UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS	**FORMATO DE SYLLABUS**	Código: CC-FR-002	SIGUD
	Macroproceso: Direccionamiento Estratégico	Versión: 01	
	Proceso: Currículo y Calidad	Fecha de Aprobación:	

FACULTAD:	**Ciencias Matemáticas y Naturales**						
PROYECTO CURRICULAR:	**Matemáticas**				**CÓDIGO PLAN DE ESTUDIOS:**		
I. IDENTIFICACIÓN DEL ESPACIO ACADÉMICO							
NOMBRE DEL ESPACIO ACADÉMICO: Programación Científica							
Código del espacio académico:			Número de créditos académicos:			3	
Distribución horas de trabajo:	HTD	3	HTC	1		HTA	8
Tipo de espacio académico:	Asignatura	**X**	Cátedra				
NATURALEZA DEL ESPACIO ACADÉMICO:							
Obligatorio Básico	**X**	Obligatorio Complementario		Electivo Intrínseco		Electivo Extrínseco	
CARÁCTER DEL ESPACIO ACADÉMICO:							
Teórico	**X**	Práctico		Teórico-Práctico		Otros:	Cuál:_____
MODALIDAD DE OFERTA DEL ESPACIO ACADÉMICO:							
Presencial	**X**	Presencial con incorporación de TIC		Virtual		Otros:	Cuál:_____

II. SUGERENCIAS DE SABERES Y CONOCIMIENTOS PREVIOS

Para el curso de programación científica, es esencial que los estudiantes tengan conocimientos básicos en matemáticas (aritmética, álgebra y lógica matemática), estén familiarizados con el uso de computadoras y la navegación por internet, y comprendan los conceptos básicos de algoritmos y pseudocódigo. Además, se recomienda tener habilidades de lectura y comprensión de inglés técnico, así como un pensamiento lógico y analítico. Para aquellos sin experiencia previa, se sugiere tomar cursos introductorios en línea y realizar ejercicios básicos de lógica y matemáticas para fortalecer sus habilidades de razonamiento y resolución de problemas.

III. JUSTIFICACIÓN DEL ESPACIO ACADÉMICO

La programación es una habilidad esencial para cualquier matemático, en este curso se pretende proporciona una introducción sólida a los principios de la programación, la algoritmia y estructuras de datos. Se orienta en enseñar las habilidades básicas necesarias para escribir y comprender programas de computadora, así como en la implementación de algoritmos matemáticos básicos y esenciales.

IV. OBJETIVOS DEL ESPACIO ACADÉMICO (GENERAL Y ESPECÍFICOS)

Objetivo General

Proporcionar a los estudiantes de matemáticas una introducción sólida a los principios de la programación, algoritmia y estructuras de datos, con un enfoque en el desarrollo de habili-dades básicas para escribir y comprender programas de computadora, así como en la im-plementación de algoritmos matemáticos esenciales, facilitando así la resolución de pro-blemas matemáticos mediante herramientas computacionales.

Objetivos Específicos

Desarrollar habilidades en programación, incluyendo conceptos básicos, estructuras de control, funciones y recursión, estructuras de datos y algoritmos, y la implementación de algoritmos matemáticos, utilizando un lenguaje de programación relevante como Python.

Realizar un trabajo escrito, una sustentación o un proyecto que permita el desarrollo de habilidades blandas, la comunicación de ideas y la interpretación de los conceptos en diferentes contextos.

V. PROPÓSITOS DE FORMACIÓN Y DE APRENDIZAJE (PFA) DEL ESPACIO ACADÉMICO

Escribe, depura y ejecuta programas de computadora de manera efectiva, utilizando estructuras de control como condicionales y bucles para resolver problemas computacionales y diseñar algoritmos eficientes.

Comprende y aplica funciones y recursión, facilitando la modularización y simplificación de problemas complejos mediante la descomposición en subproblemas manejables, y utiliza diferentes tipos de datos y estructuras de datos (listas, pilas, colas, árboles, grafos) para implementar algoritmos de búsqueda y ordenación, optimizando el procesamiento de información.

Interpreta problemas de programación con una base sólida las técnicas de programación y estructuras de datos mediante trabajos escritos, presentaciones o proyectos en grupo, enfatizando el rigor y la precisión en la implementación y optimización de algoritmos.

Comunica de manera clara y precisa los conceptos y procedimientos de programación a través de trabajos escritos, proyectos o presentaciones, demostrando rigor y capacidad de argumentación en la exposición de ideas computacionales.

