Campus de la UNA
SAN LORENZO-PARAGUAY

UNIVERSIDAD NACIONAL DE ASUNCIÓN
FACULTAD POLITÉCNICA
CONSEJO DIRECTIVO

RESOLUCIÓN 24/25/31-00
ACTA 1207/09/12/2024

"POR LA CUAL SE APRUEBA EL PROGRAMA DE ESTUDIO DE LA ASIGNATURA *ALGORITMOS Y ESTRUCTURAS DE DATOS 3*, DE LA CARRERA INGENIERÍA EN INFORMÁTICA – PLAN 2023 DE LA FP-UNA"

VISTO: El Memorando DA/2379/2024 del Director Académico de la FP-UNA, Prof. MSc. Felipe Santiago Uzabal Escurra, con el cual remite el Memorando CCPTCC/034/2024 de la Comisión Coordinadora del Proyecto de Transformación Curricular de Carreras de Grado de la FP-UNA, en el que presenta la propuesta de Programas de Estudio de las Asignaturas de la Carrera Ingeniería en Informática.

CONSIDERANDO: La Ley 4995/2013 de Educación Superior, el Estatuto de la Universidad Nacional de Asunción y las deliberaciones sobre el tema.

Que la Comisión Coordinadora del Proyecto de Transformación Curricular de Carreras de Grado, solicita la aprobación del Programa de Estudio de la asignatura **"Algoritmos y Estructuras de Datos 3",** de la carrera Ingeniería en Informática – Plan 2023, cuyo plan de estudio ya fue aprobado por el Consejo Superior Universitario.

EL CONSEJO DIRECTIVO DE LA FACULTAD POLITÉCNICA
RESUELVE:

24/25/31-01 APROBAR el Programa de Estudio de la Asignatura **"Algoritmos y Estructuras de Datos 3",** de la carrera Ingeniería en Informática – Plan 2023 de la FP-UNA, detallado en el ANEXO 17 de la presente Acta.

24/25/31-02 COMUNICAR, copiar y archivar.

Prof. Abg. Joel Arsenio Benítez Santacruz
Secretario

Prof. Ing. Silvia Teresa Leiva León, MSc.
Presidenta

TEL/FAX: +595 21 5887000 – C.C. 1130 (Asunción) – 2111 (San Lorenzo)
www.pol.una.py

DEPARTAMENTO DE ENSEÑANZA DE INFORMÁTICA
PROGRAMA DE ESTUDIO

I. IDENTIFICACIÓN

Asignatura	Algoritmos y Estructuras de Datos 3				
Carrera	**Plan**	**Sede/Filial**	**Carácter**	**Semestre**	**Prerrequisitos**
Ingeniería en Informática	2023	Sede San Lorenzo	Obligatoria	Tercero	Algoritmos y Estructuras de Datos 2, Lógica y Matemática Discreta
Horas semanales	4				
Total de horas teóricas semestral	36				
Total de horas prácticas semestral	36				
Total de horas semestral	72				
Valor en créditos académicos	La valoración en créditos académicos será comunicada en su oportunidad, ajustada al reglamento para la aplicación del Sistema Nacional de Créditos Académicos – Paraguay en la UNA; ajuste que se encuentra en proceso de elaboración conforme a las disposiciones de la Resolución CONES N° 221/2024, en su artículo N° 10.				
Actualización	Al egreso de la primera cohorte.				

II. FUNDAMENTACIÓN

Los algoritmos y estructura de datos son los pilares fundamentales para forman parte de los diferentes mecanismos y servicios que ofrecen hoy en día la ciencia de datos, la ingeniería de software y la inteligencia artificial, las que se encuentran cada vez más presentes en la cotidianidad.

En asignaturas previas se han introducido el pensamiento computacional, la aplicación de estructuras de datos básicas, la evaluación de rendimiento temporal y espacial de las soluciones algorítmicas a través del análisis de algoritmos y el cálculo asintótico y algunas las estrategias de diseño de algoritmos.

En esa misma línea, este curso presenta tópicos avanzados en análisis de algoritmos, estructuras de datos avanzadas, algoritmos avanzados en grafos y cadenas, técnicas de diseño para diseño de algoritmos voraces y soluciones basadas en Programación Dinámica y finalmente introduce los rudimentos necesarios para comprender cómo determinar problemas difíciles de resolver computacionalmente a través de la teoría de NP-completitud.

De esta manera el estudiante completará su bagaje de herramientas algorítmicas y de estructura de datos que permitirán ampliar su criterio de selección, aplicación y evaluación de soluciones computacionales.
Los ejes temáticos y las unidades de contenido están alineados de la siguiente forma:

- Estructuras de datos avanzadas (Tablas de dispersión, Union-Find, Tries, Árboles B): vinculadas con la Unidad 1: Estructuras de datos avanzadas, que se enfoca en estas estructuras críticas para optimizar el rendimiento de algoritmos complejos.

