Tecnológico de Monterrey

# Programación de estructuras de datos y algoritmos fundamentales

# TC1031.632

## Act 2.4 - Actividad Integral estructura de datos lineales (Evidencia Competencia)


| Nombre(s) | Apellidos | Matricula |
|---|---|---|
| Amy Vanesa | Díaz García | A01656419 |
| Olivia Bianelli | Carrasco Navarrete | A01656898 |
| Daniel | Nava Mondragón | A01661436 |

**Profesor: David Alejandro Escárcega Centeno**


**Octubre 2022**

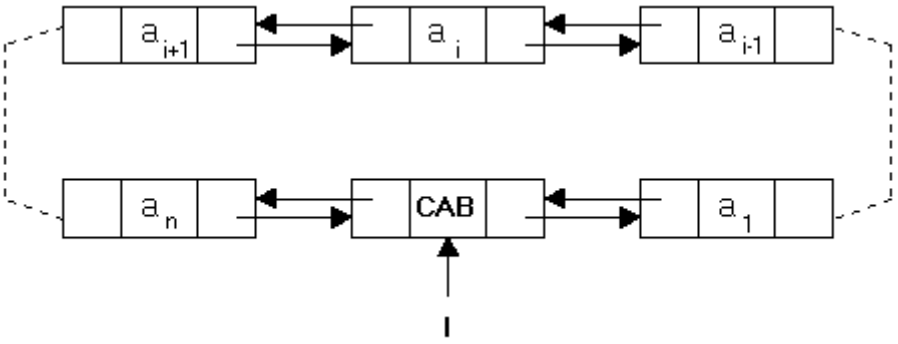
## Listas doblemente ligadas

Una lista doblemente ligada es aquella que cuenta con una lista lineal donde cada nodo se comprende de dos enlaces. El primer enlace al nodo previo y el otro al nodo siguiente.

Las listas doblemente ligadas tienen una gran eficiencia cuando se trata de ordenar un número indeterminado de datos.

Una de las ventajas más destacables de las listas doblemente ligadas es que no necesitan de un nodo en particular, por lo que pueden recorrer en dirección hacia adelante y hacia atrás, lo que hace que la eliminación de nodos sea más fácil, de igual manera, la inserción de datos se hace más eficiente.

Usualmente, se utilizan las listas doblemente ligadas para la navegación de páginas web en ambos sentidos tanto adelante como atrás, de igual manera se puede utilizar para diferentes estructuras de datos.




## *Stack*

Las pilas son un tipo de almacenamiento que tiene un funcionamiento de tipo LIFO (*Last In First Out*). En estas se puede agregar y eliminar datos del extremo superior. Las *Stacks* funciona por un objeto encapsulado, ya sea de *deque*, de un vector o lista, el cual usa como un contenedor subyacente, el cual se encarga de proveer un grupo particular de funciones miembro para entrar a sus elementos.

Las funciones principales de *Stack* son:

- Empty()

  Función booleana que regresa un verdadero si el stack está vacío o un falso en caso de que no. **Complejidad:** O(1)

- Size()

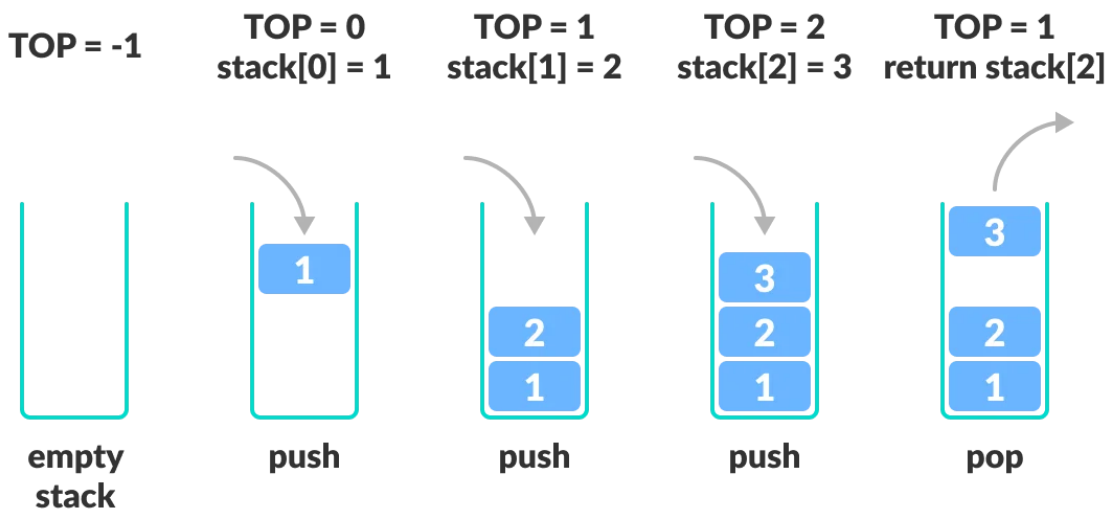
  Esta función regresa el tamaño de la pila. **Complejidad:** O(1)

