3001A-V22 ESTRUCTURA DE DATOS Y ALGORITMOS

ÁRBOLES BINARIOS DE BÚSQUEDA

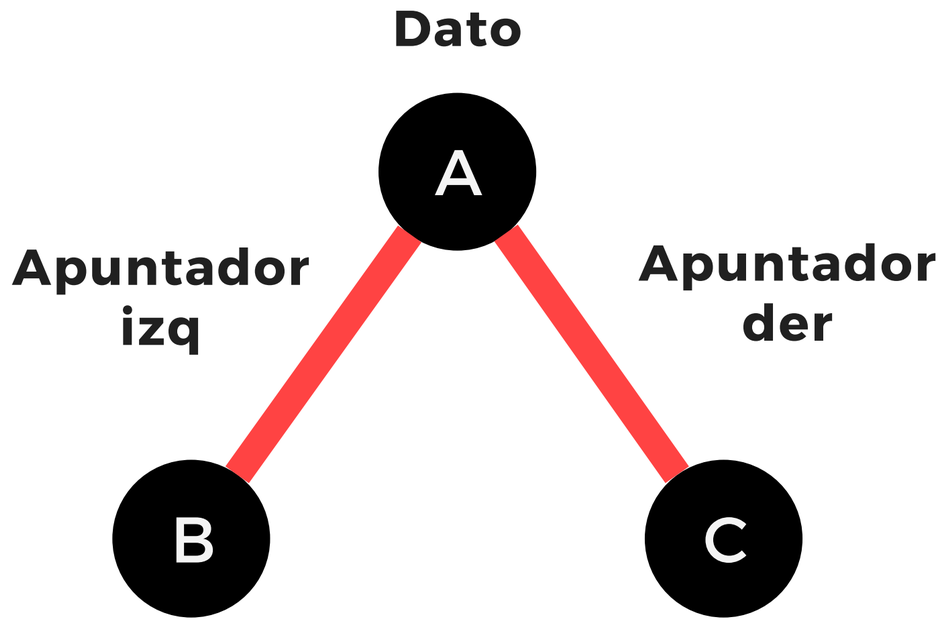
Profesor: Rafael Pérez Aguirre

VERANO 2022

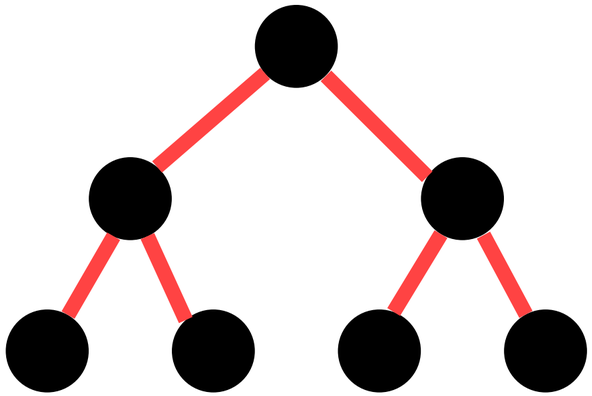
Árbol binario

Un árbol binario es una estructura de datos de árbol en la que cada nodo padre puede tener como máximo dos hijos. Cada nodo de un árbol binario consta de tres elementos:

- Dato
- Apuntador del hijo izquierdo
- Apuntador del hijo derecho



Tipos de árboles binarios



ÁRBOL BINARIO LLENO

Este árbol tiene 2 hijos en todos sus nodos (raíz no) o 0 hijos.

ÁRBOL BINARIO PERFECTO

Este árbol tiene todos sus nodos hoja en el mismo nivel.

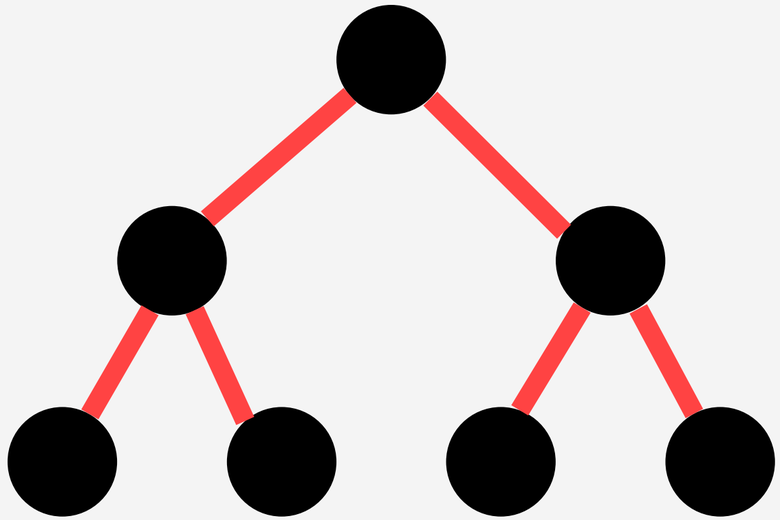
ÁRBOL DEGENERADO

Es el árbol que tiene un solo hijo, ya sea a la izquierda o a la derecha.