- Análisis amortizado y correctitud de algoritmos: abordados en la Unidad 2: Análisis de algoritmos avanzado y correctitud de algoritmos, donde se estudian técnicas avanzadas de análisis y la verificación formal de algoritmos.

- Algoritmos de búsqueda en cadenas (Fuerza Bruta, KMP, Boyer-Moore, Autómatas): directamente relacionados con la Unidad 3: Algoritmos de cadenas, que cubre estos métodos de búsqueda eficientes en secuencias de caracteres.

- Algoritmos avanzados sobre grafos (Camino mínimo, árbol de recubrimiento, redes de flujo): se desarrollan en la Unidad 4: Algoritmos avanzados en grafos, abordando problemas fundamentales en teoría de grafos.

- Programación dinámica y algoritmos voraces: tratados en la Unidad 5: Técnicas de diseño de algoritmos avanzadas, que explora estas estrategias clave en la resolución óptima de problemas complejos.

- Teoría de NP-Completitud: corresponde a la Unidad 6: NP-Completitud, que introduce la complejidad de los problemas y los límites de las técnicas algorítmicas actuales.

III. COMPETENCIAS DEL PERFIL DE EGRESO ASOCIADAS

1. Evaluar el comportamiento de diversos fenómenos disciplinares e interdisciplinares relacionados con la ingeniería en informática con una visión de sistema, mediante modelos teóricos validados y actualizados, capaces de abarcarlos integralmente en un contexto de incertidumbre.

2. Planificar, proyectar, diseñar y ejecutar proyectos sostenibles e integrales para la resolución de problemas, la mejora y la innovación en el ámbito de la ingeniería informática.

IV. ORGANIZACIÓN DE LA ASIGNATURA

Unidades	Contenidos	Resultados de Aprendizaje
1. Estructura de datos avanzadas.	.1. Implementaciones de Estructura datos en un lenguaje orientado a objetos. .1. Tablas de dispersión. Definición. Operaciones e implementaciones. .2. Árboles B y variantes. 1.1.1. Problema de ordenación y acceso a memoria secundaria (externa). 1.1.2. Algoritmos y estructuras sencillas para ordenación externa. 1.1.3. Definición de árbol B y variantes. 1.1.4. Operaciones en un árbol B. 1.1.5. Implementación de árboles B.	1. Utiliza un lenguaje orientado a objetos para implementar diversas estructuras de datos. 2. Describe las características de la estructura de datos Tablas de Dispersión, sus variantes y sus operaciones. 3. Utiliza la estructura de datos Tablas de Dispersión a problemas algorítmicos que requieren búsqueda. 4. Utiliza el análisis de algoritmos para determinar el rendimiento temporal y espacial de una Tabla de Dispersión. 5. Describe la problemática de ordenación y acceso eficiente a memoria secundaria. 6. Describe los algoritmos y las estructuras simples para resolver

Unidades	Contenidos	Resultados de Aprendizaje
		el problema de ordenación en memoria secundaria. 7. Describe las características de la estructura de datos de árboles B, sus variantes y sus operaciones. 8. Utiliza la estructura de datos árboles B a problemas algorítmicos que requieren búsqueda. 9. Utiliza el análisis de algoritmos para determinar el rendimiento temporal y espacial de un árbol B. 10. Desarrolla soluciones algorítmicas utilizando las estructuras de datos avanzadas adecuadas a los problemas algorítmicos planteados.
2. Análisis de algoritmos avanzado y correctitud de algoritmos.	2.1. Revisión del análisis de algoritmos. 2.2. Análisis amortizado. 2.3. Correctitud de algoritmos. 2.3.1.Introducción. 2.3.2.Inducción. 2.3.3.Contradicción. 2.3.4.Lazo invariante.	1. Utiliza el análisis de algoritmos para determinar correctamente el rendimiento espacial y temporal de algoritmos secuenciales y recursivos. 2. Describe los escenarios de aplicación del análisis amortizado. 3. Identifica el uso del análisis amortizado en situaciones algorítmicas específicas. 4. Utiliza el análisis amortizado en problemas algorítmicos y estructura de datos. 5. Describe la importancia de las técnicas para determinar la correctitud de los algoritmos. 6. Describe la técnica del lazo invariante aplicado para determinar la correctitud de los algoritmos. 7. Describe la técnica de inducción para determinar la correctitud de algoritmos. 8. Describe la técnica del razonamiento por contradicción para determinar la correctitud de los algoritmos. 9. Utiliza las técnicas apropiadas para determinar la correctitud de los algoritmos.
3. Algoritmos de cadenas	3.1.Problema de búsqueda en cadenas. 3.2.Algoritmos de búsqueda en cadenas:	1. Describe la problemática de la búsqueda de cadenas y su terminología. 2. Describe los algoritmos de Boyer-

