

Programación en Python y C++

Estructuras de Datos

Rafael De Luna Loredo

Facultad de Ciencias, UASLP



Indice

① Introducción

② Pilas

③ Colas

④ Listas simples

⑤ Listas dobles

⑥ Arboles

⑦ Referencias

1 Introducción

2 Pilas

③ Colas

4 Listas simples

5 Listas dobles

6 Arboles

⑦ Referencias

Introducción

- Son un grupo o colección de datos que contienen operaciones para poder manipularlos
 - Pueden ser estáticos o dinámicos
 - Nos ayudan a representar datos reales en información
 - Pueden ser lineales o no lineales

Métodos

⑤ Listas dobles

① Introducción

③ Colas

⑥ Arboles

② Pilas

④ Listas simples

⑦ Referencias

Pilas

- Son una estructura de tipo LIFO (Last In First Out)
 - Puede ser estática o dinámica
 - Cuenta con 2 operaciones básicas
 - Tiene un tope
 - Si es estática hay que definir un tamaño máximo

Métodos

⑤ Listas dobles

① Introducción

③ Colas

⑥ Arboles

② Pilas

④ Listas simples

⑦ Referencias

Inserción

- Hay que comprobar que no este llena
 - El tope es el nuevo elemento insertado

Eliminación

- Hay que comprobar que no este vacía
- El nuevo tope será el elemento debajo del que eliminamos

1 Introducción

2 Pilas

3 Colas

Métodos

4 Listas simples

5 Listas dobles

6 Arboles

7 Referencias

Colas

- Son una estructura de tipo FIFO (First In First Out)
- Puede ser estática o dinámica
- Cuenta con 2 operaciones básicas
- El final es el último elemento
- Tiene un inicio
- Si es estática hay que definir un tamaño máximo

1 Introducción

2 Pilas

3 Colas

Métodos

4 Listas simples

5 Listas dobles

6 Arboles

7 Referencias

Inserción

- Hay que comprobar que no este llena
- El tope es el nuevo elemento insertado

Eliminación

- Hay que comprobar que no este vacía
- El inicio es el elemento siguiente al que eliminamos

1 Introducción

2 Pilas

3 Colas

4 Listas simples

Métodos

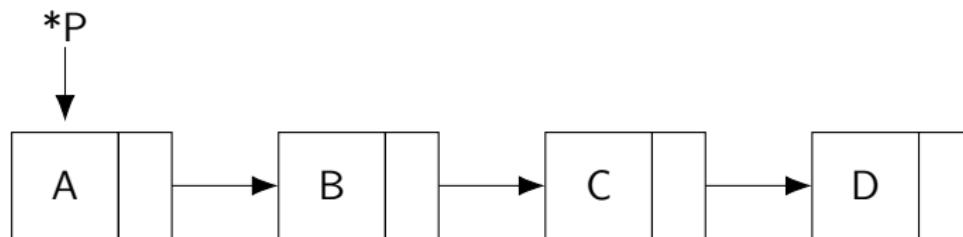
5 Listas dobles

6 Arboles

7 Referencias

Listas simples

- Son una estructura lineal
- Es dinámica
- Son una colección de nodos
- Cada nodo es un puntero
- Cada nodo apunta al que le sigue



1 Introducción

2 Pilas

3 Colas

4 Listas simples

Métodos

5 Listas dobles

6 Arboles

7 Referencias

Inserción al inicio

- Si está vacía el nodo será nulo
- Hay que localizar el primer nodo
- Creamos un nuevo nodo y apuntara al primer nodo
- El nuevo nodo será el primer nodo

Inserción al final

- Hay que localizar el último nodo
- Creamos un nodo y el último nodo apuntara a este nuevo nodo
- El nuevo nodo apuntara a un valor nulo

Inserción entre nodos

- Hay que localizar los nodos entre los cuáles queremos hacer la inserción
- El nodo anterior apunta al nuevo nodo
- El nuevo nodo apunta al nodo siguiente

Eliminación al inicio

- Hacemos que el inicio de la lista apunte al siguiente elemento
- El inicio será el elemento siguiente al que eliminamos
- Si la lista queda vacía el nodo será nulo

Eliminación al final

- Localizamos el último
- Hacemos que el nodo anterior al último sea nulo
- Eliminamos el último nodo

Eliminación entre nodos

- Localizamos el nodo a eliminar
- Hacemos que el nodo anterior apunte al nodo siguiente del que queremos eliminar
- Eliminamos el nodo

