1. Datos Generales de la asignatura

Nombre de la asignatura:	Fundamentos de programación
Clave de la asignatura:	IAD-2413
SATCA[1]:	2-3-5
Carrera:	Ingeniería en Inteligencia Artificial

2. Presentación

Caracterización de la asignatura

La asignatura de fundamentos de programación es fundamental para el desarrollo de competencias básicas en programación, que son esenciales para un ingeniero en inteligencia artificial. Esta asignatura proporciona a los estudiantes los conocimientos y habilidades necesarios para escribir, entender y analizar programas de computadora, que son la base para el desarrollo de algoritmos y soluciones de inteligencia artificial. Los egresados podrán desarrollar soluciones en inteligencia artificial combinando habilidades de programación y conocimientos técnicos para integrar sistemas completos.

La asignatura es crucial ya que sienta las bases para todas las habilidades de programación avanzadas necesarias en el campo de la inteligencia artificial. Sin una comprensión sólida de los fundamentos de programación, los estudiantes no podrán abordar con éxito tareas más complejas como la implementación de algoritmos de aprendizaje automático, el procesamiento de grandes volúmenes de datos, o la optimización de modelos de IA.

La asignatura de fundamentos de programación abarca los conceptos básicos de la programación que son esenciales para el desarrollo de habilidades más avanzadas en el campo de la inteligencia artificial. Los estudiantes aprenderán sobre los tipos de datos y estructuras básicas, operadores y expresiones, control de flujo mediante condicionales y bucles, y la importancia de las funciones y la modularización en el diseño de software. Además, se introduce el manejo de cadenas y archivos, así como una primera aproximación a las estructuras de datos como listas, pilas y colas. Un componente clave del curso es la depuración y las pruebas de programas, habilidades fundamentales para garantizar la funcionalidad y la eficiencia del código.

La relación de fundamentos de programación con otras materias es fundamental para el desarrollo de competencias específicas en la carrera de ingeniería en inteligencia artificial. Con algoritmia y estructuras de datos, los conocimientos adquiridos sobre tipos de datos, control de flujo y funciones se profundizan y se aplican en el diseño y análisis de algoritmos, así como en la optimización de código. Esto es esencial para el desarrollo de aplicaciones que requieren la manipulación eficiente de grandes volúmenes de información.

[1] Sistema de Asignación y Transferencia de Créditos Académicos

©TecNM 2024

En programación orientada a objetos, los conceptos de modularización y manejo de funciones aprendidos en fundamentos de programación se expanden para incluir el diseño de clases y objetos, herencia y polimorfismo. Este enfoque es crucial para la creación de sistemas de gestión que permiten el seguimiento y análisis de datos de manera estructurada y reutilizable.

En el contexto de bases de datos, el manejo de archivos aprendido en fundamentos de programación se extiende a la conexión y manipulación de bases de datos, permitiendo a los estudiantes realizar consultas y gestionar datos de manera efectiva. La integración de estos conocimientos facilita el desarrollo de aplicaciones que requieren operaciones CRUD (create, read, update, delete).

Intención didáctica

La intención didáctica de la asignatura de fundamentos de programación es proporcionar a los estudiantes una comprensión sólida y práctica de los conceptos básicos de la programación, sentando las bases para su desarrollo como ingenieros en inteligencia artificial.

Abordaje de contenidos:
los contenidos se abordan de manera progresiva y estructurada, comenzando con conceptos simples como tipos de datos y operadores, y avanzando hacia temas más complejos como el control de flujo, las funciones y las estructuras de datos. Se enfatiza la importancia de la práctica activa mediante la resolución de problemas y ejercicios prácticos, lo que permite a los estudiantes internalizar los conceptos teóricos y desarrollar habilidades prácticas de programación.

Enfoque de tratamiento:
el enfoque de tratamiento es práctico y orientado a la resolución de problemas. Se fomenta el pensamiento lógico y algorítmico, así como la capacidad de descomponer problemas en pasos lógicos y diseñar soluciones efectivas utilizando los conocimientos adquiridos. Se promueve el aprendizaje autónomo mediante la experimentación y la búsqueda de recursos adicionales, lo que ayuda a los estudiantes a desarrollar habilidades de resolución de problemas y autoaprendizaje que son fundamentales en el campo de la programación.

