaluación de fondo:** • En la solución de problemas utiliza atributos y métodos. • Genera objetos a partir de una clase. • Hace uso de los principios de abstracción, encapsulamiento, herencia y polimorfismo. • No se presentan errores lógicos ni de sintaxis.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Unidad Didáctica 1:	Arquitectura del software y propiedades de Programación Orientada a Objetos	**Nivel:**	4to
Propósito General:	Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		
Unidad de Competencia No 1:	Usa los elementos del software para programar con base en su estructura, comportamiento e interacción, tomando en cuenta las propiedades de herencia, encapsulamiento, polimorfismo y abstracción, respetando la simbología y buenas prácticas de programación.		
Aprendizaje Esperado No 2:	Usa los principios del modelado de clases en UML en la solución de problemas en un entorno real, utilizando diagramas de clases, diagramas de actividades, entre otros. De forma ordenada y funcional.	**Tiempo estimado para obtener el Aprendizaje Esperado:**	12 horas

Contenidos de Aprendizaje

Conceptuales	Procedimentales	Actitudinales
1.2 UML 1.2.1 Definición 1.2.2 Elementos 1.2.2.1 Diagrama de casos de uso 1.2.2.2 Diagrama de clases 1.2.2.3 Diagrama de secuencias 1.2.2.4 Diagrama de colaboración 1.2.2.5 Diagrama de estado 1.2.2.6 Diagrama de actividades 1.2.2.7 Diagrama de componentes	Emplea los principios de UML (Unified Modeling Language) de forma ordenada y funcional en la solución de problemas de un entorno real, por medio de diagramas de clase, diagramas de actividades, entre otros para la elaboración de los programas informáticos. ✓ **Práctica 2**. Modelado de software	Muestra una actitud respetuosa hacia sus compañeros y docentes, el desarrollo de sus actividades lo realiza de una forma analítica, reflexiva, con una comunicación efectiva y el trabajo se cumple de forma colaborativa.

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aula invertida / Método del caso

Apertura: Los estudiantes, con la guía del docente conocen los conceptos de UML y la estructura de los diagramas de este lenguaje de modelado, por medio de una investigación que se le solicitará al estudiante realizar previo a la clase, para que posteriormente el docente retroalimente los conocimientos adquiridos, por medio de presentaciones y esquemas.
Desarrollo: El docente presentará a los estudiantes los conceptos clave de la UML, lanzará una serie de preguntas para consolidar el entendimiento del estudiante.

Se propondrá la Práctica 2. Modelado de software después de abordar el subtema 1.2.2.7 Diagrama de componentes.

El docente Planteará situaciones (casos) que requieran la aplicación de diagramas de casos de uso, clases, secuencia, colaboración, estado, actividades y componentes para su resolución

Cierre: Al concluir el tiempo destinado a la actividad planteada, el docente solicitará un informe con el proceso de la solución.

Ambiente de Aprendizaje: Aula y Laboratorio de Programación

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Instrumento y Criterios de Evaluación
Herramientas Tecnológicas: Equipo de cómputo Plataforma digital Software orientado a la unidad de competencia Internet Proyector **Recursos Didácticos:** Video tutoriales Repositorio digital con material de consulta Infografías	Informe con la solución a los casos integrando los diagramas UML	**Instrumentos de Evaluación:** • Lista de cotejo **Criterios de Evaluación:** Criterios de evaluación de forma: • El informe cuenta con una portada que incluye los datos de identificación del estudiante. • Los textos están creados con fuente Arial 12, interlineado intermedio. Criterios de evaluación de fondo: • El informe está conformado por una sección de análisis, representación gráfica de los diagramas UML y una conclusión. • La redacción del informe mantiene un tono académico y profesional, crítico y responsable. • Los requerimientos funcionales son identificados. • Los diagramas UML se hacen con la simbología establecida. • EL diagrama de casos de uso representa los actores y funcionalidades de la aplicación a tratar. • El diagrama de secuencia representa la comunicación entre elementos. • El diagrama de colaboración muestra la colaboración entre componentes. • El diagrama de estado muestra el estado de algún componente. • El diagrama de actividades representa el flujo de actividades entre el usuario y la lógica. • El diagrama de componentes representa la integración de componentes

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Unidad Didáctica 2:	Persistencia de datos y manejo de archivos	**Nivel:**	4to
Propósito General:	Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		
Unidad de Competencia No 2:	Demuestra software para resguardar información usando elementos de persistencia de datos y manejo de archivos de forma ordenada, analítica y responsable.		
Aprendizaje Esperado No 1:	Practica programas para almacenar, consultar y quitar información de forma temporal utilizando clases, objetos, métodos, arreglos, listas y vectores de forma persistente y ordenada.	**Tiempo estimado para obtener el Aprendizaje Esperado:**	12 horas

Contenidos de Aprendizaje

Conceptuales	Procedimentales	Actitudinales
2.1 Listas genéricas de matrices 2.1.2 Tipos de listas, Composición y características de las listas genéricas 2.1.2 Declaración de características de clases con métodos get() y set(). 2.1.3 Acceso e iteración 2.1.5 Algoritmos para almacenar, consultar y quitar elementos de las listas.	Demuestra programas de listas genéricas con algoritmos para almacenar, consultar y quitar información de las listas genéricas. ✓ **Práctica 3**. Listas genéricas.	Hace uso del pensamiento criticó y reflexivo Capacidad para analizar y resolver casos

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Método del caso

Apertura: el docente titular mediante presentaciones y material de apoyo explica a los estudiantes los tipos de listas genéricas de matrices, su composición y sus características, así como la estructura de una clase con métodos get() y set().