- Push(n)

  Agrega el valor 'n' en la parte superior del *stack.* **Complejidad:** O(1)

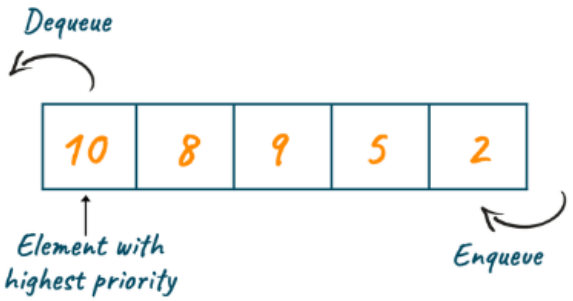
- Pop()

  Este se encarga de eliminar el valor más alto del *Stack*. **Complejidad:** O(1)



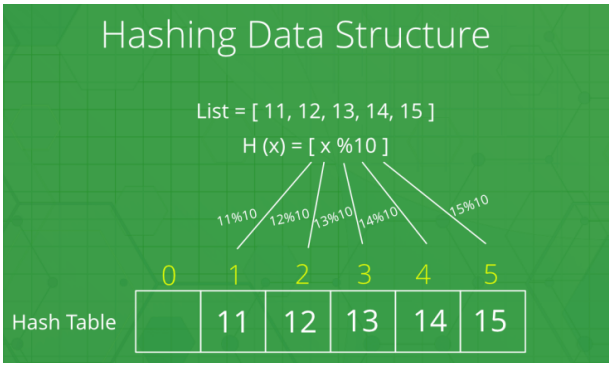
## Priority List

Las listas prioritarias son contenedores al igual que *stack*, sin embargo, estas fueron creadas para que el primer valor de la lista ya esté ordenado, sea el menor o mayor de todos los demás valores que se encuentran en la cola. Este ordenamiento no es de manera creciente, por lo que tenemos la posibilidad de observar que cada uno de los valores cuentan con una prioridad de {*fixed order*}. La estructura interna de estas son los vectores o las matrices.




## Hash Table

Dentro del 'hashing' existe una función que se encarga de designar claves a ciertos datos dentro de la tabla hash. La finalidad de esta función es que se tenga un acceso más ágil a los valores. La eficacia de este mapeo depende de la operatividad de la tabla hash empleada. El problema de esta función, es que se pueden generar colisiones, lo anterior significa que pueden encontrarse dos o más claves que designan el mismo dato.



**Reflexiones:**

Por medio de esta actividad pudimos aplicar de manera práctica los conocimientos aprendidos a lo largo de estas cinco semanas. Gracias a las *stacks, priority queues* y *hashtables* fue que pudimos desarrollar esta actividad viendo los usos y eficacia de cada uno en diferentes casos.

**Referencias:**


*C Con Clase | Estructuras de datos (cap5)*. (s. f.). Recuperado 7 de octubre de 2022, de https://conclase.net/c/edd/cap5.

GeeksforGeeks. (2022, 14 junio). *Applications, Advantages and Disadvantages of Doubly Linked List*. Recuperado 7 de octubre de 2022, de https://www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-doubly-linked-list/

GeeksforGeeks. (2022, 23 agosto). *Stack in C++ STL*. Recuperado 23 de octubre de 2022, de https://www.geeksforgeeks.org/stack-in-cpp-stl/

GeeksforGeeks. (2022a, julio 8). *Priority Queue in C++ Standard Template Library (STL)*. Recuperado 23 de octubre de 2022, de https://www.geeksforgeeks.org/priority-queue-in-cpp-stl/

GeeksforGeeks. (2022d, agosto 27). *C++ program for hashing with chaining*. Recuperado 23 de octubre de 2022, de https://www.geeksforgeeks.org/c-program-hashing-chaining/

GeeksforGeeks. (s. f.). *Hashing Data Structure*. Recuperado 23 de octubre de 2022, de https://www.geeksforgeeks.org/hashing-data-structure/