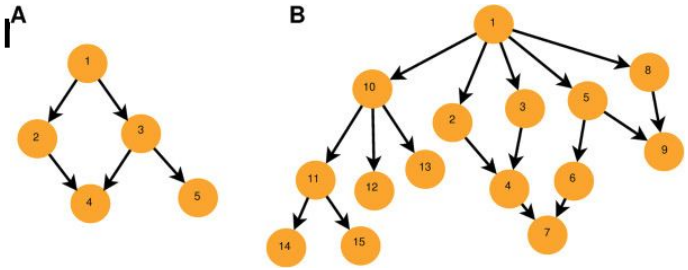
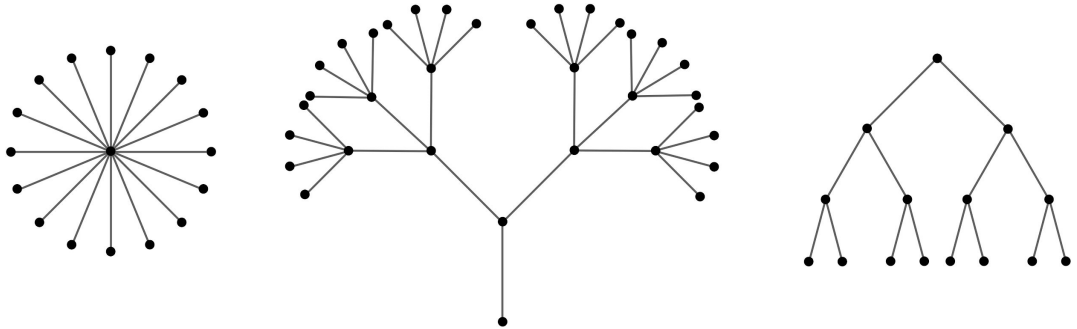
Arboles

Arbol: = Grafo no dirigido, conectado, y sin ciclos
Arbol binario:= Cada nodo tiene 2 hijos
Arbol binario de busqueda:= orden



BST = binary search tree

No confundir con DAG = directed acyclic graph = sin ciclos dirijidos



Sea G un grafo de n nodos, los siguientes son equivalentes:

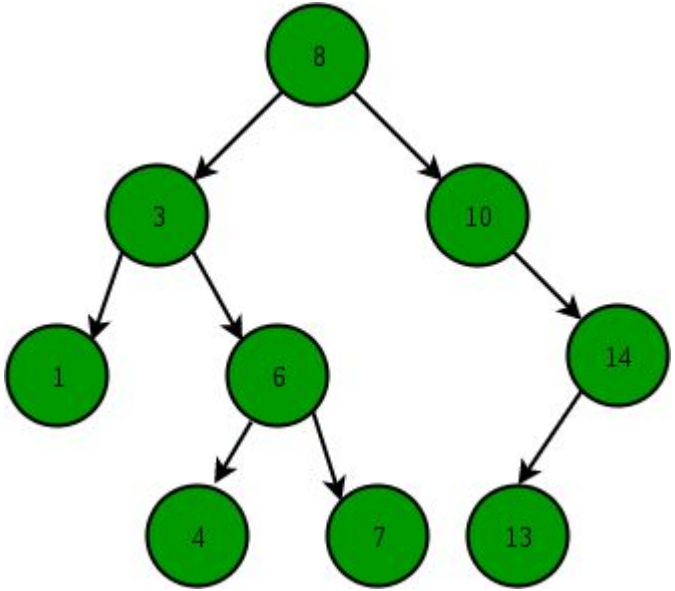
- G es un arbol
- Entre cada par de nodos de G hay camino unico
- G es conectado, si quita un arista G se desconecta
- G no tiene ciclos, si anade un arista, tendria ciclo
- G es conectado y tiene exactament n-1 aristas
- G no tiene ciclos y tiene exactamente n-1 ciclos