Unidades	Contenidos	Resultados de Aprendizaje
	3.2.1. Boyer-Moore 3.2.2. Autómatas finitos 3.2.3. KMP 3.2.4. String Hashing (Rabin-Karp) 3.3.Árboles de sufijos. Definición. Operaciones e implementaciones.	Moore, autómatas finitos, KMP y Rabin-Karp para la búsqueda de cadenas. 3. Utiliza los algoritmos de búsqueda de cadenas en problemas algorítmicos 4. Utiliza el análisis de algoritmos para determinar el rendimiento espacial y temporal de los algoritmos de búsqueda de cadenas. 5. Desarrolla soluciones utilizando los algoritmos de búsqueda de cadenas y sus variantes en problemas algorítmicos. 6. Determina la correctitud de los algoritmos de búsqueda de cadenas utilizando las técnicas apropiadas desarrolladas. 7. Describe los árboles de sufijos y su utilización en el problema de búsqueda parcial de cadenas. 8. Utiliza el árbol de sufijo en problemas algorítmicos. 9. Desarrolla soluciones utilizando árboles de sufijos en problemas algorítmicos. 10. Utiliza el análisis de algoritmos para determinar el rendimiento espacial y temporal en algoritmos basados en árboles de sufijos.
4. Algoritmos avanzados en grafos	4.1. Definición y terminología formal de Grafos. 4.2. Algoritmos. 4.2.1. Camino mínimo. 4.2.1.1. Definición. 4.2.1.2.De una fuente (Dijkstra y Belland-Ford). 4.2.1.3.De todas las fuentes (Floyd-Warshall). 4.2.2. Árbol de Recubrimiento mínimo. 4.2.2.1. Definición del problema. 4.2.2.2. UNION-FIND. Operaciones e implementaciones. 4.2.2.3. Algoritmo de Prim 4.2.2.4. Algoritmo de Kruskal 4.2.3. Redes de flujo 4.2.3.1. Definición 4.2.3.2. MinCut Max-Flow 4.2.3.3. Algoritmo de Ford-Fulkerson	1. Describe la definición y la terminología formal en grafos. 2. Describe el problema del camino mínimo en un grafo con pesos positivos y negativos y sus utilidades. 3. Describe los algoritmos en grafos con pesos para solucionar el problema de camino mínimo de una o varias fuentes con pesos positivos o negativos. 4. Utiliza los algoritmos para encontrar el camino mínimo de una o varias fuentes en problemas algorítmicos. 5. Describe el problema de encontrar el árbol de recubrimiento mínimo en un grafo con pesos y sus utilidades. 6. Describe los algoritmos en grafos con pesos para solucionar el

Unidades	Contenidos	Resultados de Aprendizaje
	4.2.3.4. Aplicaciones	problema de encontrar el árbol de recubrimiento mínimo. 7. Describe la estructura UNION-FIND y sus implementaciones utilizadas en varios algoritmos en grafos. 8. Utiliza los algoritmos para encontrar el árbol de recubrimiento mínimo en problemas algorítmicos. 9. Describe el problema de redes de flujo en grafos con pesos y sus aplicaciones. 10. Describe los algoritmos de redes de flujo en grafos con pesos. 11. Utiliza los algoritmos de redes de flujo en grafos con pesos en problemas algorítmicos. 12. Utiliza el análisis de algoritmos para determinar el rendimiento espacial y temporal de los algoritmos en grafos. 13. Utiliza el análisis adecuado para determinar la correctitud de los algoritmos aplicados en grafos con pesos. 14. Desarrolla soluciones utilizando los algoritmos o variantes basadas en grafo con pesos en problemas algorítmicos.
5. Técnicas de diseño de algoritmos avanzadas.	5.1. Revisión de técnicas de diseño de algoritmos. 5.2. Algoritmos voraces 5.2.1. Definición y aplicación. 5.2.2. Algoritmos clásicos resueltos con la técnica. 5.3. Programación Dinámica. 5.3.1. Definición y aplicación. 5.3.2. Problemas clásicos resueltos con la técnica.	1. Describe la importancia de las técnicas de diseño de algoritmos. 2. Describe las características y uso de la técnica de solución con algoritmos voraces. 3. Describe las características y uso de la técnica de Programación Dinámica. 4. Utiliza la técnica de diseño aplicando algoritmos voraces a problemas algorítmicos clásicos. 5. Utiliza la técnica de diseño basado en programación dinámica a problemas algorítmicos clásicos. 6. Desarrolla soluciones algorítmicas aplicando la técnica uso de algoritmos voraces en problemas computacionales. 7. Desarrolla soluciones algorítmicas aplicando la técnica de Programación Dinámica en problemas computacionales.