Desarrollo: el docente realizará una serie de demostraciones en las que expondrá a los estudiantes cómo hacer iteraciones a una lista así como los algoritmos para almacenar, consultar y quitar elementos de las listas, además propondrá una serie de ejercicios para saber aplicar las funciones de almacenar, consultar y quitar elementos de las listas, los estudiantes deberán programar y ejecutar los algoritmos realizados utilizando una computadora personal, con un software especializado para compilar y ejecutar programas de acuerdo con el lenguaje de programación que el docente esté utilizando.

Se llevará a cabo la Práctica 3. Listas genéricas. (después del subtema 2.1.5 Algoritmos para almacenar, consultar y quitar elementos de las listas)

Después, con el objetivo de reafirmar los conocimientos adquiridos por el estudiante, el docente planteará un caso a resolver en donde se apliquen las declaraciones de clases con los métodos get() y set() , iteraciones a listas y aplicaciones de los algoritmos para almacenar, consultar y quitar elementos de las listas. Es requerido que los docentes brinden un acompañamiento presencial en el laboratorio para solucionar dudas, realizar evaluaciones y realimentaciones a los estudiantes .

Cierre: al terminar la solución del caso, el docente solicitará un informe con el proceso de la solución.

Ambiente de Aprendizaje: Aula y Laboratorio de programación.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Instrumento y Criterios de Evaluación
Herramientas Tecnológicas: Equipo de cómputo Plataforma digital Software orientado a la unidad de competencia Internet Proyector **Recursos Didácticos:** Video tutoriales Repositorio digital con material de consulta Infografías	Informe con el proceso de solución del caso.	**Instrumento de Evaluación:** • Lista de cotejo **Criterios de Evaluación:** Criterios de evaluación de forma: • El informe cuenta con una portada que incluye los datos de identificación del estudiante. • El informe tiene textos creados con fuente Arial 12, interlineado intermedio. • El informe está conformado por un análisis, representación gráfica, codificación, pruebas del software y conclusión • La redacción del informe mantiene un tono académico y profesional, critico y responsable. • Usa las listas genéricas con algoritmos de almacenar, consultar y quitar información para la solución del caso.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Unidad Didáctica 2:	Persistencia de datos y manejo de archivos	**Nivel:**	Cuarto
Propósito General:	Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		
Unidad de Competencia No 2:	Demuestra software para resguardar información usando elementos de persistencia de datos y manejo de archivos de forma ordenada, analítica y responsable		
Aprendizaje Esperado No 2:	Pone a prueba software para guardar, buscar y eliminar información de manera definitiva implementando archivos de forma analítica y responsable.	**Tiempo estimado para obtener el Aprendizaje Esperado:**	12 horas

Contenidos de Aprendizaje

Conceptuales	Procedimentales	Actitudinales
2.2. Archivos 2.2.1 Características de un archivo 2.2.2 Lectura y escritura de bytes 2.2.3 Secuencias y flujo de datos 2.2.4 Manejo de ficheros	Realiza software capaz de resguardar, eliminar y editar información en archivos. **Práctica 4.** Manejo de archivos.	Hace uso del pensamiento criticó y reflexivo Capacidad para analizar y resolver casos

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Método del caso

Apertura: el docente mediante un presentación explica a los estudiantes los elementos requeridos para poder implementar archivos en programas de software así como el proceso de lectura y escritura de bytes, secuencias y flujos de datos.

Desarrollo: el docente en el laboratorio mostrará ejemplos utilizando entornos para el desarrollo de aplicaciones de software, donde explicará a los estudiantes la implementación de librerías para el manejo de archivos y ficheros de acuerdo con el lenguaje de programación que este utilizando.

Se realizará la Práctica 4. Manejo de archivos. (después del subtema 2.2.4 Manejo de ficheros)
Una vez concluida la práctica, los estudiantes realizarán ejercicios con diferentes niveles de complejidad para reafirmar los conocimientos adquiridos, además el docente diseñara y planteará un caso a resolver, éste deberá resolver empleando algoritmos capaces de resguardar, eliminar y editar información en archivos. Es requerido que los docentes brinden un acompañamiento presencial en el laboratorio para solucionar dudas, realizar evaluaciones y realimentaciones a los estudiantes.

Cierre: al terminar la solución del caso, el docente solicitará un informe con el proceso de la solución.