BST

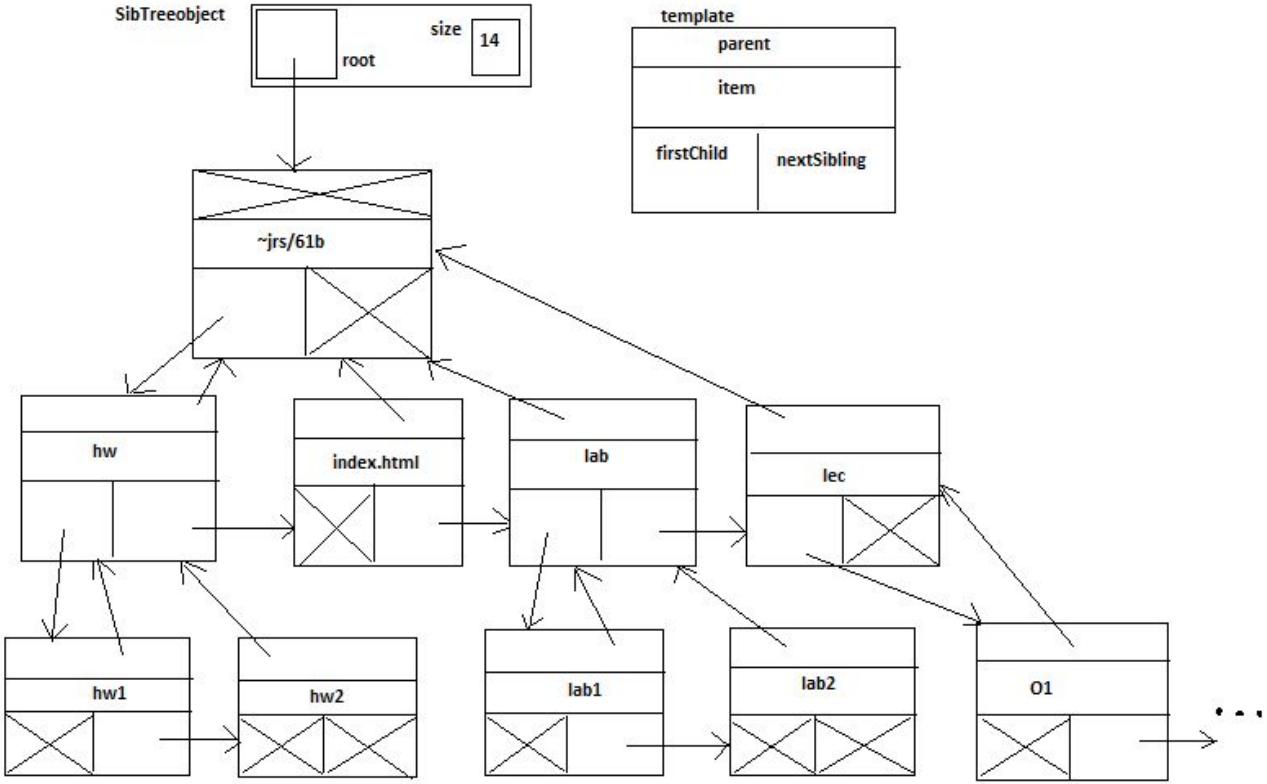
```
Struct Tree {
  Int key;
  Tree* left;
  Tree* right;
};
```

NOTA: no hay arbol sin nodo

Terminos: raiz, hoja, hijo / padre

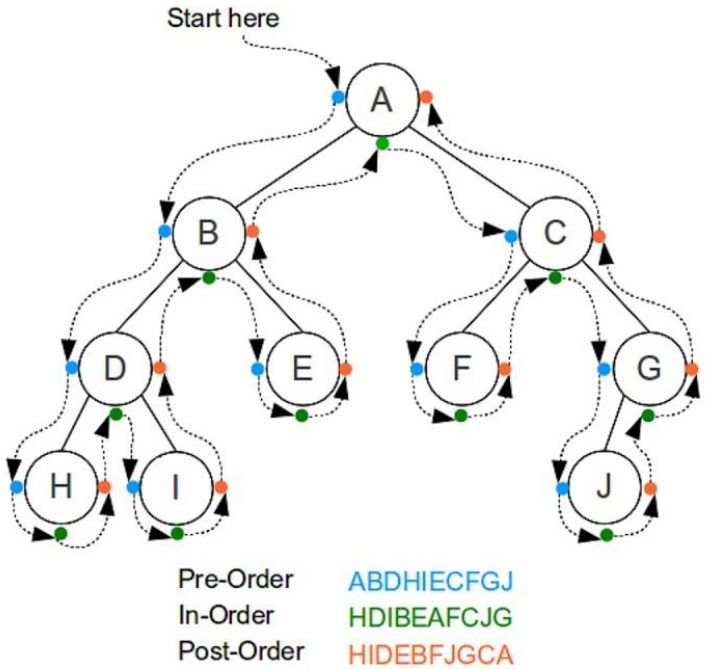


Otras representaciones



BST Recorido

Simplest Trick to find PreOrder InOrder PostOrder



```
Void visit(Tree* tree) {
    // base case omitido
    visit(tree->left);
    visit(tree->right);
    print(tree->key);
}
```

Recursiones

Si resuelvo el problema para los hijos, como resuelvo mi problema?