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

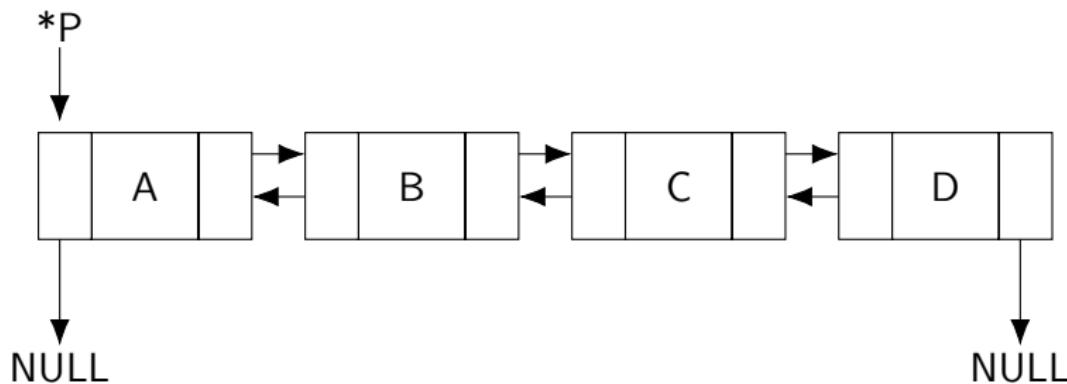
Métodos

6 Arboles

7 Referencias

Listas dobles

- Tienen dos nodos
- Un nodo apunta al siguiente nodo
- El otro nodo apunta al nodo anterior



① Introducción

② Pilas

③ Colas

④ Listas simples

⑤ Listas dobles

Métodos

⑥ Arboles

⑦ Referencias

Inserción al inicio

- Si esta vacía, ambos nodos serán anulos
- Localizamos ini ¹
- El nodo anterior de ini apuntara al nodo siguiente de ins²
- El nodo siguiente de ins apuntara al nodo anterior de ini
- El nodo anterior de ins será nulo

¹elemento inicial

²elemento a insertar

Inserción al final

- Localizamos fin ³
- El nodo siguiente de fin apuntara al nodo anterior de ins⁴
- El nodo anterior de ins apuntara al nodo siguiente de fin
- EL nodo siguiente de ins será nulo

³elemento final

⁴elemento a insertar

Inserción entre nodos

- Hay que localizar los nodos entre los cuáles queremos hacer la inserción ⁵
- El nodo siguiente de izq apuntara a nodo anterior de ins ⁶
- El nodo anterior de ins apuntara a nodo siguiente de izq
- Nodo siguiente de ins apuntara a nodo anterior de der
- Nodo anterior de der apuntara a nodo siguiente de ins

⁵izq y der

⁶elemento a insertar

Eliminación al inicio

- Si quedará vacía la lista apuntará a nulo
- Nodo anterior de nwini⁷ será nulo
- Eliminamos nodo siguiente de elim⁸

⁷segundo elemento

⁸elemento a eliminar

Eliminación al final

- Localizamos fin
- Hacemos nodo siguiente de nwfin⁹ nulo
- Hacemos nulo al nodo anterior de fin

⁹elemento anterior a fin

Eliminación entre nodos

- Localizamos elim
- Nodo siguiente de der apunta a nodo anterior de izq
- Nodo anterior de izq apunte a izq

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

6 Arboles

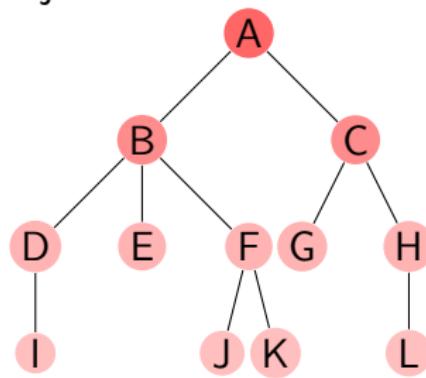
Arboles binarios

(ABB)
Arboles AVL
Arboles B
Arboles B+

7 Referencias

Arboles

Son estructuras dinámicas no lineales ordenadas y una de las mas importantes en computación. Pueden cambiar tanto de forma como de tamaño durante la ejecución.



Conceptos clave I

- **Raíz:** Es el nodo principal o inicial
- **Padre:** Un nodo que apunta hacia otros nodos
- **Hijo:** Un nodo que es apuntado por otro nodo
- **Hoja:** Nodo sin hijos
- **Nodo interior:** Nodo que no es raíz ni hoja
- **Grado:** Numero de hijos de un nodo
- **Grado del árbol:** Máximo grado de todos los nodos del árbol
- **Brazo:** Conexión entre un nodo y otro
- **Camino:** Secuencia de nodos y brazos conectados por un hijo
- **Nivel:** Número de brazos entre un nodo y la raíz mas uno
- **Altura:** Número de brazos entre en el camino mas largo entre un nodo y una hoja.

Conceptos clave II

- **Altura del árbol:** Altura del nodo raíz
- **Profundidad:** Número de brazos desde la raíz hasta un nodo
- **Rama:** Una ruta de la raíz a cualquier nodo

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

6 Arboles

Arboles binarios

(ABB)

Operaciones

Arboles AVL

Arboles B

Arboles B+

7 Referencias

Arboles binarios

Son árboles de grado 2 donde cada nodo tiene máximo dos subárboles, siendo subárbol izquierdo y derecho respectivamente., a la derecha se van colocando los valores mayores que la raíz y a la izquierda los menores.