Extensión y profundidad de los contenidos:
los contenidos se presentan con la extensión y profundidad necesarias para sentar una base sólida en programación. Se cubren los conceptos básicos de manera exhaustiva, asegurando que los estudiantes comprendan los fundamentos antes de avanzar a temas más avanzados. Se incluyen ejemplos prácticos y casos de estudio que ilustran la aplicación de los conceptos en situaciones del mundo real, lo que ayuda a los estudiantes a comprender la relevancia y el impacto de la programación en diversos contextos.
En el tema 1 el estudiante realizara una introducción a los fundamentos de la programación, cubriendo conceptos básicos como variables, tipos de datos, operadores y estructuras de control de flujo. Se explicaría la importancia de la programación en la resolución de problemas y se brindaría una visión general de los diferentes paradigmas de programación.

©TecNM 2024

En el tema 2 los estudiantes se centran en el proceso de análisis y resolución de problemas utilizando la programación como herramienta. Se enseñarían técnicas para descomponer problemas complejos en pasos más simples, identificar patrones y diseñar algoritmos eficientes. Se promovería el desarrollo de habilidades de pensamiento lógico y crítico mediante la resolución de una variedad de problemas prácticos.

En el tema 3 se presentan las herramientas y entornos de desarrollo más comunes utilizados en la programación. Se enseñaría a instalar y configurar un entorno de desarrollo integrado (IDE), así como a utilizar herramientas de depuración y control de versiones. Se brindaría una introducción a la estructura de un programa típico y se explicaría el proceso de compilación y ejecución de un programa.

En el tema 4 los estudiantes aplican los conocimientos adquiridos en los temas anteriores para desarrollar aplicaciones prácticas. Se les proporcionaría un conjunto de problemas o proyectos y se les guiaría a través del proceso de análisis, diseño, implementación y prueba de soluciones utilizando los conceptos de programación aprendidos. Se fomentaría la creatividad y la experimentación, permitiendo a los estudiantes explorar diferentes enfoques para resolver los problemas planteados.

Se destacan actividades como la resolución de problemas, la escritura de código, la depuración y las pruebas de programas. Los estudiantes son alentados a participar activamente en clase, tanto de manera individual como en grupos, para discutir y resolver problemas prácticos. Se fomenta el desarrollo de proyectos pequeños que permitan a los estudiantes aplicar los conceptos aprendidos en situaciones concretas, lo que les brinda la oportunidad de poner en práctica sus habilidades de programación y desarrollar soluciones creativas.

Entre las competencias genéricas que se promueven se encuentran el pensamiento crítico, la resolución de problemas, la comunicación efectiva y el trabajo en equipo. Específicamente, se desarrollan habilidades como la capacidad de diseño de algoritmos, la depuración de código y la optimización de programas, que son fundamentales en el campo de la ingeniería en inteligencia artificial.

El papel del docente es facilitar el aprendizaje proporcionando orientación, apoyo y retroalimentación a los estudiantes. Se espera que el docente tenga un profundo conocimiento de los conceptos de programación y sea capaz de transmitirlos de manera clara y accesible. Además, el docente debe fomentar un ambiente de aprendizaje activo y colaborativo, donde los estudiantes se sientan motivados a participar y a compartir ideas. Se espera que el docente promueva el desarrollo de habilidades prácticas de programación, brindando oportunidades para la práctica y la experimentación, y estimulando el pensamiento crítico y la creatividad en la resolución de problemas

©TecNM 2024

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 06 de marzo del 2024.	Representantes de los Institutos Tecnológicos de: Celaya, Chihuahua, Iztapalapa III, La Paz, Matehuala, Mérida, Minatitlán, Querétaro, Saltillo, Tijuana. Institutos Tecnológico Superior de Teziutlán. Tecnológico de Estudios Superiores de Ixtapaluca.	Propuesta sintética de la carrera de Ingeniería en Inteligencia Artificial.
Tecnológico Nacional de México del 22 al 26 de abril del 2024	Representantes de los Institutos Tecnológicos de: Celaya, Chihuahua, Iztapalapa III, La Paz, Matehuala, Mérida, Minatitlán, Querétaro, Saltillo, Tijuana. Institutos Tecnológico Superior de Teziutlán, Tecnológico de Estudios Superiores de Ixtapaluca.	Diseño y/o desarrollo curricular de la carrera de Ingeniería en Inteligencia Artificial.
Tecnológico Nacional de México del 27 al 31 de mayo del 2024.	Representantes de los Institutos Tecnológicos de: Celaya, La Paz, Matehuala, Mérida, Minatitlán.	Consolidación curricular de la carrera de Ingeniería en Inteligencia Artificial.