- Altura: a(r) = 1 + max(a(l), a(r))
- Numeros de nodos: n(r) = 1 + n(l) + n(r)

Implementar map con BST

- Insertar
- Eliminar
- encontrar

Arboles aumentados

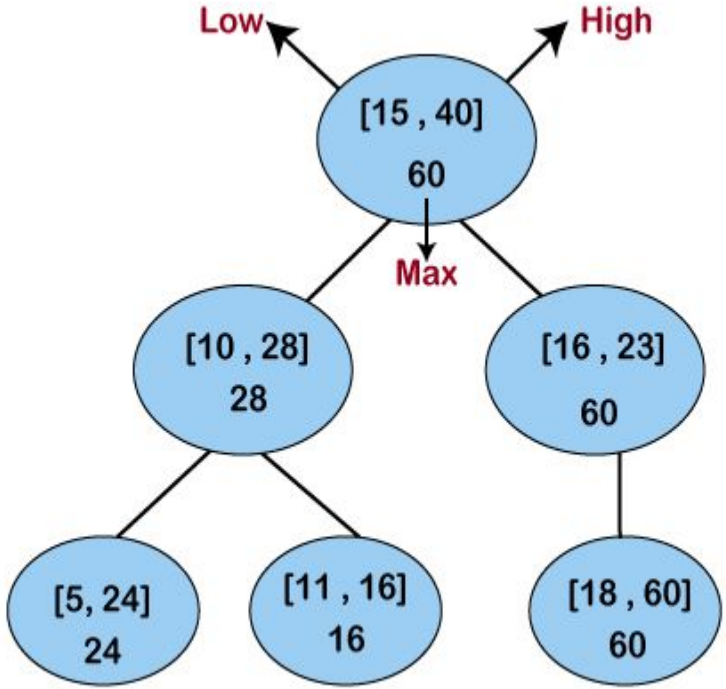
- Datos extra en los nodos:
  - Datos extra en el node solo depende de sus hijos.
- Operaciones normales: insertar, eliminar = O(log n)
- + Operacion especial dependiendo del dato adicional

k-estadistica

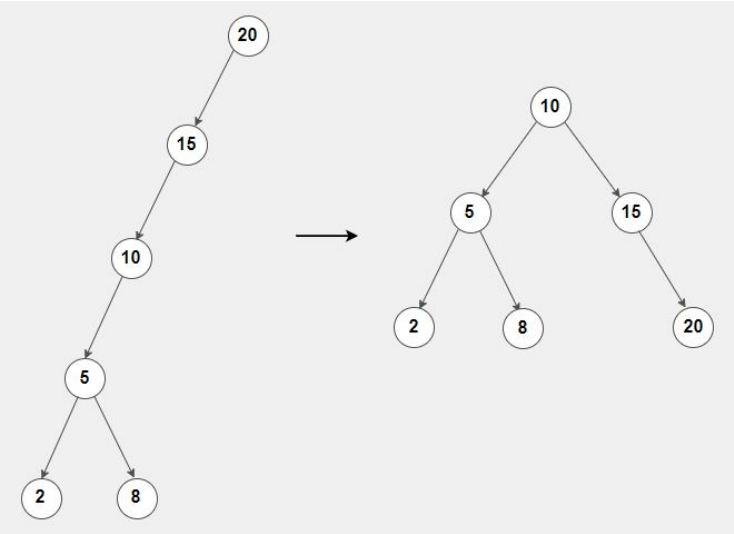
- Retornar k-grande del arbol
- Datos a aumentar: numeros de elementos en el arbol