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

6 Arboles

Arboles binarios

(ABB)
Operaciones

Arboles AVL

Arboles B

Arboles B+

7 Referencias

Inserción

- Si está vacío la raíz apunta a null
- Si insertamos un elemento en un árbol vacío, raíz apuntara a este nuevo elemento y sus dos subárboles apuntaran a null
- Si no está vacío comparamos el nuevo valor a insertar con la raíz, y vamos comparando con cada nodo según corresponda

Recorrido en preorden

- Visitamos la raíz
- Recorremos el subárbol izquierdo
- Recorremos el subárbol derecho

Recorrido en inorden

- Recorremos el subárbol izquierdo
- Visitamos la raíz
- Recorremos el subárbol derecho

Recorrido en posorden

- Recorremos el subárbol izquierdo
- Recorremos el subárbol derecho
- Visitamos la raíz

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

6 Arboles

Arboles binarios

(ABB)
Arboles AVL
Arboles B
Arboles B+

7 Referencias

Arboles balanceados

Son árboles que después de una inserción o eliminación se reacomodan manteniendo una diferencia de altura menor o igual 1 entre las ramas derecha e izquierda.

Ventajas respecto a árboles binarios

Los árboles balanceados surgieron como una solución a un problema que tienen los ABB, y es que si en un ABB se insertan datos ordenados de manera ascendente o descendente, estos crecerán de manera descontrolada.

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

6 Arboles

Arboles binarios

(ABB)
Arboles AVL
Arboles B
Arboles B+

7 Referencias

Arboles B

Son una generalización de los árboles AVL y representan un método para almacenar y recuperar información en medios externos.

Características

- Un grupo de nodos se llama página
- En cada página se almacena la información del grupo de nodos y se identifica por medio de claves o llaves.
- Cada página, excepto la raíz contiene **d** y **2d** elementos, siendo **d** el grado del árbol
- La página raíz tiene al menos dos descendientes
- Las páginas hoja están todas al mismo nivel
- Crecen de las hojas hacia la raíz

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

6 Arboles

Arboles binarios

(ABB)
Arboles AVL
Arboles B
Arboles B+

7 Referencias

Arboles B+

Son muy parecidos a los árboles B con la diferencia de que la información se encuentra en las hojas, mientras que los nodos raíz e interiores almacenan claves que se utilizan como índices.

Ventajas respecto a árboles B

Pese a que ocupan un poco mas de espacio debido a la duplicidad en algunas claves, esto es aceptable si el archivo se modifica frecuentemente, ya que esto evita la reorganización.

Características

- Cada página, excepto la raíz contiene m elementos, donde m es un elemento entre d y $2d$
- La raíz tiene entre 1 a $2d$
- Cada página, excepto la raíz, tiene entre $d + 1$ y $2d + 1$ descendientes
- La raíz tiene al menos dos descendientes
- Las hojas están todas al mismo nivel
- Toda la información, con las claves que las identifican, se encuentran en las hojas
- Las claves en la raíz y páginas interiores se utilizan como índices

1 Introducción

2 Pilas

3 Colas

4 Listas simples

5 Listas dobles

6 Arboles

7 Referencias

Referencias I

- [1] Bhasin, H.
Python Basics, 3 ed.
David Pallai, Mercury Learning and Information, 2019.
- [2] Brassard, G., and Bratley, P.
Algorithmics: Theory and Practice, 31 ed.
Prentice Hall, 1988.
- [3] CAIRÓ, O., and Guardati, S.
Algorithms, 3 ed.
McGraw-Hill Interamericana, 2006.
- [4] CAIRÓ, O., and Guardati, S.
Estructuras de Datos, 3 ed.
McGraw-Hill Interamericana, 2006.

Referencias II

- [5] Deitel, P., and Deitel, H.
C++ Como Programar, 9 ed.
Pearson Educación de México, 2014.
- [6] Downey, B., A.
Think Python, 2 ed.
O'Reilly, 2016.
- [7] Jaworski, M., and Ziadé, T.
Expert Python Programming, 3 ed.
Packt Publishing, 2019.
- [8] Laakmann, M., G.
Cracking the Coding Interview, 6 ed.
CarrerCup, 2016.

Referencias III

- [9] Matthes, E.
Python Crash Course, 2 ed.
No Star Press, 2019.
- [10] Ramalho, L.
Fluent Python, 1 ed.
OReilly, 2014.
- [11] Reek, K.
Pointers On C.
Addison-Wesley Longman, 1997.
- [12] Sedgewick, R., and Wayne, K.
Algorithms, 4 ed.
Pearson Education, 2011.

Referencias IV

[13] Stroustrup, B.

The C++ Programming Language, 4 ed.
Pearson Education, 2013.

[14] Stroustrup, B.

Principles and Practice Using C++, 2 ed.
Pearson Education, 2014.

[15] unknown.

Cplusplus.