4. Competencia(s) a desarrollar

Competencia(s) específica(s) de la asignatura
Aplica las herramientas de programación en la modelización y desarrolla soluciones para una variedad de problemas del mundo real.

5. Competencias previas

Antes de cursar fundamentos de programación se espera que los estudiantes tengan conocimientos básicos en el uso de computadoras, comprensión elemental de matemáticas, habilidades de pensamiento lógico y resolución de problemas, así como una actitud positiva hacia el aprendizaje tecnológico y la capacidad de aprender de forma autónoma. Estas competencias proporcionarán una base accesible para que los estudiantes, incluso aquellos con poca o ninguna experiencia previa en tecnología, puedan abordar con éxito el contenido del curso y desarrollar habilidades en programación.

©TecNM 2024

6. Temario

No.	Temas	Subtemas
1	Introducción a la programación y conceptos fundamentales.	1.1. Evolución de la programación. 1.2. Conceptos básicos de programación: variables, tipos de datos, operadores. 1.3. Sintaxis en diferentes lenguajes de programación. 1.4. Estructuras de control, condicionales y bucles. 1.5. Sintaxis para definición de funciones y métodos.
2	Análisis metodología de resolución de problemas.	2.1. Descripción y análisis del problema. 2.2. Definición de soluciones. 2.3. Diseño de algoritmos y soluciones. 2.4. Implementación de soluciones en código. 2.5. Depuración, pruebas y manejo de errores. 2.6. Documentación y buenas prácticas de programación. 2.7. Aplicación de la metodología en la resolución de problemas prácticos.
3	Herramientas de programación y desarrollo de aplicaciones.	3.1. Entornos de desarrollo integrados. 3.2. Uso de bibliotecas y módulos estándar. 3.3. Creación y organización de proyectos de software.
4	Desarrollo de aplicaciones prácticas.	4.1. Introducción Desarrollo de aplicaciones web básicas. 4.2. Introducción al desarrollo de aplicaciones móviles. 4.3. Creación de aplicaciones de escritorio simples. 4.4. Exploración de herramientas y frameworks para desarrollo de software.

©TecNM 2024

7. Actividades de aprendizaje de los temas

1. Introducción a la programación y conceptos fundamentales	
Competencias	Actividades de aprendizaje
Específica(s): El alumno adquiere las habilidades para resolver problemas mediante la comprensión de los principios de la programación, y comunicar de manera efectiva la descripción y análisis de objetos en diferentes contextos, tanto tangibles como intangibles, utilizando un lenguaje claro y preciso. *Genérica(s):* ● Capacidad de análisis y síntesis. ● Comunicación oral y escrita. ● Habilidades básicas de manejo de equipos y máquina en Robótica. ● Habilidad para buscar y analizar información proveniente de fuentes diversas. ● Trabajo en equipo ● Aplicar el pensamiento analítico, lógico, creativo e innovador para el análisis y la toma de decisiones. ● Compromiso ético. ● Capacidad de aprender. ● Habilidad para trabajar en forma autónoma. ● Búsqueda del logro.	● Investigación sobre hitos clave en la historia de la programación. ● Investigación de variables, tipos de datos y operadores en los diferentes lenguajes de programación. ● Ejercicios de codificación en Python, C++, Java y R. ● Implementación de condicionales y bucles en un contexto práctico. ● Creación de códigos con uso de funciones en distintos lenguajes de programación.
2. Análisis metodología de resolución de problemas	
Competencias	Actividades de aprendizaje
Específica(s): Los estudiantes aprenden a resolver problemas y comunicarse en programación mediante la unidad de análisis metodología de resolución de problemas, donde se desglosan problemas, definen soluciones, diseñan algoritmos, implementan código, depuran, prueban, documentan y aplican estas habilidades en situaciones prácticas.	● Los estudiantes realizarán un análisis detallado del problema propuesto, identificando sus componentes clave, requisitos y posibles desafíos. ● A partir del análisis previo, los estudiantes generarán posibles soluciones al problema, evaluando su viabilidad y eficacia. ● Se procederá al diseño de algoritmos y soluciones específicas para abordar el problema identificado, asegurándose de considerar la eficiencia y la escalabilidad.

