Código de la Asignatura: IO3N1

Nombre del programa académico	Maestría en Investigación Operativa y Ciencia de Datos
Nombre completo de la asignatura	Nivelatorio en Lenguaje de Programación
Área académica o categoría	Investigación de Operaciones
Semestre y año de actualización	Primer semestre 2024
Tipo de asignatura	[x] Obligatoria [ ] Electiva
Número de créditos ECTS	0 ECTS
Director o contacto del programa	Eliana Mirledy Toro Ocampo
Coordinador o contacto de la asignatura	Alejandro Valencia Díaz – alejovd4512@utp.edu.co Google Scholar Alejandro Valencia

Descripción y Contenidos

1. Breve descripción

En esta asignatura se proporciona a los estudiantes una sólida introducción a la programación, abarcando desde los conceptos básicos hasta la aplicación práctica. Los estudiantes adquieren conocimientos fundamentales sobre la sintaxis del lenguaje, estructuras de control, tipos de datos y algoritmos básicos. Además, la asignatura busca que los estudiantes desarrollen habilidades de resolución de problemas y pensamiento lógico, preparando a los estudiantes para cursos avanzados y aplicaciones prácticas en programación.

2. Objetivos del programa académico

OP1. Analizar y combinar conocimientos y técnicas de la estadística y la optimización matemática bajo enfoques sostenibles, con el objetivo de tomar decisiones informadas, esenciales para abordar los desafíos actuales y futuros en ambientes académicos y empresariales. OP5. Fomentar dimensiones humanas que permitan generar espacios participativos de diálogo y reflexión con un enfoque sostenible, que contribuyan a la formación de profesionales capaces de abordar desafíos de manera integral y responsable

Objetivos de la asignatura

OA1. Estructurar los conceptos fundamentales de la computación (sintaxis, estructuras de control y algoritmos básicos) mediante el pensamiento crítico, permitiendo la traducción de modelos matemáticos a soluciones digitales eficientes.OA2. Desarrollar habilidades de resolución de problemas complejos mediante el uso de lenguajes de programación (como Python o R), garantizando que la implementación de algoritmos se realice bajo los principios de rigor y honestidad de la formación humana. OA3. Preparar al estudiante para operar en diversos contextos y paradigmas de programación, orientando el desarrollo de software hacia la sostenibilidad de los recursos computacionales y la transparencia en el manejo de la información.

3. Resultados de aprendizaje de la asignatura

RAA1. Diseña y construye programas básicos y avanzados (bucles anidados, funciones y condicionales complejos) empleando el pensamiento crítico, asegurando que la lógica del código sea coherente con la resolución de problemas de investigación operativa. RAA2. Implementa estructuras de datos adecuadas según la naturaleza del problema, optimizando la memoria y el procesamiento para garantizar la sostenibilidad del software desarrollado y la legibilidad del código. RAA3. Depura y corrige errores en códigos de programación con rigor científico, validando la integridad de los resultados bajo los estándares de honestidad de la formación humana. RAA4. Desarrolla programas eficientes y documentados, comunicando la lógica del algoritmo de forma transparente para fortalecer la ciudadanía y el trabajo colaborativo en entornos multidisciplinarios.

4. Contenido

T1: Introducción a la programación. T2: Variables y tipos de datos. T3: Estructuras de control. T4: Funciones y modularidad. T5: Estructuras de datos. T6: Programación orientada a objetos. T7: Depuración y optimización de código.

5. Requisitos: Conocimientos básicos de informática, razonamiento lógico y matemática básica.

6. Recursos: Libros ● Stroustrup, Bjarne (2023). Programming: Principles and Practice Using C++, y Zelle, John (2022).Python Programming: An Introduction to Computer Science. Artículos académicos y lecturas complementarias proporcionadas durante el curso.

Software de desarrollo integrado, como PyCharm, Visual Studio Code, RStudio, entre otros. Documentación y recursos en línea, como Stack Overflow, GitHub, entre otros.

7. Herramientas técnicas de soporte para la enseñanza: Talleres. Ejercicios prácticos en clase.

8. Trabajos en laboratorio y proyectos · Ejercicios prácticos para que los estudiantes practiquen la sintaxis y estructura de un lenguaje de programación mediante la creación de programas simples que realicen tareas específicas. · Ejercicios donde los estudiantes practiquen la identificación y corrección de errores en código proporcionado, ayudándoles a desarrollar habilidades de depuración.

9. Métodos de aprendizaje: Clases magistrales, Desarrollo de los ejercicios propuestos en clase.

10. Métodos de evaluación: Talleres en clase: contenidos presentados en T1, T2, T3, T4, T5, T6 y T7: (50%)· Trabajo en clase: contenidos presentados en T1, T2, T3, T4, T5, T6 y T7: (50%)