Universidad Metropolitana

PROGRAMA DE ESTRUCTURAS DE DATOS

Facultad:	Ingeniería	**Departamento:**	Gestión de Proyectos y Sistemas
Código:	BPTSP03	**Asignatura:**	Estructuras de Datos
Créditos:	3	**Tipo:**	X Obligatoria ___ Electiva
Carreras:	Ingeniería de Sistemas (IS). Matemáticas Industriales (MI)	**Trimestres:**	V (MI) IV (IS)
Prerrequisito	Algoritmos y Programación BPTSP05	**Modalidad:**	Presencial

Número de horas semanales			
En aula	**Prácticas supervisadas**	**Laboratorio**	**Aprendizaje Autónomo**
4			4
Coordinador:	Christian Guillén Drija	**Fecha de actualización**	Sep. 2025

1. ***Justificación:*** El estudio de las estructuras de datos es fundamental para el diseño de soluciones algorítmicas eficientes en función de las restricciones impuestas por un contexto en el que se requiere la automatización de procesos. En tal sentido, resulta imperativo para un ingeniero de sistemas la obtención de conocimientos sobre los distintos tipos de datos abstractos, así como el desarrollo de destrezas que le permitan determinar la aplicabilidad de las distintas estructuras de datos en el diseño de componentes de software con características de calidad tales como eficiencia de desempeño, adaptabilidad y seguridad.

2. ***Propósito:*** Este curso busca fortalecer las capacidades técnicas y analíticas necesarias para modelar eventos, objetos y procesos del mundo real mediante el uso de estructuras de datos, considerando principios como la eficiencia computacional, la adaptabilidad y la seguridad. Además, fomenta un enfoque ingenieril que integra conocimientos teóricos con aplicaciones prácticas, preparando a los estudiantes para enfrentar desafíos en el diseño y desarrollo de soluciones algorítmicas dentro del marco profesional.

3. ***Objetivos***

- Evaluar la eficiencia de diferentes algoritmos para la resolución de problemas, considerando factores como la complejidad temporal y espacial. (Objetivos 1 y 2 de la carrera)
- Diseñar y desarrollar soluciones de software utilizando el paradigma de programación orientada a objetos, considerando principios de diseño y buenas prácticas.
- Seleccionar la estructura de datos dinámica más adecuada para resolver un problema específico, justificando la elección.
- Implementar diferentes estructuras de datos dinámicas utilizando el paradigma de programación orientada a objetos. (Objetivo 1 de la carrera)
- Aplicar la recursividad en la construcción de algoritmos para resolver problemas de diversa complejidad. (Objetivo 1 de la carrera)

4. ***Resultados de aprendizaje***

a) **RA8 - Resolución de problemas de ingeniería.** Capacidad para comprender, definir y resolver problemas de análisis de ingeniería en el campo de estudio pertinente, con el uso de conocimientos básicos y avanzados de métodos analíticos modernos. (N1)

5. ***Contenido***

Tema	Contenido	Herramientas técnicas y actividades (proyectos, trabajos, laboratorios)	Horas dedicadas
1	Estructuras de datos lineales: Listas simples, listas dobles. Listas circulares, pilas y colas, listas multienlazadas.	Visualización mediante animaciones y simulaciones de operaciones (inserción/borrado). Implementación en POO. Análisis comparativo de eficiencia y memoria.	16
2	Análisis de algoritmos: Orden de crecimiento. Notación O. Análisis de ejecución de ciclos	Ejercicios de contextualización. Simulaciones y demostraciones del comportamiento algorítmico según complejidad temporal (Big O).	4
3	Árboles y Grafos: Recorridos de árboles, Árboles de búsqueda, Árboles Binarios de Búsqueda, Árboles Balanceados (B-trees), Árboles AVL, Grafos, Implementación de grafos.	Animaciones de recorridos (Preorden, Inorden, Postorden). Resolución de ejercicios de visita de nodos. Proyecto: Selección de estructura óptima para casos hipotéticos.	16