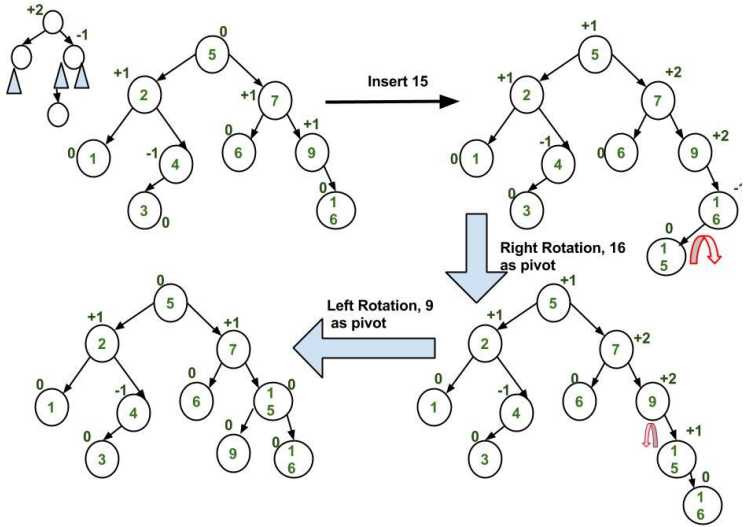
Arboles de intervalo

- Intervalos cerrados
- Ordenado por el limite menor
- Datos extra: maximo del subarbol.
- Operaciones extra: dado un intervalo, retornar si algun intervalo en el arbol intersecta con el dado.



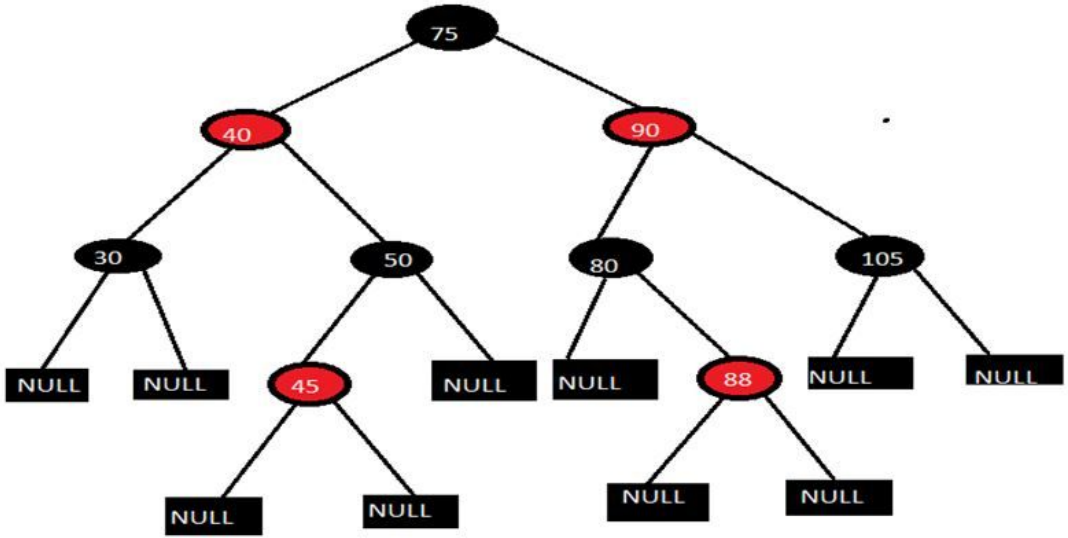
Balancear arbol





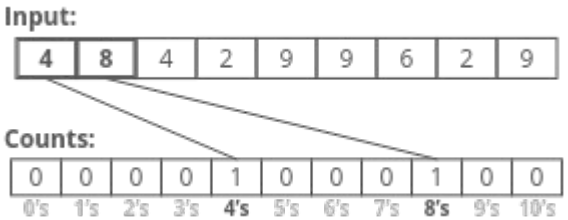
https://en.wikipedia.org/wiki/AVL_tree#/media/File:AVL_Tree_Example.gif