Unidades	Contenidos	Resultados de Aprendizaje
		8. Utiliza el análisis de algoritmos para evaluar el rendimiento espacial y temporal de los algoritmos desarrollados con algoritmos voraces y programación dinámica. 9. Utiliza el análisis adecuado para determinar la correctitud de las soluciones con algoritmos voraces y con programación dinámica.
6. NP-Completitud.	6.1. Introducción. 6.2. Problemas de decisión. 6.3. Clase de Problemas P y NP. 6.4. NP-Completitud y reducibilidad. 6.5. Problemas NP-Completos clásicos. 6.5.1. Clique. 6.5.2. Cobertura de vértices. 6.5.3. Ciclo Hamiltoniano. 6.5.4. TSP.	1. Valora la importancia de conocer los fundamentos de la complejidad computacional. 2. Enumera las clases de problemas computacionales. 3. Clasifica los diferentes problemas computacionales de acuerdo con su complejidad computacional. 4. Definir el concepto de problema de decisión. 5. Describe las características de la clase de complejidad P y NP. 6. Describe las características de la clase de NP-Completo y su importancia. 7. Describe el proceso de reducción de problemas en el contexto de demostrar el grado de dificultad de un problema computacional. 8. Describe los problemas NP-Completos clásicos. 9. Describe la demostración para determinar si un problema es NP-Completo. 10. Utiliza el proceso de reducción para determinar el grado de complejidad de un problema computacional y su relación con otro problema computacional.

V. ESTRATEGIAS DIDÁCTICAS

En el desarrollo del programa se aplicarán estrategias didácticas conducentes a la apropiación teórica y la ejecución práctica de procesos y procedimientos, a saber:

- **Aprendizaje basado en problemas:** exposición por parte del docente de los conceptos básicos por unidad, con materiales de lectura y ejemplos orientados a la enseñanza de las competencias específicas de la asignatura. El estudiante buscará resolver un problema a través del conocimiento que adquirió en el aula.

- **Aprendizaje basado en proyectos:** el docente propondrá la realización de un proyecto que involucre todos los resultados de aprendizaje de la materia. De esta forma el estudiante

participa activamente en su aprendizaje, desarrollando diferentes habilidades para solucionar un problema a través de este proyecto.

La elección particular de la estrategia didáctica aplicada será explícita en el plan de clases, de acuerdo con el perfil de los estudiantes, los recursos disponibles y el contexto educativo.

VI. ESTRATEGIAS EVALUATIVAS

Procesos de producción grupales e individuales, pruebas individuales escritas o de forma de cuestionario durante el desarrollo de las unidades, implementación de las ideas de los algoritmos en un lenguaje de programación orientado a objetos como tareas entregables. Todos estos serán valorados y que en su conjunto aportarán para la calificación y promoción, las que serán aplicadas según normativas institucionales.

Con fines de calificación y promoción se aplicará la normativa sobre evaluación vigente en la institución que prevé valoraciones de proceso y final.

VII. MEDIOS AUXILIARES

Aula virtual, pizarrón, proyector, marcadores, equipo de audio, acceso a Sala de Máquinas.

VIII. BIBLIOGRAFÍA

- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to algorithms. MIT press.
- Dasgupta, S., Papadimitriou, C. H., & Algorithms, U. V. (2006). McGraw-Hill Science. Engineering/Math.
- Weiss, M. A. (2013). Estructuras de datos en Java. 4ta. Edición. Pearson.
- Clifford, Shaffer (2013). Data Structures and Algorithm Analysis. Java versión. Accesible en http://people.cs.vt.edu/~shaffer/Book/JAVA3elatest.pdf.
- Guerequeta, R.; Vallecillo, A. (1998). Técnicas de Diseño de Algoritmos. Servicio de Publicaciones de la Universidad de Málaga. Accesible en: http://www.lcc.uma.es/~av/Libro/indice.html
- Joyanes A., Luis ; Zahonero Martínez, Ignacio. (2011). Programación en Java 6. Algoritmos y Programación Orientada a Objetos.
- Brassard, G.; Bratley, P. (1996) Fundamentos de Algoritmia. Prentice Hall.
- Erickson, J. (2019) Algorithms. Accesible en: https://jeffe.cs.illinois.edu/teaching/algorithms
- Steven S. Skiena (2020). The Algorithm Design Manual. 3ra Edicion. Springer.