VI. CONTENIDOS TEMÁTICOS

Fundamentos de Programación: Introducción a la programación (conceptos básicos, entorno de desarrollo), Programación en Python, MatLab (u otro lenguaje relevante).
Estructuras de Control y Funciones: Estructuras de control (condicionales, bucles), Funciones y recursión.
Estructuras de Datos y Algoritmos: Tipos de datos y estructuras de datos (listas, pilas, colas, árboles, grafos), Algoritmos de búsqueda y ordenación.
Programación Avanzada y Aplicaciones Prácticas: Programación orientada a objetos, prácticas de programación con problemas matemáticos

VII. ESTRATEGIAS DE ENSEÑANZA QUE FAVORECEN EL APRENDIZAJE

Las siguientes estrategias son comunes a todos los espacios del programa académico de matemáticas. Las clases alternan entre sesiones magistrales y trabajo en grupos pequeños. En las sesiones magistrales, el profesor ejemplifica detalladamente la resolución de problemas, ejercicios y el desarrollo de la teoría. Se incorpora el uso de herramientas computacionales para presentar, explorar o interpretar propiedades de los objetos matemáticos o realizar simulaciones que refuercen el aprendizaje.

En el trabajo en los grupos pequeños se asignan problemas, temas, proyectos o ejercicios previamente estructurados por el profesor. A lo largo del proceso, el profesor lleva a cabo una evaluación formativa continua, brindando retroalimentación que facilita el avance y mejora del trabajo en grupo. Estas actividades pueden tener ciclos de cierre en cada corte académico o bien desarrollarse de manera transversal durante todo el semestre en función de las características de cada espacio académico.

En los cursos de los primeros semestres se hará énfasis en los procesos algorítmicos e intuitivos con un mayor acompañamiento del profesor y los monitores académicos; lo cual requiere que el número de estudiantes por espacio académico no sea mayor de 25 estudiantes (resolución 037, art 1 C.A, de 2022). A medida que el estudiante avanza en su carrera, se hará énfasis en el desarrollo riguroso de la teoría, así como en la autonomía del estudiante en su proceso formativo.

VIII. EVALUACIÓN

La evaluación está dividida en dos partes: pruebas escritas individuales y trabajos grupales. Los porcentajes de las pruebas pueden variar dependiendo de la naturaleza y ubicación del espacio académico en la malla curricular dentro de los siguientes parámetros.

Las pruebas escritas individuales pueden incluir quices, talleres, parciales y el examen final. En cada corte esta nota debe tener un peso del 15%-20% y en el examen final el 30%. Estas pruebas pretenden observar las habilidades del estudiante en el uso conceptual; en la resolución de ejercicios, problemas y demostraciones de teoremas.

Las pruebas grupales pueden incluir trabajos escritos, pósteres, proyectos, videos o exposiciones y deben tener un peso en cada corte del 15%-20%. Estas pruebas pretenden observar las habilidades del estudiante para trabajar en grupo, comunicar de manera escrita, oral y visual ideas matemáticas e interpretar resultados.

IX. MEDIOS Y RECURSOS EDUCATIVOS

Plataformas de Cursos en Línea:

Coursera: Ofrece cursos como "¡A Programar!" que enseña los principios fundamentales de la programación utilizando Scratch1.
edX: Cursos de introducción a la programación en varios lenguajes, incluyendo Python y Java.
Khan Academy: Recursos gratuitos sobre programación y algoritmos.

Plataformas de Práctica:
LeetCode: Ofrece una amplia variedad de problemas de programación para practicar algoritmos y estructuras de datos.
HackerRank: Proporciona desafíos de programación en varios lenguajes y niveles de dificultad.
CodeSignal: Otra plataforma excelente para practicar habilidades de programación y algoritmos.

IDEs (Entornos de Desarrollo Integrados):
PyCharm: Un IDE muy popular para Python que ofrece muchas herramientas útiles para el desarrollo.
Visual Studio Code: Un editor de código versátil que soporta múltiples lenguajes de programación y tiene una gran cantidad de extensiones.
Compiladores y Simuladores en Línea:
Repl.it: Permite escribir y ejecutar código en varios lenguajes directamente desde el navegador.
Jupyter Notebooks: Ideal para escribir y ejecutar código Python, especialmente útil para algoritmos matemáticos y análisis de datos.

X. PRÁCTICAS ACADÉMICAS - SALIDAS DE CAMPO

No aplica

XI. BIBLIOGRAFÍA

Básicas
-Matthes, Eric. Python Crash Course. No Starch Press, 2019.
-Cormen, Thomas H., et al. Introduction to Algorithms. MIT Press, 2009.

Complementarias
-Gamma, Erich, Helm, Richard, Johnson, Ralph, y Vlissides, John. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.
-Sipser, Michael. Introduction to the Theory of Computation. Cengage Learning, 2012.
-McConnell, Steve. Code Complete. Microsoft Press, 2004.

Páginas web

-YouTube: Canales como "freeCodeCamp" y "CS50" de Harvard ofrecen tutoriales y cursos completos sobre programación.
https://www.youtube.com/@freecodecamp
https://www.youtube.com/@cs50

-TED Talks: Charlas sobre la importancia de la programación y la computación en la educación y la vida diaria.
www.youtube.com/@TEDx

XII. SEGUIMIENTO Y ACTUALIZACIÓN DEL SYLLABUS

Fecha revisión por Consejo Curricular:			
Fecha aprobación por Consejo Curricular:		Número de acta:	