©TecNM 2024

Genérica(s): ● Capacidad de análisis y síntesis. ● Comunicación oral y escrita ● Habilidades básicas de manejo de equipos y máquina en Robótica. ● Habilidad para buscar y analizar información proveniente de fuentes diversas. ● Trabajo en equipo. ● Aplicar el pensamiento analítico, lógico, creativo e innovador para el análisis y la toma de decisiones. ● Compromiso ético. ● Capacidad de aprender. ● Habilidad para trabajar en forma autónoma. ● Búsqueda del logro.	● Los estudiantes llevarán a cabo la implementación de las soluciones diseñadas en código, utilizando el lenguaje de programación adecuado y aplicando las mejores prácticas de programación. ● Se realizarán pruebas exhaustivas para detectar posibles errores en el código implementado, depurándolo y aplicando técnicas de manejo de errores para garantizar su funcionalidad y fiabilidad. ● Se elaborará documentación detallada que describa el proceso de desarrollo y funcionamiento del código, además de aplicar y promover buenas prácticas de programación para facilitar la comprensión y mantenimiento del código. ● Finalmente, los estudiantes aplicarán la metodología desarrollada a la resolución de problemas prácticos del mundo real, demostrando su capacidad para utilizar herramientas y técnicas de programación de manera efectiva y eficiente.
3. Herramientas de programación y desarrollo de aplicaciones	
Competencias	Actividades de aprendizaje
Específica(s): En Herramientas de Programación y Desarrollo de Aplicaciones, los estudiantes se introducen en el uso de IDEs, bibliotecas estándar, y estructuras de control como condicionales y bucles. También aprenden a organizar proyectos de software eficientemente. *Genérica(s):* ● Capacidad de análisis y síntesis. ● Comunicación oral y escrita. ● Habilidades básicas de manejo de equipos y máquina en Robótica. ● Habilidad para buscar y analizar información proveniente de fuentes diversas. ● Trabajo en equipo.	● Exploración y práctica con diferentes IDEs para programación. ● Implementación de funcionalidades mediante bibliotecas y módulos predefinidos. ● Dominio y aplicación de estructuras condicionales, bucles, selección y repetición en la programación. ● Desarrollo de habilidades para crear y organizar proyectos de software de manera eficiente.

©TecNM 2024

• Aplicar el pensamiento analítico, lógico, creativo e innovador para el análisis y la toma de decisiones. • Compromiso ético. • Capacidad de aprender. • Habilidad para trabajar en forma autónoma. • Búsqueda del logro.	
4. Desarrollo de aplicaciones prácticas	
Competencias	Actividades de aprendizaje
Específica(s): En desarrollo de aplicaciones prácticas, los estudiantes aprenden a crear aplicaciones web básicas con HTML, CSS y JavaScript, exploran el desarrollo de aplicaciones móviles para Android e iOS, y también abordan la creación de aplicaciones de escritorio simples con Java y C#. Además, se familiarizan con herramientas y frameworks populares para el desarrollo de software. *Genérica(s):* • Capacidad de análisis y síntesis. • Comunicación oral y escrita. • Habilidades básicas de manejo de equipos y máquina en Robótica. • Habilidad para buscar y analizar información proveniente de fuentes diversas. • Trabajo en equipo. • Aplicar el pensamiento analítico, lógico, creativo e innovador para el análisis y la toma de decisiones. • Compromiso ético. • Capacidad de aprender • Habilidad para trabajar en forma autónoma.	• Aprendizaje de los fundamentos para crear aplicaciones web simples y funcionales. • Exploración de los conceptos básicos del desarrollo de aplicaciones móviles con énfasis en los principios de diseño y desarrollo de interfaces de usuario. • Desarrollar aplicaciones de escritorio básicas, comprendiendo los principios de la programación orientada a objetos y la creación de interfaces de usuario. • Investigación y experimentación con herramientas y frameworks populares en el desarrollo de software, con el objetivo de entender su funcionamiento y aplicabilidad en diferentes contextos de desarrollo.