Ambiente de Aprendizaje: Aula y Laboratorio de programación

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Instrumento y Criterios de Evaluación
Herramientas Tecnológicas: Equipo de cómputo Plataforma digital IDE de desarrollo Software orientado a la unidad de competencia Internet Proyector **Recursos Didácticos:** Video tutoriales Repositorio digital con material de consulta Esquemas Gráficos Infografías Presentaciones	Informe con el proceso de solución del caso.	**Instrumento de Evaluación:** • Lista de cotejo **Criterios de Evaluación:** Criterios de evaluación de forma: • El informe cuenta con una portada que incluye los datos de identificación del estudiante. • Los textos están creados con fuente Arial 12, interlineado intermedio. • La redacción del informe mantiene un tono académico y profesional, critico y responsable Criterios de evaluación de fondo: • El informe está conformado por un análisis, representación gráfica, codificación, pruebas del software y conclusión. • Implementa algoritmos capaces de resguardar, eliminar y editar información en archivos. • La información en los archivos se mantiene integra al cerrar y cargar el programa. • La redacción del informe mantiene un tono académico y profesional, critico y responsable. • La aplicación del software emplea algoritmos de almacenar, consultar y quitar información en archivos.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Unidad Didáctica 3:	Interfaces gráficas	**Nivel:**	4to
Propósito General:	Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		
Unidad de Competencia No 3:	Crea software con interfaces gráficas para mejorar la experiencia de la interacción con el usuario, mediante componentes de entrada, salida, controles y eventos considerando las cualidades de usabilidad		
Aprendizaje Esperado No 1:	Ordena una interfaz gráfica para permitir al usuario ingresar y visualizar información empleando componentes de entrada, salida y controles, fácil de operar y usar.	**Tiempo estimado para obtener el Aprendizaje Esperado:**	12 horas

Contenidos de Aprendizaje

Conceptuales	Procedimentales	Actitudinales
3.1 Interfaz gráfica de usuario 3.1.1 Arquitectura Modelo-Vista-Controlador 3.1.2 Contenedores 3.1.3 Componentes de entrada 3.1.4 Controles y sus propiedades 3.1.5 Jerarquía de componentes en una aplicación 3.1.6 Gráficos, iconos e imágenes 3.1.7 Mensajes de dialogo	Ejemplifica componentes y controles para presentar interfaces gráficas de usuario. Implementa formularios para clasificar datos de entrada. ✓ **Práctica 5. "**Interfaz gráfica".	Aplica la creatividad para dar solución a casos. Actúa de manera ética y responsable.

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje Basado en Retos

Apertura: el docente, a través de un esquema gráfico muestra las capas y elementos que integran la Arquitectura Modelo-Vista-Controlador, muestra en un esquema de árbol la jerarquía de los elementos de la interfaz gráfica tales como contenedores, componentes de entrada, controles y sus propiedades, gráficos, iconos e imágenes y mensajes de dialogo.

Desarrollo: el docente, en el laboratorio usando el entorno de desarrollo para aplicaciones de software y librerías para el manejo de componentes de interfaces gráficas realiza y explica demostraciones de la integración de los elementos gráficos, componentes y controles a la interfaz gráfica de usuario para la creación de formularios.

Se llevará a cabo la Práctica 5**.** Interfaz gráfica. (después del subtema 3.1.7 Mensajes de dialogo)

El docente planteará una serie de retos de menor a mayor nivel, se le solicitará a los estudiantes resolverlos realizando aplicaciones de software capaces de mostrar una interfaz gráfica que cumpla con la solución para cada reto, los estudiantes integrarán las soluciones en un informe. Es requerido que los docentes brinden un acompañamiento presencial en el laboratorio para solucionar dudas, realizar evaluaciones y realimentaciones a los estudiantes .

Cierre: En equipo los estudiantes resolverán un reto planteado por el docente y expondrán su solución.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Ambiente de Aprendizaje: Aula y laboratorio

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Instrumento y Criterios de Evaluación
Herramientas Tecnológicas: Equipo de cómputo Plataforma digital IDE de desarrollo Software orientado a la unidad de competencia Internet Proyector **Recursos Didácticos:** Video tutoriales Repositorio digital con material de consulta Esquemas Gráficos Infografías	Informe con el proceso de solución de los retos.	**Instrumento de Evaluación:** • Lista de cotejo **Criterios de Evaluación:** Criterios de evaluación de forma: • El informe cuenta con una portada que incluye los datos de identificación del estudiante. • Los textos están creados con fuente Arial 12, interlineado intermedio. Criterios de evaluación de fondo: • El informe está conformado por un análisis, representación gráfica, codificación, pruebas del software y conclusión. • Implementa componentes y controles para presentar interfaces gráficas de usuario. • Implementa formularios con campos de entrada acorde al tipo de dato que corresponde. • La redacción del informe mantiene un tono académico, profesional, critico y responsable.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Unidad Didáctica 3:	Interfaces gráficas	**Nivel:**	4to
Propósito General:	Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		
Unidad de Competencia No 3:	Crea software con interfaces gráficas para mejorar la experiencia de la interacción con el usuario, mediante componentes de entrada, salida, controles y eventos considerando las cualidades de usabilidad.		
Aprendizaje Esperado No 2:	Demuestra software con funcionalidades para manipular información implementando eventos para crear interactividad entre la interfaz gráfica y el usuario, intuitivo y fácil de usar.	**Tiempo estimado para obtener el Aprendizaje Esperado:**	12 horas

Contenidos de Aprendizaje

Conceptuales	Procedimentales	Actitudinales
3.2 Programación orientada a eventos 3.2.1 Eventos con el mouse 3.2.2 Eventos con el teclado 3.2.3 Menús interactivos 3.2.4 Cajas de dialogo 3.2.5 Métodos para guardar, registrar, editar y mostrar información. 3.2.1 Tablas y Arboles	Implementa una interfaz gráfica para introducir, buscar y mostrar información. Integra elementos interactivos en la interfaz gráfica para la facilidad de uso del software. ✓ **Práctica 6. "**Eventos"	Aplica la creatividad para dar solución a problemas. Actúa de manera ética y responsable.