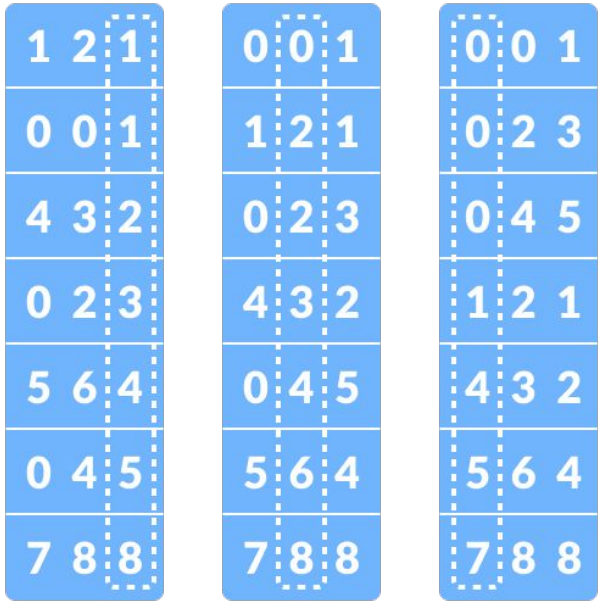
- AVL :
  - Aumentar numero de altura
  - Diferencia = 1
- Red - Black:
  - Diferencia = 2x
  - Aumentar boolean (red/black



Ordenamiento Rapidos

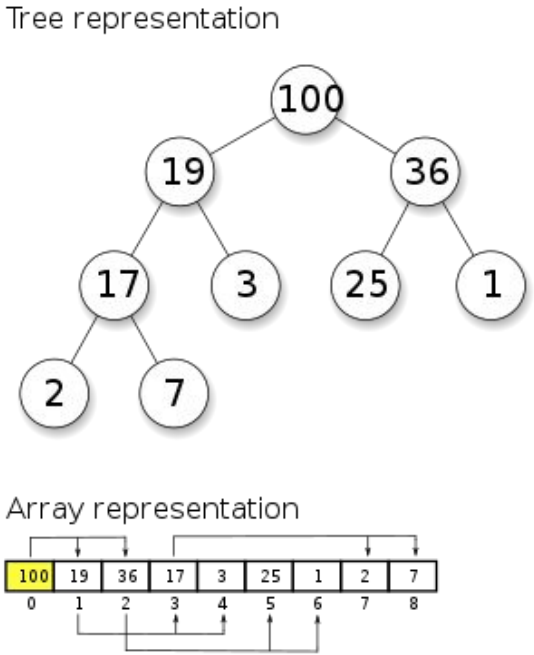
- Ordenamiento de conteo (counting sort)
  - O(n)
  - rango debe ser pequeno
- Ordenamiento de radix
  - O(kn) - k = numero de digitos
  - Solo enteros





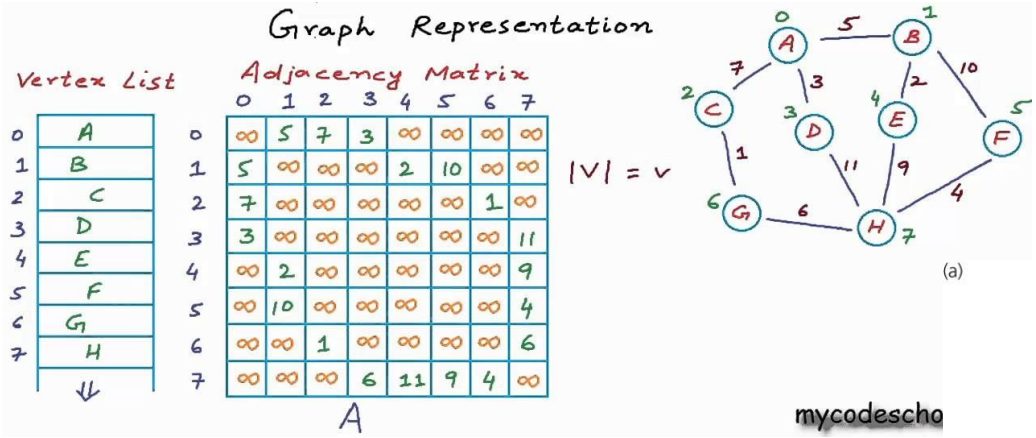
sorting the integers according to units, tens and hundreds place digits