©TecNM 2024

8. Práctica(s)

Tema 1: Introducción a la programación y conceptos fundamentales
Práctica 1: Calculadora básica en varios lenguajes

Objetivo: Desarrollar una calculadora básica que permita realizar operaciones aritméticas simples (suma, resta, multiplicación y división) en diferentes lenguajes de programación (Python, C++, Java, y R).

Instrucciones:

1. **Evolución de la programación:** Investigar y escribir un breve resumen sobre la evolución de los lenguajes de programación.
2. **Conceptos básicos:**
   - Definir y utilizar variables, tipos de datos y operadores aritméticos.
3. **Sintaxis en diferentes lenguajes:**
   - Implementar la calculadora en Python, C++, Java y R.
4. **Estructuras de control:**
   - Utilizar condicionales (if-else) para manejar operaciones.
   - Utilizar bucles (for o while) para permitir múltiples cálculos sin reiniciar el programa.
5. **Funciones y métodos:**
   - Definir funciones para cada operación (suma, resta, multiplicación, división).

Entrega:
- Código fuente de la calculadora en los cuatro lenguajes.
- Informe breve sobre la evolución de los lenguajes de programación y comparación de sintaxis.

Tema 2: Análisis de resolución de problemas
Práctica 2: Resolución de problemas de clasificación de números

Objetivo: Desarrollar un programa que lea una lista de números y los clasifique en positivos, negativos y ceros.

Instrucciones:

1. **Descripción y análisis del problema:**
   - Describir el problema y analizar los requisitos.
2. **Definición de soluciones:**
   - Proponer diferentes enfoques para clasificar los números.
3. **Diseño de algoritmos:**
   - Diseñar un algoritmo que clasifique los números.
4. **Implementación:**
   - Implementar el algoritmo en un lenguaje de programación (por ejemplo, Python).

©TecNM 2024

5. **Depuración y pruebas:**
    - Realizar pruebas con diferentes conjuntos de datos y depurar errores.
6. **Documentación:**
    - Documentar el código y el proceso de desarrollo.
7. **Aplicación práctica:**
    - Aplicar el programa a un conjunto de datos real o simulado.

Entrega:

- Código fuente del programa.
- Documentación del análisis, diseño, implementación y pruebas.
- Ejemplos de datos utilizados y resultados obtenidos.

Tema 3: Herramientas de programación y desarrollo de aplicaciones
Práctica 3: Gestión de proyectos de software con un IDE

Objetivo: Crear y organizar un proyecto de software utilizando un entorno de desarrollo integrado (IDE) como Visual Studio Code, PyCharm, o Eclipse.

Instrucciones:

1. **Entornos de desarrollo integrados:**
    - Seleccionar un IDE y familiarizarse con sus características.
2. **Uso de bibliotecas y módulos:**
    - Crear un proyecto que haga uso de una biblioteca estándar (por ejemplo, math en Python).
3. **Creación y organización de proyectos:**
    - Estructurar el proyecto con directorios adecuados para código fuente, pruebas y documentación.

Actividad del proyecto:

- Desarrollar una aplicación simple que permita gestionar una lista de tareas (crear, editar, eliminar y marcar como completadas).
- Implementar pruebas unitarias para las funciones principales de la aplicación.

Entrega:

- Proyecto completo en el IDE seleccionado.
- Capturas de pantalla del entorno de desarrollo y la organización del proyecto.
- Código fuente y pruebas unitarias.

©TecNM 2024

Tema 4: Desarrollo de aplicaciones prácticas
Práctica 4: Desarrollo de una aplicación web básica

Objetivo: Desarrollar una aplicación web básica que permita a los usuarios registrar sus datos personales y visualizarlos en una página de perfil.

Instrucciones:

1. **Introducción a desarrollo de aplicaciones Web:**
   - Crear una página web con HTML para el formulario de registro.
   - Estilizar la página con CSS.