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje Basado en Retos

Apertura: El docente mediante una presentación abordará los conceptos de eventos con el mouse, eventos con el teclado, menús interactivos, cajas de diálogo, métodos para guardar, registrar, editar y mostrar información, tablas y arboles.

Desarrollo: el docente en las sesiones de laboratorio con el IDE de desarrollo para aplicaciones en un computadora personal mostrará a los estudiantes como hacer la integración de los eventos de mouse y teclado en el software, así como el despliegue de mensajes de dialogo.

Se realizará la Práctica 6. "Eventos" (después del subtema 3.2.4 Cajas de dialogo)

El docente propondrá un reto de alto nivel en el que se pueda desarrollar una aplicación de software con interfaz gráfica capaz y mensajes de dialogo. Los estudiantes serán capaces de organizarse en quipo para dar solución al reto propuesto, tendrán que realizarlo con el lenguaje de programación e IDE de desarrollo, crearan la interfaz gráfica que permita introducir, buscar, mostrar información y guardar en archivos, la aplicación deberá mostrar mensajes de dialogo para facilitar su uso. Es requerido que los docentes brinden un acompañamiento presencial en el laboratorio para solucionar dudas, realizar evaluaciones y realimentaciones a los estudiantes .

Cierre: De forma colaborativa los estudiantes realizarán un informe detallado con la metodología usada y expondrán la solución

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Ambiente de Aprendizaje: Aula y laboratorio

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Instrumento y Criterios de Evaluación
Herramientas Tecnológicas: Equipo de cómputo Plataforma digital IDE de desarrollo Software orientado a la unidad de competencia Internet Proyector **Recursos Didácticos:** Video tutoriales Repositorio digital con material de consulta Esquemas Gráficos Infografías	Informe con la metodología de solución	**Instrumento de Evaluación:** • Lista de cotejo **Criterios de Evaluación:** Criterios de evaluación de forma: • El informe cuenta con una portada que incluye los datos de identificación del estudiante. • Los textos están creados con fuente Arial 12, interlineado intermedio. • En el repositorio digital los archivos se encuentran ordenados y con la nomenclatura indicada. Criterios de evaluación de fondo: • El informe está conformado por un análisis, representación gráfica, codificación, pruebas del software y conclusión. • Implementa eventos de mouse y teclado en las interfaces gráficas. • Implementa menús y Integra elementos interactivos en la interfaz gráfica para la facilidad de uso del software. • La redacción del informe mantiene un tono académico y profesional, critico y responsable. • Muestra una aplicación con interfaz gráfica que permita al usuario de introducir, buscar y mostrar información y guardar información en archivos. • La aplicación de software muestra mensajes de dialogo que permiten al usuario usarla de manera fácil e intuitiva.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

PRÁCTICAS

Nombre de la Práctica:	Hola mundo	**N° de la Práctica:**	1	**Tiempo:**	9 hrs.
Unidad de Competencia:	Usa los elementos del software para programar con base en su estructura, comportamiento e interacción, tomando en cuenta las propiedades de herencia, encapsulamiento, polimorfismo y abstracción, respetando la simbología y buenas prácticas de programación.				
Aprendizajes Esperados Relacionados con la Práctica:	Clasifica escenarios para aplicar los conceptos de clase, método y objeto utilizando las propiedades de abstracción, encapsulamiento, herencia y polimorfismo de forma ordenada y funcional.				

Contenidos de Aprendizaje Relacionados con la Práctica

Conceptuales	Procedimentales	Actitudinales
1.1 Conceptos básicos de programación orientada a objetos 1.1.1 Clase	Revisa las herramientas de programación orientada a objetos disponibles en los sistemas de cómputo.	Desarrolla sus actividades de una forma analítica

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje basado en problemas.
El docente muestra la instalación y configuración del entorno de desarrollo para mostrar un mensaje "Hola Mundo"
Los estudiantes reconocen los elementos del entorno de desarrollo, instalan los elementos requeridos, codifican una aplicación que compile y ejecute un saludo "Hola Mundo" utilizando los principios de Programación Orientada a Objetos.
Los estudiantes en su computadora compila y ejecuta un saludo "Hola Mundo" y elaboran un informe con el proceso de instalación, configuración, compilación y ejecución del programa.
Ambiente de Aprendizaje: Laboratorio.

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Criterios e Instrumentos de Evaluación
Herramientas Tecnológicas: IDE de desarrollo **Recursos Didácticos:** Presentación en PowerPoint	Programa Orientado a Objetos mostrando un saludo.	**Instrumento de Evaluación:** • Lista de cotejo **Criterios de Evaluación:** • Verifica la instalación del entorno de desarrollo • Configura correctamente los elementos del entorno de desarrollo • Codifica un programa con los principios de programación orientada a objetos • El programa compila y ejecuta sin mostrar errores de sintaxis o lógicos.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Nombre de la Práctica:	Modelado de software	**N° de la Práctica:**	2	**Tiempo:**	9 hrs.
Unidad de Competencia:	Usa los elementos del software para programar con base en su estructura, comportamiento e interacción, tomando en cuenta las propiedades de herencia, encapsulamiento, polimorfismo y abstracción, respetando la simbología y buenas prácticas de programación.				
Aprendizajes Esperados Relacionados con la Práctica:	Clasifica escenarios para aplicar los conceptos de clase, método y objeto utilizando las propiedades de abstracción, encapsulamiento, herencia y polimorfismo de forma ordenada y funcional.				

