

Listas

Emmanuel Cruz Hernández
`emmanuel_cruzh@ciencias.unam.mx`

22 de octubre de 2020

Contenido

- 1 Introducción
- 2 Listas con Arreglos
- 3 Listas ligadas
- 4 Listas doblemente ligadas
- 5 Bibliografía

Una lista es una estructura que almacena datos en forma ordenada, uno después de otro.

Es una estructura que cuenta con las siguientes características:

- Dinámica
- Homogénea
- Lineales
- Directo o secuencial, dependiendo de la implementación

Formas de implementación

Algunas de las implementaciones para listas son las siguientes:

- Listas con Arreglos
- Listas basadas en referencias:
 - Listas ligadas
 - Listas doblemente ligadas

Cada una cuenta con características particulares. La diferencia principal entre cada implementación está en la complejidad en tiempo y espacio, sin embargo, las operaciones que puede realizar una lista son las mismas sin importar la implementación.

Operaciones sobre listas

- `longitud()`: Regresa el número de elementos contenidos en una lista.
- `esVacia()`: Regresa un booleano, *true* si la lista no contiene elementos y *false* en caso contrario.
- `obtenElemento(i)`: Regresa el elemento en la posición *i* de la lista. Ocurre un error si *i* no está en el rango $[0, \text{longitud}()-1]$.
- `estaContenido(e)`: Verifica si un elemento está en el contenido de una lista. Regresa *true* si está contenido, *false* en otro caso.

Operaciones sobre listas

- $\text{agrega}(i, e)$: Inserta un nuevo elemento e en la lista para que tenga el índice i , moviendo todos los elementos subsiguientes un índice más adelante en la lista; se produce una condición de error si i no está en el rango $[0, \text{longitud}())$.
- $\text{elimina}(i)$: Elimina y devuelve el elemento en el índice i , moviendo todos los elementos posteriores un índice antes en la lista; se produce una condición de error si i no está en el rango $[0, \text{longitud}()-1]$.

Listas con Arreglos

Esta implementación tiene como base un arreglo para almacenar los datos o elementos que formarán parte de la lista.

Representación gráfica



Características de un arreglo

- Cuando se inicializa un arreglo, se le da un tamaño fijo.
- Tiene acceso directo a sus elementos, es decir, no es necesario recorrer la estructura para acceder a uno de sus elementos.

Operaciones con listas con arreglos

Se pueden consultar las operaciones de forma visual, implementadas con listas con arreglos en el siguiente enlace:

https://docs.google.com/presentation/d/10hrwBQ63_peBhWhNG3PU01w00r3t2pEoLKFufFGQFP4/edit?usp=sharing

Complejidad en tiempo

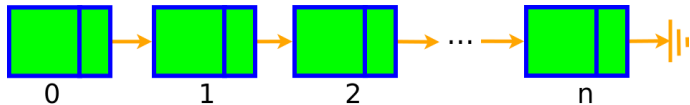
Método	Complejidad
longitud()	$O(1)$
esVacia()	$O(1)$
obtenElemento(i)	$O(1)$
estaContenido(e)	$O(n)$
agrega(i, e)	$O(n)$
elimina(i)	$O(n)$

La implementación de esta lista se basa en referencias. Existe un contenedor, el cuál se conoce como nodo, que contiene la referencia a los nodos siguientes.

Un nodo es un objeto que sirve como contenedor. Un nodo se conforma por dos elementos importantes:

- El elemento almacenado en el nodo.
- Una referencia a un nodo siguiente.

Representación gráfica



Operaciones con listas ligadas

Se pueden consultar las operaciones de forma visual, implementadas con listas ligadas en el siguiente enlace:

<https://docs.google.com/presentation/d/1T0DBW4FUK4EA5w98H8c0ZerQjavJfoULE9G1U5dnrYE/edit?usp=sharing>

Complejidad en tiempo

Método	Complejidad
longitud()	$O(1)$
esVacia()	$O(1)$
obtenElemento(i)	$O(n)$
estaContenido(e)	$O(n)$
agrega(i, e)	$O(n)$
elimina(i)	$O(n)$

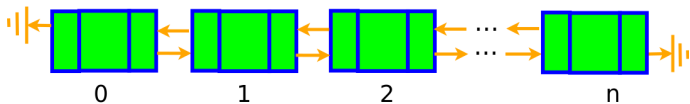
Listas doblemente ligadas

Al igual que en las listas ligadas o simplemente ligadas, La implementación de esta lista se basa en referencias. La diferencia está en la cantidad de referencias a las que está asociado un nodo.

En estas listas, un nodo se conforma por tres elementos importantes:

- El elemento almacenado en el nodo.
- Una referencia al nodo anterior.
- Una referencia a un nodo siguiente.

Representación gráfica



Operaciones con listas doblemente ligadas

Se pueden consultar las operaciones de forma visual, implementadas con listas doblemente ligadas en el siguiente enlace:

<https://docs.google.com/presentation/d/1mpoq-VlaSpKtJ0m672VGSpoxoV2tYq6CiMBXwoXx6LbM/edit?usp=sharing>

Complejidad en tiempo

Método	Complejidad
longitud()	$O(1)$
esVacia()	$O(1)$
obtenElemento(i)	$O(n)$
estaContenido(e)	$O(n)$
agrega(i, e)	$O(n)$
elimina(i)	$O(n)$



GOODRICH, M.T., TAMASSIA, R. Y GOLDWASSER, M.H., *Data Structures and Algorithms in Java*, Wiley, Sexta Edición, 2014.



WEISS, M. A., *Data Structures and Algorithm Analysis in Java*, Pearson, Tercera Edición, 2012.