4	Tablas de dispersión. Implementación. Función Hash. Técnicas para el manejo de colisiones.	Investigación independiente guiada. Proyecto colaborativo: construcción de un sistema complejo utilizando tablas de dispersión para optimización.	4

6. *Métodos de aprendizaje:*

La metodología de enseñanza para la asignatura se centrará en un enfoque mixto que combine elementos de enseñanza tradicional con estrategias pedagógicas activas, con el objetivo de promover un aprendizaje significativo y profundo en los estudiantes. Los componentes metodológicos son: **Aprendizaje activo:** Se fomenta el aprendizaje de los estudiantes mediante discusiones, debates y resolución de ejercicios prácticos en el aula. **Aprendizaje Basado en Problemas (ABP), Aprendizaje Cooperativo, Aprendizaje Basado en Proyectos:** Estos se pueden implementar con estas estrategias instruccionales:

- Demostración del comportamiento de los distintos Tipos de Datos Abstractos (TDAs) a través de simuladores en línea o elaborados por el profesor, como un paso preparatorio a la caracterización de cada una de ellas.
- Desarrollo de ejercicios en los que los estudiantes implementen las primitivas de las distintas TDAs en un lenguaje de programación orientado a objetos.
- Análisis comparativo de implementaciones de las TDAs a través de apuntadores (explícitos e implícitos) y arreglos.
- Talleres reflexivos a partir de la revisión de los proyectos realizados por sus pares, considerando los siguientes criterios: complejidad en espacio, complejidad en tiempo, correcta aplicación de los principios de la programación orientada a objetos.
- Resolución de problemas guiados, colocando énfasis en la conveniencia de utilizar las primitivas de cada una de las TDAs estudiadas y la consideración de las complejidades temporales subyacentes.
- Desarrollo de proyectos, propiciando el trabajo en equipo para el diseño de soluciones a problemas complejos en los que se requiera adecuación de los TDAs estudiados a requerimientos específicos, así como el estudio y aplicación de variantes no vistas en clases.

7. *Métodos de evaluación:*

Aprendizaje en contacto con el docente (60%)	Aprendizaje práctico experimental (30%)	Aprendizaje autónomo (10%)
Exposiciones, Participación en clases, Debates, Exámenes escritos u orales, Talleres, Defensa de proyectos, entre otros.	Resolución de problemas prácticos, Prácticas de laboratorio, Salidas de campo o visitas técnicas, Manejo de software especializado, Prototipado técnico, Estudios de caso técnicos, entre otros.	Elaboración de informes, Resolución de problemas y ejercicios, Ensayos de investigación, Creación de mapas conceptuales, Participación en foros, entre otros.

8. *Referencias*

Obligatoria:

- Barnett, G., y Del Tongo, L. (2008). *Data structures and algorithms: Annotated reference with examples*.
- Chawdhuri, D. (2017). *Java 9 data structures and algorithms*. Packt Publishing.
- Deitel, H., y Deitel, P. (2004). *Java: Cómo programar*. Prentice Hall.
- Goodrich, M., Tamassia, R., y Goldwasser, M. (2013). *Data structures and algorithms in Python*. John Wiley & Sons.
- Joyanes, L., y Zahonero, I. (2002). *Programación en Java 2: Algoritmos, estructura de datos y programación orientada a objetos*. McGraw-Hill.
- Joyanes, L. (2003). *Fundamentos de programación I: Algoritmos, estructura de datos y objetos*. McGraw-Hill.
- Lewis, J., y Chase, J. (2006). *Estructuras de datos con Java: Diseño de estructuras y algoritmos* (2.ª ed.). Addison Wesley.
- ***Adicional:***
- Liang, Y. (2019). *Java programming and data structures: Comprehensive version*.
- Shaffer, C. (2011). *Data structures and algorithm analysis (C++ version)* (Ed. 3.2). Dover Publications.
- Waite, M., y Lafore, R. (2005a). *Data structures and algorithms*. Waite Group Press.
- Waite, M., y Lafore, R. (2005b). *Object-oriented design in Java*. Waite Group Press.
- Weiss, M. (2004). *Estructuras de datos en Java*. Addison Wesley.