Contenidos de Aprendizaje Relacionados con la Práctica

Conceptuales	Procedimentales	Actitudinales
1.2 UML 1.2.1 Definición 1.2.2 Elementos 1.2.2.1 Diagrama de casos de uso 1.2.2.2 Diagrama de clases 1.2.2.3 Diagrama de secuencias 1.2.2.4 Diagrama de colaboración 1.2.2.5 Diagrama de estado 1.2.2.6 Diagrama de actividades 1.2.2.7 Diagrama de componentes	Emplea los principios de UML (Unified Modeling Language) para modelar software.	Desarrolla sus actividades de una forma analítica

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje basado en problemas
El docente plantea un problema en el que es requerido modelar una aplicación de software con UML.
Los estudiantes analizan el problema, modelan la aplicación del software empleando los diagramas de casos de uso, clases, secuencia, colaboración, estado, actividades y componentes.
Los estudiantes generan un informe con el modelado de la aplicación,

Ambiente de Aprendizaje: Laboratorio

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Criterios e Instrumentos de Evaluación
Herramientas Tecnológicas: IDE de desarrollo Lenguaje UML Software para crear modelos UML Procesador de textos **Recursos Didácticos:** Presentación en PowerPoint	Informe de los diagramas realizados con lenguaje UML	**Instrumento de Evaluación:** • Guía de observación **Criterios de Evaluación:** • Aplica el modelado correspondiente al planteamiento del problema. • Utiliza los símbolos de UML. • Genera al menos un diagrama de casos de uso, clases, secuencia, colaboración, estado, actividades y componentes. • Integra una conclusión.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Nombre de la Práctica:	Listas genéricas	N° de la Práctica:	3	Tiempo:	9 hrs.
Unidad de Competencia:	Demuestra software para resguardar información usando elementos de persistencia de datos y manejo de archivos de forma ordenada, analítica y responsable				
Aprendizajes Esperados Relacionados con la Práctica:	Practica programas para almacenar, consultar y quitar información de forma temporal utilizando clases, objetos, métodos, arreglos, listas y vectores de forma persistente y ordenada.				

Contenidos de Aprendizaje Relacionados con la Práctica

Conceptuales	Procedimentales	Actitudinales
2.1 Listas genéricas de matrices 2.1.2 Tipos de listas, Composición y características de las listas genéricas 2.1.2 Declaración de características de clases con métodos get() y set(). 2.1.3 Acceso e iteración 2.1.5 Algoritmos para almacenar, consultar y quitar elementos de las listas.	Demuestra programas de listas genéricas con algoritmos para almacenar, consultar y quitar información de las listas genéricas.	Capacidad para analizar y resolver casos

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje basado en problemas
El docente plantea un problema que deberá ser resuelto mediante la aplicación de algoritmos para almacenar, consultar y quitar elementos de las listas.
Los estudiantes en equipo analizan el problema, identifican variables y métodos a utilizar, crean el algoritmo, realizan un modelado y codifican el algoritmo.
Demuestra la funcionalidad para almacenar, consultar y quitar elementos.
Ambiente de Aprendizaje: Laboratorio

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Criterios e Instrumentos de Evaluación
Herramientas Tecnológicas: IDE de desarrollo Procesador de textos Software para crear diagramas UML **Recursos Didácticos:** Presentación en PowerPoint	Aplicación de software capaz de realizar almacenar, consultar y quitar elementos.	**Instrumento de Evaluación:** • Guía de observación **Criterios de Evaluación:** • Integra listas genéricas como elemento de almacenamiento. • Codifica una aplicación de software que permite almacenar información • La información almacena se puede consultada • Es posible quitar información

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Nombre de la Práctica:	Manejo de archivos	**N° de la Práctica:**	4	**Tiempo:**	9 hrs.
Unidad de Competencia:	Demuestra software para resguardar información usando elementos de persistencia de datos y manejo de archivos de forma ordenada, analítica y responsable				
Aprendizajes Esperados Relacionados con la Práctica:	Pone a prueba software para guardar, buscar y eliminar información de manera definitiva implementando archivos de forma analítica y responsable.				

Contenidos de Aprendizaje Relacionados con la Práctica

Conceptuales	Procedimentales	Actitudinales
2.2.3 Secuencias y flujo de datos 2.2.4 Manejo de ficheros	Realiza software capaz de resguardar, eliminar y editar información en archivos.	Capacidad para analizar y resolver casos

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje basado en problemas
El docente plantea un problema que deberá ser resuelto mediante la implementación de archivos
Los estudiantes en equipo analizan el problema, identifican variables, integra librerías para el manejo de archivos y codifica una aplicación para el manejo de información con archivos.
Demuestra la funcionalidad para resguardar, eliminar y editar información en archivos.
Ambiente de Aprendizaje: Laboratorio

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Criterios e Instrumentos de Evaluación
Herramientas Tecnológicas: IDE de desarrollo Procesador de textos Software para crear diagramas UML **Recursos Didácticos:** Presentación en PowerPoint	Aplicación de software con las funcionalidades de resguardar, eliminar y editar información en archivos.	**Instrumento de Evaluación:** • Guía de observación **Criterios de Evaluación:** • Integran listas archivos como elemento de almacenamiento. • Codifica una aplicación de software que permite resguardar información • La información almacena se puede consultada • Es posible editar información. • Se puede eliminar información de los archivos.

