Materia: Programación

DATOS GENERALES:

Descripción:	Se presenta el arte de la programación, aplicando fundamentos y desarrollo de algoritmos implementados principalmente en Fortran, los cuales pueden ser aplicados a otros lenguajes de programación científica, tal como Python y C/C++.Durante el curso se deben incluir módulos que permitan al estudiante visualizar y graficar datos, también como los fundamentos básicos de la programación en paralelo.
Seriación y Correlación:	Subsecuentes: Álgebra lineal
	Consecuentes:Métodos Numéricos Probabilidad y Estadística Física de dispositivos semiconductores
Objetivo:	Proporcionar a los estudiantes las habilidades y conocimientos fundamentales en programación que les permitan aplicar eficazmente conceptos y métodos computacionales en la investigación, modelado y resolución de problemas en el campo de la física. A través de esta materia, los estudiantes aprenderán a diseñar, implementar y depurar programas informáticos, utilizarán herramientas y lenguajes de programación relevantes, y adquirirán la capacidad de automatizar tareas repetitivas, analizar datos experimentales y realizar simulaciones

Objetivos específicos:	- Desarrollar habilidades de programación: Capacitar a los estudiantes para que adquieran un sólido dominio de al menos un lenguaje de programación, lo que les permitirá escribir y comprender código de manera efectiva. - Fomentar el pensamiento algorítmico: Desarrollar la capacidad de los estudiantes para diseñar algoritmos efectivos y eficientes para resolver problemas físicos. - Evaluar y depurar programas: Desarrollar habilidades de depuración para que los estudiantes puedan identificar y corregir errores en sus programas. - Introducir herramientas computacionales: Familiarizar a los estudiantes con herramientas y bibliotecas computacionales utilizadas comúnmente en física, como fortran, lapack, gnuplot y mpi .
Horas totales del curso:	90 horas presenciales + 60 horas de autoestudio = 150 horas.
Créditos	10 Créditos

REVISIONES Y ACTUALIZACIONES:

Líneas de investigación:	Etapa básica, No aplica
Autores o Revisores:	Felipe Puch Ceballos, Daniel Acosta Escareño, Juan Carlos, Martinez Orozco, Alejandro Puga Candelas
Fecha de actualización por academia:	Octubre 2023
Sinopsis de la revisión y/o actualización:	

PERFIL DESEABLE DEL DOCENTE:

Disciplina profesional:	Doctorado en Ciencias básicas, Ingeniería y/o maestría en sistemas computacionales.
Experiencia docente:	Experiencia profesional docente mínima de dos años

ÍNDICE TEMÁTICO:

Tema	Descripción
Introducción a Algoritmos	- Definición de algoritmos y su importancia en programación - Diseño de algoritmos: pasos, pseudocódigo, diagramas de flujo - Análisis de complejidad de algoritmos - Ejemplos de algoritmos y su implementación en Fortran
Introducción a Fortran	- Historia y contexto de Fortran - Estructura básica de un programa Fortran - Variables y tipos de datos en Fortran - Operadores y expresiones - Entrada y salida de datos
Estructuras de Control	- Condicionales: IF, ELSE, END IF - Bucles: DO, DO WHILE, END DO - Control de flujo: GOTO y etiquetas - Ejemplos de programación con estructuras de control
Arreglos y Matrices	- Declaración y asignación de arreglos - Operaciones con arreglos - Matrices y operaciones matriciales - Ejemplos de aplicación en cálculos numéricos
Subrutinas y Funciones	- Definición y uso de subrutinas - Pasaje de argumentos - Funciones en Fortran - Modularización del código

Lectura y Escritura de Archivos	- Manejo de archivos de texto y binarios Lectura y escritura de datos desde y hacia archivos - Formato de archivos y registros - Ejemplos de aplicaciones en procesamiento de datos
Uso de LAPACK en Fortran	- Introducción a LAPACK y su importancia en cálculos numéricos. - Tipos de operaciones matriciales y sistemas de ecuaciones que LAPACK puede resolver. - Instalación y configuración de LAPACK en entornos de desarrollo de Fortran - Invocación de rutinas LAPACK desde programas Fortran. - Ejemplos prácticos de uso de LAPACK en cálculos numéricos y análisis de datos.
Introducción a la Gráficos con Gnuplot	- Creación de gráficos básicos - Personalización de gráficos: etiquetas, títulos, ejes - Tipos de gráficos: líneas, puntos, barras - Gráficos 2D y 3D - Exportación de gráficos a archivos
Introducción a la Programación en Paralelo	- Conceptos básicos de programación en paralelo - Modelos de programación paralela - Lenguajes y herramientas para programación en paralelo - Ejemplos de programación en paralelo en Fortran

BIBLIOGRAFÍA

Principal:	- Joyanes Aguilar, Luis. *Fundamentos de programación*. Quinta edición. México: McGraw-Hill, 2020. - Metcalf, Michael, Reid, John, y Cohen, Malcom. *Modern Fortran Explained: Incorporating Fortran 2018*. 5th Edition. Numerical Mathematics and Scientific Computation. United Kingdom: Oxford University Press, 2018.
Enlaces digitales:	

Complementaria:	- Curcic, Milan. *Modern Fortran: Building efficient parallel*. 1a Edición. Manning, 2020. - J. Chapman, Stephen, *Fortran for Scientists and Engineers*, Fourth Ed. New York: McGraw-Hill Education, 2018. - Chivers, Ian y Sleightholme, Jane, *Introduction to Programming with Fortran*, Fourth Edition. London: Springer International Publishing AG, 2018.