2. **Desarrollo de aplicaciones móviles:**
   - (Opcional) Extender la aplicación para que sea compatible con dispositivos móviles usando técnicas de diseño responsivo.
3. **Aplicaciones de escritorio simples:**
   - (Opcional) Crear una versión de la aplicación para escritorio utilizando un framework como Electron.
4. **Exploración de herramientas y frameworks:**
   - Utilizar un framework de desarrollo web como Flask (Python) o Spring Boot (Java) para manejar la lógica de backend y la persistencia de datos.

Entrega:

- Código fuente de la aplicación web.
- Capturas de pantalla de la aplicación en funcionamiento.
- Documentación que explique el diseño, implementación y pruebas realizadas.

9. Proyecto de asignatura

Título del Proyecto: plataforma de gestión de tareas colaborativas

Descripción del Proyecto: desarrollar una plataforma web que permita a los usuarios gestionar tareas de manera colaborativa. Los usuarios podrán crear, asignar, editar, eliminar y organizar tareas en diferentes proyectos, y colaborar con otros usuarios en tiempo real. Este proyecto integrador cubrirá los contenidos de todas las unidades del curso, integrando conceptos fundamentales de programación, análisis de problemas, uso de herramientas de desarrollo, y creación de aplicaciones prácticas.

©TecNM 2024

Fundamentación

Justificación: el desarrollo de una plataforma de gestión de tareas colaborativas es un proyecto significativo que simula un entorno de trabajo real, donde la colaboración y la gestión eficiente de tareas son esenciales. Este proyecto permite a los estudiantes aplicar y consolidar los conocimientos y habilidades adquiridos durante el curso, preparándolos para enfrentar desafíos reales en el desarrollo de software.

Competencias a Desarrollar:

- **Específicas:**
  - Aplicar conceptos fundamentales de programación.
  - Utilizar herramientas y entornos de desarrollo.
  - Diseñar e implementar algoritmos eficientes.
  - Documentar y comunicar el desarrollo de software.
- **Genéricas:**
  - Capacidad de análisis y síntesis.
  - Comunicación oral y escrita.
  - Trabajo en equipo.
  - Pensamiento crítico y creativo.
  - Habilidad para buscar y analizar información.
  - Compromiso ético y profesionalismo.
  - Aprendizaje autónomo y continuo.

Planeación

Fases del Proyecto:

1. **Definición del problema:**
   - Identificar las necesidades y requisitos de una plataforma de gestión de tareas colaborativas.
   - Realizar un análisis de usuario para definir características clave.
   - Establecer las especificaciones funcionales y no funcionales.
2. **Análisis y diseño:**
   - Descomponer el problema en componentes y módulos.
   - Diseñar la arquitectura del software utilizando diagramas de flujo y pseudocódigo.
   - Planificar la base de datos para gestionar usuarios, proyectos y tareas.
3. **Desarrollo:**
   - Implementar la interfaz de usuario con HTML, CSS y JavaScript.
   - Desarrollar la lógica de la aplicación en un lenguaje de programación adecuado (Python o Java).
   - Integrar la base de datos para gestionar el almacenamiento de datos.

©TecNM 2024

- Implementar funcionalidades de colaboración en tiempo real.

4. **Pruebas y depuración:**
   - Realizar pruebas unitarias y de integración para asegurar la funcionalidad.
   - Depurar errores y optimizar el rendimiento del software.
5. **Documentación:**
   - Documentar el proceso de desarrollo, incluyendo análisis, diseño, implementación y pruebas.
   - Crear una guía de usuario y una guía técnica para la plataforma.
6. **Entrega y presentación:**
   - Preparar una presentación final del proyecto, destacando los logros y aprendizajes.
   - Entregar el código fuente, la documentación y un informe final.

Ejecución

Semana 1: Definición del problema

- Reunión de equipo para discutir y acordar los requisitos del proyecto.
- Investigación y análisis de plataformas similares existentes.
- Redacción del documento de requisitos.

Semana 2: Análisis y diseño

- Creación de diagramas de flujo y diseño de la arquitectura.
- Planificación de la base de datos.
- Diseño de la interfaz de usuario (mockups y wireframes).

Semana 3-4: Desarrollo

- Codificación de la interfaz de usuario con HTML, CSS y JavaScript.
- Desarrollo de la lógica de negocio en Python o Java.
- Configuración e integración de la base de datos.
- Implementación de la funcionalidad de colaboración en tiempo real.