INSTITUTO POLITÉCNICO NACIONAL
Dirección de Educación Media Superior

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Nombre de la Práctica:	Interfaz gráfica	**N° de la Práctica:**	5	**Tiempo:**	9 hrs.
Unidad de Competencia:	Crea software con interfaces gráficas para mejorar la experiencia de la interacción con el usuario, mediante componentes de entrada, salida, controles y eventos considerando las cualidades de usabilidad				
Aprendizajes Esperados Relacionados con la Práctica:	Ordena una interfaz gráfica para permitir al usuario ingresar y visualizar información empleando componentes de entrada, salida y controles, fácil de operar y usar.				

Contenidos de Aprendizaje Relacionados con la Práctica

Conceptuales	Procedimentales	Actitudinales
3.1.2 Contenedores 3.1.3 Componentes de entrada 3.1.4 Controles y sus propiedades 3.1.5 Jerarquía de componentes en una aplicación 3.1.6 Gráficos, iconos e imágenes 3.1.7 Mensajes de dialogo	Ejemplifica componentes y controles para presentar interfaces gráficas de usuario.	Aplica la creatividad para dar solución a casos.

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje basado en problemas
El docente presenta el bosquejo de una interfaz gráfica indicando la particularidad de cada elemento
Los estudiantes la analizan el bosquejo, identifican contenedores, componentes de entrada y controles para construir una interfaz gráfica.
Muestra una interfaz gráfica con todos los componentes y controles solicitados.
Ambiente de Aprendizaje: Laboratorio

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Criterios e Instrumentos de Evaluación
Herramientas Tecnológicas: IDE de desarrollo Procesador de textos Software para crear diagramas UML **Recursos Didácticos:** Presentación en PowerPoint	Interfaces gráficas de usuario con componentes y controles.	**Instrumento de Evaluación:** • Guía de observación **Criterios de Evaluación:** • Los elementos están contenidos en un contenedor • Integra componentes de entrada • Emplea controles • La interfaz gráfica corresponde al bosquejo proporcionado.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

Nombre de la Práctica:	**Eventos**	**N° de la Práctica:**	6	**Tiempo:**	9 hrs.
Unidad de Competencia:	Crea software con interfaces gráficas para mejorar la experiencia de la interacción con el usuario, mediante componentes de entrada, salida, controles y eventos considerando las cualidades de usabilidad.				
Aprendizajes Esperados Relacionados con la Práctica:	Demuestra software con funcionalidades para manipular información implementando eventos para crear interactividad entre la interfaz gráfica y el usuario, intuitivo y fácil de usar.				

Contenidos de Aprendizaje Relacionados con la Práctica

Conceptuales	Procedimentales	Actitudinales
3.2.1 Eventos con el mouse 3.2.4 Cajas de dialogo	Integra elementos interactivos en la interfaz gráfica para la facilidad de uso del software.	Aplica la creatividad para dar solución a problemas.

Estrategia Didáctica y Ambiente de Aprendizaje

Estrategia Didáctica: Aprendizaje basado en problemas
El docente presenta un problema que debe ser resuelto con eventos del mouse y cajas de dialogo.
Los estudiantes la analizan el problema planteado, identifican los flujos de interacción, crean un algoritmo, diagrama de flujo y codifican una interfaz gráfica con botones, eventos y cajas de dialogo.
Muestra una interfaz gráfica con botones, eventos y cajas de dialogo.
Ambiente de Aprendizaje: Laboratorio

Herramientas Tecnológicas y Recursos Didácticos	Evidencia de Aprendizaje para la Evaluación Formativa	Criterios e Instrumentos de Evaluación
Herramientas Tecnológicas: IDE de desarrollo Procesador de textos Software para crear diagramas UML **Recursos Didácticos:** Presentación en PowerPoint	Interfaces gráficas con eventos, controles y cajas de dialogo.	**Instrumento de Evaluación:** • Guía de observación **Criterios de Evaluación:** • Integra botones con eventos • El evento genera otra acción • Emplea cajas de dialogo para mejorar la experiencia de usuario. • No se generan errores al interactuar con la aplicación.

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

PLAN DE EVALUACIÓN SUMATIVA

N°	Unidad de Competencia	Evidencia Integradora	Criterios e Instrumento de Evaluación	Porcentaje de Acreditación
1	Usa los elementos del software para programar con base en su estructura, comportamiento e interacción, tomando en cuenta las propiedades de herencia, encapsulamiento, polimorfismo y abstracción, respetando la simbología y buenas prácticas de programación.	Aplicación de software creada a partir de un modelo UML	Instrumento de evaluación: • Rúbrica Criterios de evaluación: • Utiliza los principios de Programación Orientada a Objetos para codificar. • La aplicación no presenta errores de sintaxis • La aplicación no presenta errores lógicos. • Identifica los requerimientos funcionales. • Los diagramas UML representan la estructura, comportamiento e interacción.	30 %
2	Demuestra software para resguardar información usando elementos de persistencia de datos y manejo de archivos de forma ordenada, analítica y responsable	Aplicación de software con funcionalidades para manipular información en archivos	Instrumento de evaluación: • Rúbrica Criterios de evaluación: • Emplea clases de tipo entidad. • Implementa métodos para obtener y modificar información • La información en los archivos se mantiene integra al cerrar y cargar el programa. • La aplicación del software emplea algoritmos de almacenar, consultar y quitar información en archivos.	30 %