PLANEACIÓN EDUCACIONAL:

Competencias generales:	**Competencia en Programación en Fortran:** Los estudiantes adquieren habilidades sólidas en la escritura y comprensión de código en Fortran, lo que les permite desarrollar programas eficientes y confiables en este lenguaje. **Pensamiento Algorítmico:** Aprenden a pensar de manera algorítmica y a diseñar algoritmos eficientes para resolver problemas en diversos dominios. **Resolución de Problemas:** Desarrollan habilidades en la identificación, descomposición y resolución de problemas de programación, lo que les permite abordar desafíos prácticos. **Conocimiento de Estructuras de Control:** Entienden y aplican las estructuras de control en Fortran, incluyendo condicionales y bucles, para crear flujos lógicos en sus programas. **Manejo de Datos:** Aprenden a manejar datos, incluyendo entrada, salida y procesamiento de información a través de archivos y variables. **Programación Modular:** Comprenden la importancia de la programación modular y aprenden a diseñar programas organizados en módulos y subrutinas. **Optimización de Rendimiento:** Adquieren habilidades en la optimización de rendimiento, lo que les permite escribir programas eficientes y minimizar el uso de recursos. **Programación en Paralelo y Librerías Específicas:** Se desarrollan conocimientos sobre programación en paralelo y el uso de librerías específicas como LAPACK para realizar cálculos numéricos. **Habilidades de Depuración y Pruebas:** Aprenden a depurar programas y a realizar pruebas efectivas para garantizar la calidad y confiabilidad del código. **Habilidades de Comunicación Técnica:** Desarrollan habilidades para comunicar sus soluciones técnicas de manera clara y efectiva, ya sea a través de documentación, presentaciones o informes. **Pensamiento Crítico y Análisis:** Fomentan el pensamiento crítico y analítico al evaluar soluciones, identificar problemas y tomar decisiones informadas. **Autoaprendizaje:** Adquieren la capacidad de aprender de manera autodirigida, investigando y aplicando nuevas tecnologías y técnicas a medida que evolucionan.

Competencias específicas:	- **Competencia en Fortran:** Los estudiantes deberían ser capaces de escribir código en Fortran de manera eficiente, comprender la sintaxis y aplicar las mejores prácticas de codificación en este lenguaje. - **Diseño de Algoritmos en Fortran:** Aprender a diseñar algoritmos eficientes y efectivos en Fortran para resolver una variedad de problemas. - **Uso de Estructuras de Control:** Utilizar condicionales (IF, ELSE) y bucles (DO, DO WHILE) para controlar el flujode ejecución de los programas. - **Manejo de Datos en Fortran:** Aprender a manipular datos, incluyendo la entrada y salida de datos desde y hacia archivos y el procesamiento de información. - **Programación Modular:** Desarrollar la habilidad de dividir programas en módulos y subrutinas para lograr una estructura de código organizada y mantenible. - **Optimización de Rendimiento en Fortran:** Comprender cómo optimizar el rendimiento de los programas Fortran, incluyendo la minimización de recursos y la mejora de la velocidad de ejecución. - **Depuración y Pruebas en Fortran:** Adquirir habilidades para depurar programas y realizar pruebas efectivas para garantizar que el código funcione según lo previsto. - **Gestión de Memoria:** Aprender a administrar la memoria y los recursos de manera eficiente, evitando problemas como fugas de memoria. - **Uso de Librerías Externas:** Desarrollar la capacidad de integrar y utilizar librerías externas, como LAPACK, para realizar tareas específicas. - **Programación en Paralelo:** Aprender a desarrollar programas en paralelo utilizando técnicas como OpenMP o MPI. - **Resolución de Problemas Prácticos:** Aplicar los conocimientos adquiridos para resolver problemas prácticos en diversas disciplinas, como matemáticas, física o ingeniería. - **Comunicación Técnica:** Desarrollar habilidades para comunicar soluciones técnicas de manera efectiva, ya sea a través de documentación, presentaciones o informes. - **Aplicación de Conceptos Matemáticos:** Aplicar conceptos matemáticos en la resolución de problemas y cálculos numéricos utilizando Fortran. - **Autoaprendizaje Continuo:** Fomentar la capacidad de aprender de manera autodirigida, investigando y aplicando nuevas tecnologías y técnicas a medida que evolucionan.

