

M2 Estructuras de datos FIN A

ACTIVIDAD 2

| **Tutor:** | **Luis Sierra Betancourt** |
| --- | --- |
| **Estudiante:** | **José Ramón Ibáñez Posadas** |
| **Matricula:** | **BNL098377** |

| Monterrey, Nuevo León | Lunes, 12 de Febrero de 2024 |
| --- | --- |

INTRODUCCIÓN

En el vasto mundo de la programación, la eficiente manipulación y organización de datos son esenciales para el desarrollo de software robusto y funcional. En este contexto, los tipos de datos abstractos (TDAs) se destacan como herramientas fundamentales que permiten a los programadores trabajar con conjuntos de datos de manera estructurada y eficaz. Desde conjuntos y árboles hasta pilas y grafos, los TDAs ofrecen una abstracción poderosa que simplifica la manipulación de datos y facilita la implementación de algoritmos complejos. Exploraremos cómo estos conceptos juegan un papel crucial en la programación moderna y cómo su comprensión puede potenciar el desarrollo de software en una variedad de aplicaciones.

.

DESARROLLO

REPORTE

En el ámbito de la programación, se encuentran diversos lenguajes de programación que han surgido y evolucionado a lo largo del tiempo, adaptándose continuamente a las necesidades de los programadores.

Un tipo de dato abstracto (TDA) se define como un modelo matemático que engloba un conjunto de operaciones definidas sobre un conjunto de datos, o también como un tipo definido por el usuario que incluye un conjunto de valores y operaciones. Este concepto proporciona al programador objetos con operaciones básicas independientes de la implementación específica, caracterizándose por su encapsulamiento, protección, representación abstracta y capacidad de compilación independiente.

Los TDA se distinguen por un conjunto de operaciones conocido como interfaz pública, que representa su comportamiento, mientras que la implementación permanece oculta como parte privada del software que lo utiliza. Los lenguajes de alto nivel suelen incluir TDAs predefinidos, como tipos simples y estructuras, con una interfaz general que especifica ciertas operaciones.

La implementación de los operadores y elementos internos de un TDA se oculta del usuario, permitiendo una implementación flexible y adaptada a las necesidades del programa. Los TDAs se organizan de manera estructurada para traducir comportamientos específicos y pueden clasificarse en políticos, que dependen del comportamiento estructural, y monolíticos, que admiten un único valor en cada momento.

La abstracción de datos es la función principal de los TDAs, permitiendo ocultar y simplificar las propiedades de un objeto para utilizar únicamente su nombre en el programa. Algunos ejemplos de TDAs incluyen conjuntos, árboles binarios de búsqueda, pilas, colas y grafos, cada uno implementando operaciones básicas y complejas para el manejo eficiente de datos en programas informáticos.

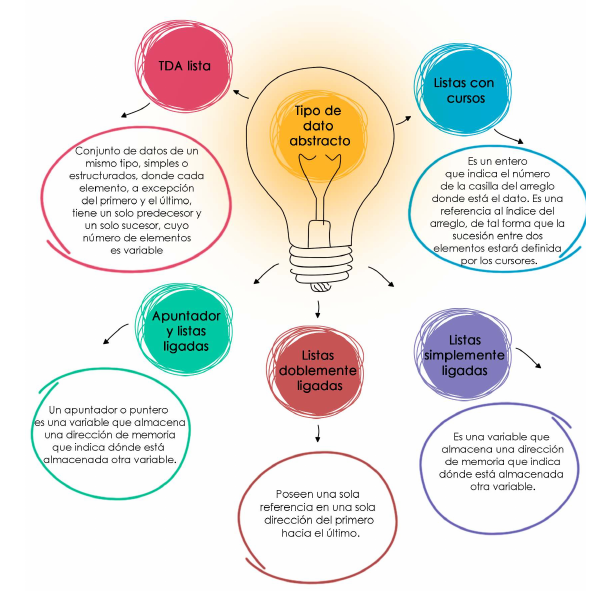
Por ejemplo, los conjuntos se utilizan para representar conjuntos de elementos y ofrecen operaciones como unión, intersección y diferencia, así como inserción, borrado y búsqueda de elementos. Los árboles binarios de búsqueda son estructuras utilizadas para organizar datos de manera jerárquica, facilitando la búsqueda eficiente de elementos. Aunque se consideran como TDAs separados, también forman parte de la familia de los grafos debido a su naturaleza de conexiones entre nodos.

Las pilas y colas son estructuras de datos fundamentales que implementan algoritmos de acceso a datos conocidos como FIFO (First In, First Out) y LIFO (Last In, First Out), respectivamente. Estas estructuras son esenciales en aplicaciones donde se requiere un orden específico en la manipulación de datos, como la gestión de tareas o la navegación en un historial.

Por último, los grafos representan conjuntos de vértices conectados entre sí mediante arcos o aristas. Estas estructuras son utilizadas para modelar una amplia gama de relaciones entre objetos, desde redes sociales hasta rutas de transporte. La implementación de grafos como un TDA permite operaciones como la búsqueda de caminos más cortos, la identificación de ciclos y la determinación de conectividad entre nodos.

En resumen, los TDAs son herramientas fundamentales en la programación que permiten abstraer y organizar datos de manera eficiente, ocultando detalles de implementación y proporcionando una interfaz clara y consistente para el manejo de datos en diferentes contextos de aplicación. Su versatilidad y capacidad para adaptarse a diversas necesidades hacen que sean una parte esencial del desarrollo de software moderno.

MAPA MENTAL



CONCLUSIÓN

Los tipos de datos abstractos (TDAs) representan una piedra angular en el desarrollo de software, proporcionando una abstracción poderosa que simplifica la manipulación de datos y la implementación de algoritmos. Desde conjuntos simples hasta estructuras complejas como grafos, los TDAs ofrecen una manera estructurada y eficiente de organizar información en aplicaciones informáticas. Su capacidad para ocultar detalles de implementación y ofrecer una interfaz clara y consistente hace que sean herramientas esenciales para los programadores en la creación de software robusto y funcional. Al comprender y utilizar eficazmente los TDAs, los desarrolladores pueden mejorar significativamente la calidad y la eficiencia de sus programas, abriendo la puerta a nuevas posibilidades en el mundo de la programación y la informática.

BIBLIOGRAFÍA

Universidad CNCI. (2024). TDA lista. Obtenido de Blackboard Universidad CNCI: <https://cnci.blackboard.com/bbcswebdav/courses/BbCont10/Contenido/2Test/I_EstDatos_Claudia_DEMOMAR20/clase_06/clase_06.html>

Universidad CNCI. (2024). Listas con cursores. Obtenido de Blackboard Universidad CNCI: <https://cnci.blackboard.com/bbcswebdav/courses/BbCont10/Contenido/2Test/I_EstDatos_Claudia_DEMOMAR20/clase_07/clase_07.html>

UNED (2024). Revisado el 12 de Febrero de 2024, Obtenido de: <https://www6.uniovi.es/usr/cesar/Uned/EDA/Apuntes/ED1TEMA1.pdf>