			• La aplicación es libre de errores de sintaxis y lógicos.	
3	Crea software con interfaces gráficas para mejorar la experiencia de la interacción con el usuario, mediante componentes de entrada, salida, controles y eventos considerando las cualidades de usabilidad	Aplicación de software con interfaz gráfica	Instrumento de evaluación: • Rúbrica Criterios de evaluación: • Demuestra que el desarrollo de la aplicación fue con base en un análisis, modelado UML y codificación con un lenguaje de programación. • Implementa eventos de mouse y teclado en las interfaces gráficas. • Integra menús y elementos. interactivos en la interfaz gráfica para la facilidad de uso del software. • Muestra una aplicación con interfaz gráfica que permita al usuario de introducir, buscar, mostrar y guardar información en archivos. • La aplicación de software muestra mensajes de dialogo que permiten al usuario usarla de manera fácil e intuitiva.	40 %
Propósito de la Unidad de Aprendizaje		**Evidencia Integradora**	**Criterios e Instrumento de Evaluación**	**Porcentaje de Acreditación**
Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.		Aplicación de software con interfaz gráfica desarrollada con los principios de programación orientada a objetos a partir de un modelo UML	Instrumento de evaluación: • Rúbrica Criterios de evaluación:	**100%**

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

		• En un informe define las características de las funcionalidades para introducir, mostrar, editar y guardar información. • Realiza un modelado del software empleando el Lenguaje de Modelado Unificado UML. • Crea un bosquejo de interfaz gráfica. • Emplea la arquitectura y propiedades de la Programación Orientada a Objetos. • Hace uso de listas para almacenar información. • Emplea algoritmos para registrar, mostrar, editar y guardar información en archivos. • Integra formularios con campos acorde a los tipos de dato a almacenar. • Integra componentes y controles de interfaz gráfica. • Hace uso de cuadros de diálogo para dar usabilidad a la aplicación. • La aplicación funciona de forma eficiente y eficaz para cada una de las funcionalidades planeadas.	

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

PROGRAMA SINTÉTICO

PROPÓSITO DE LA UNIDAD DE APRENDIZAJE

Desarrolla aplicaciones de software con interfaces gráficas para dar solución a problemas de diversos contextos utilizando la arquitectura y propiedades de la Programación Orientada a Objetos, de forma responsable, creativa, eficaz y eficiente.

N°	UNIDAD DE COMPETENCIA	APRENDIZAJES ESPERADOS	CONTENIDOS DE APRENDIZAJE/SABERES
1	Usa los elementos del software para programar con base en su estructura, comportamiento e interacción, tomando en cuenta las propiedades de herencia, encapsulamiento, polimorfismo y abstracción, respetando la simbología y buenas prácticas de programación.	1. Clasifica escenarios para aplicar los conceptos de clase, método y objeto utilizando las propiedades de abstracción, encapsulamiento, herencia y polimorfismo de forma ordenada y funcional. 2. Usa los principios del modelado de clases en UML en la solución de problemas en un entorno real, utilizando diagramas de clases, diagramas de actividades, entre otros. De forma ordenada y funcional.	**Conceptual:** 1.1 Conceptos básicos de programación orientada a objetos 1.1.1 Clase 1.1.2 Objeto 1.1.3 Abstracción 1.1.4 Encapsulamiento 1.1.5 Herencia 1.1.6 Polimorfismo 1.2 UML 1.2.1 Definición 1.2.2 Elementos 1.2.2.1 Diagrama de casos de uso 1.2.2.2 Diagrama de clases 1.2.2.3 Diagrama de secuencias 1.2.2.4 Diagrama de colaboración 1.2.2.5 Diagrama de estado 1.2.2.6 Diagrama de actividades 1.2.2.7 Diagrama de componentes **Procedimental:** Reconoce las herramientas de programación orientada a objetos disponibles en los sistemas de cómputo. **Práctica 1**. Hola mundo (Se realiza después del subtema 1.1.2 Objeto) Emplea los principios de UML (Unified Modeling Language) de forma ordenada y funcional en la solución de problemas de un entorno real, por medio de diagramas de clase, diagramas de actividades, entre otros para la elaboración de los programas informáticos.

			Práctica 2. Modelado de software. (se realiza después del subtema 1.2.2.7 Diagrama de componentes) **Actitudinal:** Muestra una actitud respetuosa hacia sus compañeros y docentes, el desarrollo de sus actividades lo realiza de una forma analítica, reflexiva, con una comunicación efectiva y el trabajo se cumple de forma colaborativa.
2	Demuestra software para resguardar información usando elementos de persistencia de datos y manejo de archivos de forma ordenada, analítica y responsable	1. Practica programas para almacenar, consultar y quitar información de forma temporal utilizando clases, objetos, métodos, arreglos, listas y vectores de forma persistente y ordenada. 2. Pone a prueba software para guardar, buscar y eliminar información de manera definitiva implementando archivos de forma analítica y responsable.	**Conceptual:** 2.1 Listas genéricas de matrices 2.1.2 Tipos de listas, Composición y características de las listas genéricas 2.1.2 Declaración de características de clases con métodos get() y set(). 2.1.3 Acceso e iteración 2.1.5 Algoritmos para almacenar, consultar y quitar elementos de las listas. 2.2. Archivos 2.2.1 Características de un archivo 2.2.2 Lectura y escritura de bytes 2.2.3 Secuencias y flujo de datos 2.2.4 Manejo de ficheros **Procedimental:** Demuestra programas de listas genéricas con algoritmos para almacenar, consultar y quitar información de las listas genéricas. **Práctica 3**. Listas genéricas. (Se realiza después del subtema 2.1.5 Algoritmos para almacenar, consultar y quitar elementos de las listas) Realiza software capaz de resguardar, eliminar y editar información en archivos. **Práctica 4.** Manejo de archivos. (Se realiza al final del subtema 2.2.4 Manejo de ficheros) **Actitudinal:** Hace uso del pensamiento criticó y reflexivo Capacidad para analizar y resolver casos