Semana 5: Pruebas y depuración

- Ejecución de pruebas unitarias y de integración.
- Depuración de errores y optimización del código.
- Revisión de la funcionalidad completa y ajustes finales.

©TecNM 2024

Semana 6: Documentación y presentación

- Redacción de la documentación técnica y de usuario.
- Preparación de la presentación final y demostración de la plataforma.
- Revisión final y entrega del proyecto completo.

Evaluación

Criterios de Evaluación:

- **Calidad del análisis y diseño:**
  - Claridad y precisión en la definición de requisitos.
  - Efectividad del diseño de la arquitectura del software.
- **Funcionalidad del software:**
  - Implementación correcta de todas las funcionalidades.
  - Usabilidad y experiencia del usuario de la plataforma.
- **Calidad del código:**
  - Eficiencia y organización del código fuente.
  - Uso adecuado de estructuras de control, funciones y manejo de errores
- **Documentación:**
  - Exhaustividad y claridad de la documentación técnica y de usuario.
  - Calidad de los diagramas y descripciones del proceso de desarrollo.
- **Presentación:**
  - Claridad y efectividad en la comunicación de los resultados.
  - Habilidad para responder preguntas y explicar decisiones de diseño y desarrollo.

- **Trabajo en equipo:**
  - Colaboración y contribución de cada miembro del equipo.
  - Manejo efectivo del tiempo y cumplimiento de plazos.

Instrumentos de Evaluación:

- Rúbrica de evaluación para análisis y diseño.
- Lista de verificación de funcionalidades implementadas.
- Revisión de la calidad del código fuente.
- Evaluación de la documentación entregada.
- Observación y retroalimentación de la presentación final.
- Autoevaluación y evaluación por pares del trabajo en equipo.

©TecNM 2024

Este proyecto integral permitirá a los estudiantes aplicar y consolidar sus conocimientos de manera práctica y significativa, desarrollando competencias esenciales para el desarrollo de software y preparándose para enfrentar desafíos reales en el campo.

10. Evaluación por competencias

Son las técnicas, instrumentos y herramientas sugeridas para constatar los desempeños académicos de las actividades de aprendizaje.

- Ejercicios y problemas en clase
- Exposición de temas por parte de los alumnos con apoyo y asesoría del profesor
- Evaluación trabajos de investigación entregados en forma escrita
- Evaluación por unidad para comprobar el manejo de aspectos teóricos y declarativos
- Evaluación de las prácticas por unidad, considerando los temas que ésta contiene
- Evaluación de las aplicaciones del contenido de la materia
- Considerar reporte de un proyecto final que describa las actividades realizadas y las conclusiones de este.

11. Fuentes de Información

1. García Serrano, A. (2016). Inteligencia Artificial: Fundamentos, práctica y aplicaciones. Bookwire GmbH.
2. Solares Riachi, D. M. S. (2021). Fundamentos De Programación. Estados Unidos: Palibrio.
3. Nolasco, J. S. (2021). Fundamentos de programación con Python 3. España: Marcombo.
4. Trejos Buriticá, O. I., Muñoz Guerrero, L. E. (2021). Introducción a la programación con Python. España: Ra-Ma S.A. Editorial y Publicaciones.
5. VEGAS, J. M. (2022). JAVA 17: Fundamentos prácticos de programación. Colombia: Ediciones de la U.
6. Ramírez Gil, C. M. (2023). Programación de Inteligencia Artificial. Curso Práctico. España: Ra-Ma S.A. Editorial y Publicaciones.
7. Pineda Pertuz, C. (2022). Aprendizaje automático y profundo en python: Una mirada hacia la inteligencia artificial. Colombia: Ediciones de la U.
8. Smyth, N. (2020). Android Studio 4. 0 Development Essentials - Java Edition: Build Android Apps with Android Studio 4. 0 and Java. Estados Unidos: Packt Publishing, Limited.
9. Smyth, N. (2023). Android Studio Electric Eel Essentials - Kotlin Edition: Developing Android Apps Using Android Studio 2022.1.1 and Kotlin. (n.p.): Payload Media

©TecNM 2024