Monticulos

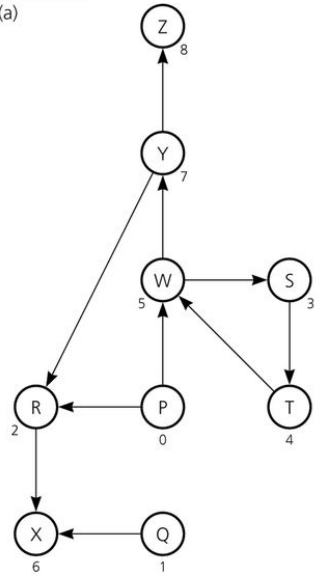


- Stdlib: make_heap / push_heap / pop_heap
-

Matrix de adjacencia



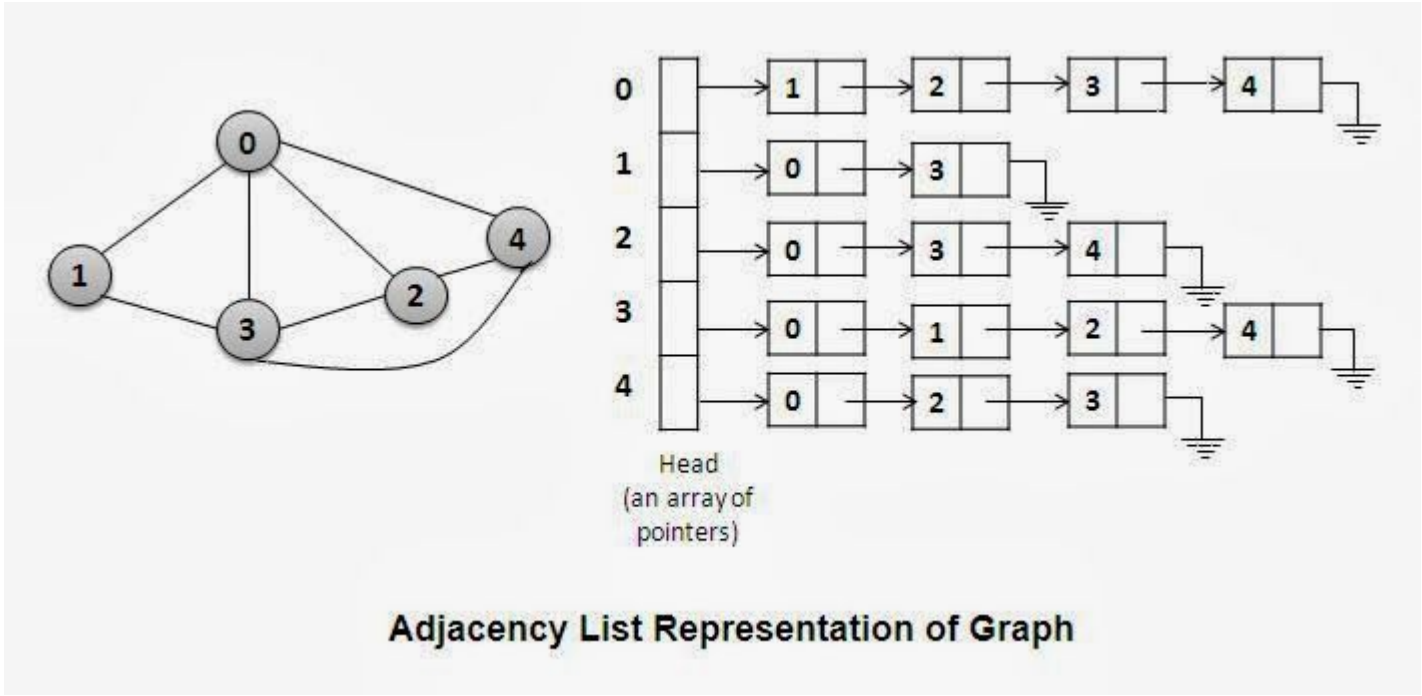
(a)



(b)

		P	Q	R	S	T	W	X	Y	Z
0	P	0	0	1	0	0	1	0	0	0
1	Q	0	0	0	0	0	0	1	0	0
2	R	0	0	0	0	0	0	1	0	0
3	S	0	0	0	0	1	0	0	0	0
4	T	0	0	0	0	0	1	0	0	0
5	W	0	0	0	1	0	0	0	1	0
6	X	0	0	0	0	0	0	0	0	0
7	Y	0	0	1	0	0	0	0	0	1
8	Z	0	0	0	0	0	0	0	0	0

Lista de adjacencia / lista de punteros



Adjacency List Representation of Graph