			Usa el pensamiento crítico para identificar y reconocer errores. Demuestra habilidad de resolución de casos.
3	Crea software con interfaces gráficas para mejorar la experiencia de la interacción con el usuario, mediante componentes de entrada, salida, controles y eventos considerando las cualidades de usabilidad	1. Ordena una interfaz gráfica para permitir al usuario ingresar y visualizar información empleando componentes de entrada, salida y controles, fácil de operar y usar. 2. Demuestra software con funcionalidades para manipular información implementando eventos para crear interactividad entre la interfaz gráfica y el usuario, intuitivo y fácil de usar.	**Conceptuales** 3.1 Interfaz gráfica de usuario 3.1.1 Arquitectura Modelo-Vista-Controlador 3.1.2 Contenedores 3.1.3 Componentes de entrada 3.1.4 Controles y sus propiedades 3.1.5 Jerarquía de componentes en una aplicación 3.1.6 Gráficos, iconos e imágenes 3.1.7 Mensajes de dialogo 3.2 Programación orientada a eventos 3.2.1 Eventos con el mouse 3.2.2 Eventos con el teclado 3.2.3 Menús interactivos 3.2.4 Cajas de dialogo 3.2.5 Métodos para guardar, registrar, editar y mostrar información. 3.2.1 Tablas y Arboles **Procedimental** Ejemplifica componentes y controles para presentar interfaces gráficas de usuario. Implementa formularios para clasificar datos de entrada. **Práctica 5.** Interfaz gráfica. (Se realiza después del subtema 3.1.7 Mensajes de dialogo) Implementa una interfaz gráfica para introducir, buscar y mostrar información. Integra elementos interactivos en la interfaz gráfica. **Práctica 6.** Eventos (Se realiza después del subtema 3.2.4 Cajas de dialogo) **Actitudinal** Aplica la creatividad para dar solución a problemas. Actúa de manera ética y responsable

Programa Académico: Técnico en Programación

Unidad de Aprendizaje: Programación Intermedia

BIBLIOGRAFÍA BÁSICA Y COMPLEMENTARIA

Número y Nombre de la Unidad Didáctica	FORMATO APA	CLASIFICACIÓN	
		Básico	Consulta
Unidad 1: Arquitectura del software y propiedades de Programación Orientada a Objetos	Booch,G. (2000). Análisis y diseño orientado a objetos con aplicaciones. México: Pearson Educación.		X
Unidad 1: Arquitectura del software y propiedades de Programación Orientada a Objetos	Fowler, M.,y Kendall, S. (2000). UML gota a gota. México: Prentice-Hall.		X
Unidad 1: Arquitectura del software y propiedades de Programación Orientada a Objetos	Parga, C., (2015). UML aplicaciones en Java y C++. Ciudad de México, México: Ra-Ma.	X	
Unidad 1: Arquitectura del software y propiedades de Programación Orientada a Objetos Unidad 2: Persistencia de datos y manejo de archivos	Joyanes, L., Zahonero, I. (2014). Programación en C, C++, Java y UML. Ciudad de México, México: McGraw-Hill.	X	
Unidad 2: Persistencia de datos y manejo de archivos Unidad 3: Interfaces gráficas	Bell, D., y Parr, M. (2011). Java para estudiantes (6a ed.). Estado de México, Naucalpan de Juárez: Pearson Educación.		X
Unidad 2: Persistencia de datos y manejo de archivos Unidad 3: Interfaces gráficas	Ceballos, J. G. (2011). Java 2-curso de programación (4a ed.), Alfaomega editores.		X
Unidad 2: Persistencia de datos y manejo de archivos Unidad 3: Interfaces gráficas	Deitel, H. M., y Deitel, P. J. (2011). Como programar en C/C ++ y Java (4a ed.). Estado de México, Naucalpan de Juárez: Pearson.	X	
Unidad 2: Persistencia de datos y manejo de archivos Unidad 3: Interfaces gráficas	Sierra, F. J. (2011). Java 2 Curso de programación (4a ed.). España: Alfaomega Ra-Ma.	X	
Unidad 2: Persistencia de datos y manejo de archivos Unidad 3: Interfaces gráficas	Horstmann C. S., Cornell Gary (2021) Core Java 2 Volumen I Fundamentos, 12va edición, Oracle Press.	X	

PODER EJECUTIVO FEDERAL ESTADOS UNIDOS MEXICANOS MÉXICO CDMX

INSTITUTO POLITÉCNICO NACIONAL
Dirección de Educación Media Superior