CONTRIBUCIÓN AL PERFIL DE EGRESO:

CONOCIMIENTO:	HABILIDADES:	VALORES:
- Conceptos Generales de Algoritmos - Fundamentos de Programación - Sintaxis de Fortran - Entrada y Salida de Datos - Estructuras de Control - Manejo de Datos - Subrutinas y Funciones - Estructuras de Datos en Fortran - Depuración de Código - Buenas Prácticas de - Codificación - Optimización de Código - Desarrollo de Programas Pequeños	- Habilidades dePrograмación - Resolución de Problemas - Análisis de Datos - Simulaciones Numéricas - Optimización de Código	- Integridad - Responsabilidad - Honestidad - Servicio - Libertad - Disciplina - Capacidad

ESTRATEGIAS PEDAGÓGICAS:

Estrategias de enseñanza:	Estrategias de aprendizaje:
• **Aprendizaje Activo:** Fomentar la participación activa de los estudiantes, a través de discusiones en clase, ejercicios prácticos, resolución de problemas o proyectos. • **Clases Magistrales y Explicaciones Claras:** Proporcionar presentaciones claras de conceptos y teoría. • **Laboratorios Prácticos:** Sesiones de laboratorio donde los estudiantes puedan aplicar los conceptos teóricos en la práctica, escribir código y resolver problemas. • **Aprendizaje Basado en Proyectos:** Asignar proyectos a los estudiantes que	• **Aprendizaje Activo:** Participar activamente en la escritura de código, resolución de problemas y proyectos para aplicar conceptos en la práctica. • **Autoevaluación:** Revisar y evaluar el propio código y soluciones para identificar áreas de mejora y errores. • **Investigación Independiente:** Explorar recursos en línea, manuales de Fortran y tutoriales para aprender más allá de lo que se enseña en clase. • **Práctica Regular:** Práctica regular de programación en Fortran para fortalecer habilidades.

requieran que apliquen lo que han aprendido para resolver problemas del mundo real. • **Evaluaciones Formativas:** Realizar evaluaciones regulares a lo largo del curso para verificar la comprensión de los estudiantes y brindar retroalimentación continua. • **Enseñanza entre Pares:** Fomentar la colaboración entre estudiantes para que trabajen en grupos y se enseñen mutuamente. • **Tecnología Educativa:** Utiliza herramientas tecnológicas, como software de desarrollo, sistemas de gestión del aprendizaje (LMS) y recursos en línea para complementar la enseñanza. • **Casos de Estudio y Problemas Prácticos:** Utilizar ejemplos del mundo real y problemas prácticos para ilustrar la aplicabilidad de la programación en Fortran. • **Retroalimentación Constructiva:** Proporcionar retroalimentación detallada sobre el código de los estudiantes, destacando áreas de mejora y buenas prácticas. • **Flexibilidad y Adaptabilidad:** Adaptar las estrategias según las necesidades y el nivel de habilidad de los estudiantes. • **Desarrollo de Proyectos de Largo Plazo**: Diseñar proyectos que se extiendan a lo largo del curso, lo que permite a los estudiantes adquirir habilidades gradualmente y ver el progreso a lo largo del tiempo. • **Pruebas de Código:** Animar a los estudiantes a escribir y depurar código de manera regular, lo que es esencial en programación. • **Discusión en Grupo y Colaboración en Línea:** Utilizar foros de discusión en línea y herramientas de colaboración para que los estudiantes compartan ideas y resuelvan problemas juntos.	• **Trabajo en Grupo:** Colaborar con compañeros en proyectos y ejercicios para aprender de otros, resolver problemas de manera conjunta y compartir conocimientos. • **Lectura y Análisis de Código Fuente:** Estudiar y analizar el código fuente de programas más avanzados en Fortran para comprender cómo se escriben y se estructuran. • **Documentación y Notas:** Tomar notas y documentar el proceso de programación, incluyendo decisiones de diseño, problemas encontrados y soluciones implementadas. • **Resolución de Problemas Prácticos:** Abordar problemas prácticos relacionados con la programación en Fortran, como la optimización de código o la solución de problemas específicos. • **Autoevaluación de Rendimiento:** Evaluar regularmente el rendimiento de los programas escritos y buscar oportunidades de mejora. • **Comunicación con el Instructor:** Hacer preguntas y buscar retroalimentación del instructor cuando se enfrenten a desafíos o dificultades. • **Participación en Comunidades en Línea:** Unirse a comunidades en línea de programadores en Fortran para hacer preguntas, compartir experiencias y aprender de otros expertos. • **Experimentación:** Realizar experimentos y ajustes en programas para comprender cómo se comportan en diferentes situaciones y cómo se pueden optimizar. • **Pruebas y Depuración Autónoma:** Aprender a realizar pruebas de código y depuración de manera independiente para identificar y corregir errores. • **Aplicación en Proyectos Personales:** Aplicar lo aprendido en proyectos personales relacionados con los intereses de cada estudiante. • **Aprendizaje Continuo:** Mantenerse actualizado con las últimas tendencias y avances en Fortran y la programación en general.

PROPUESTA DE CRITERIOS DE EVALUACIÓN:

Criterio de evaluación:	Porcentaje:
Exámenes parciales Tareas Exposiciones Participación en clase Asistencia Proyecto	40-60 % 20-40 % 0-10 % 0-10 % 0 % 0-20 %