ÁRBOL BINARIO SESGADO

Es un árbol degenerado en el que el árbol está dominado por los nodos izquierdos o los nodos derechos

Árbol binario de búsqueda (BST)

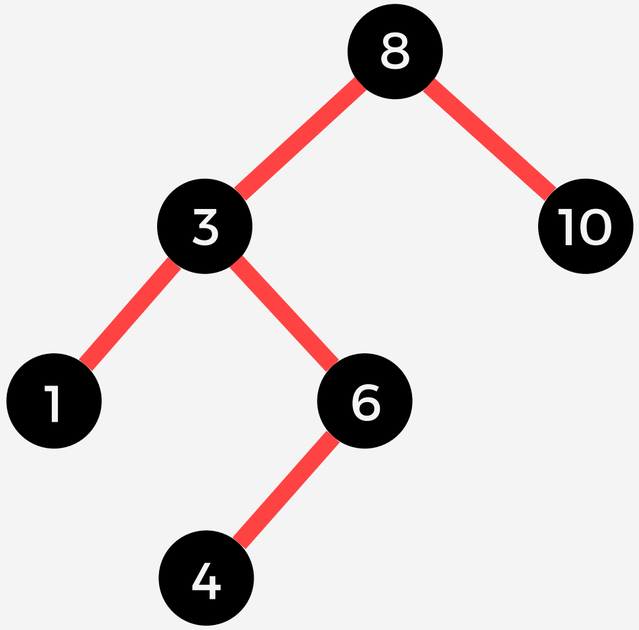


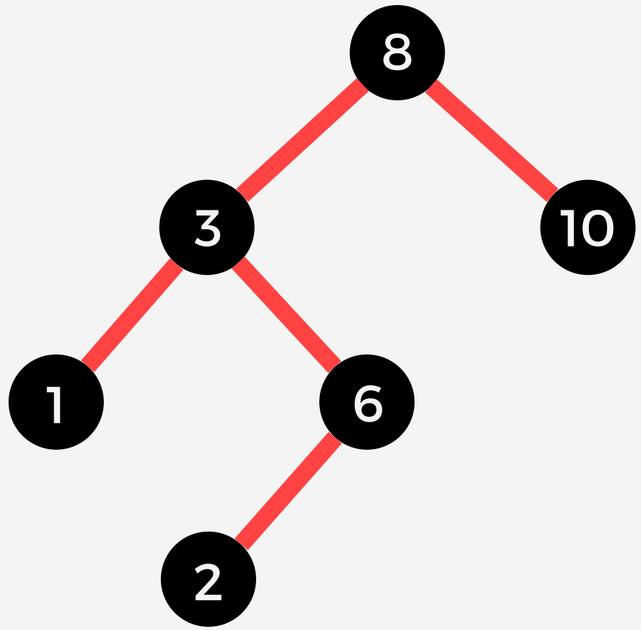
Es una estructura de datos que nos permite mantener rápidamente una lista ordenada de números.

- Se llama árbol binario porque cada nodo del árbol tiene un máximo de dos hijos.
- Se llama árbol de búsqueda porque puede usarse para buscar la presencia de un número en el tiempo O(log(n)).

Propiedades de los BST

- Todos los nodos del subárbol izquierdo son menores que el nodo raíz
- Todos los nodos del subárbol derecho son más que el nodo raíz
- Ambos subárboles de cada nodo también son BST, es decir, tienen las dos propiedades anteriores





Operaciones de los árboles binarios de búsqueda

Hay dos operaciones básicas que puede realizar en un árbol de búsqueda binaria:

- **Operación de búsqueda**
- **Operación de inserción**

Operación de búsqueda

¿EXISTE? **OBTENER**

El algoritmo depende de la propiedad del árbol, de que si cada subárbol izquierdo tiene valores por debajo de la raíz y cada subárbol derecho tiene valores por encima de la raíz.

Si el valor está por debajo de la raíz, podemos decir con certeza que el valor no está en el subárbol derecho; solo necesitamos buscar en el subárbol izquierdo y si el valor está por encima de la raíz, podemos decir con seguridad que el valor no está en el subárbol izquierdo; solo necesitamos buscar en el subárbol derecho.

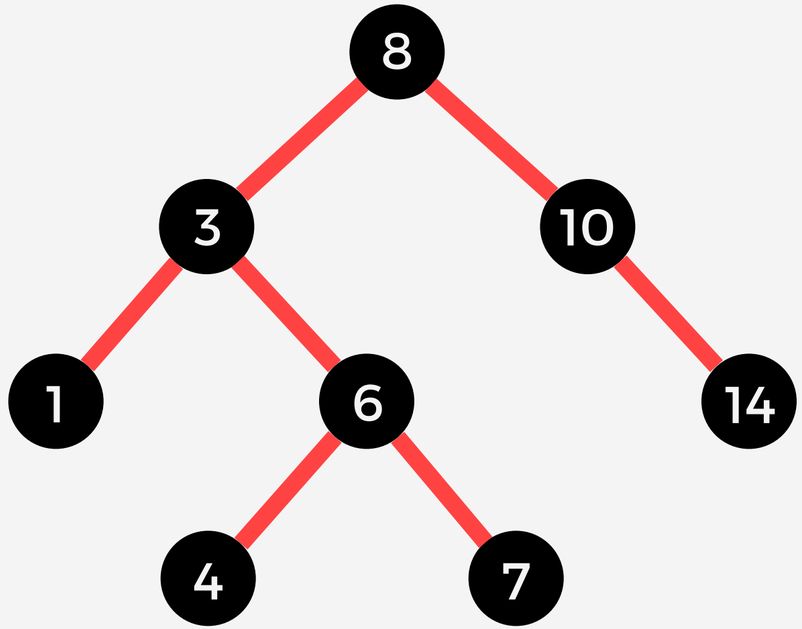
Operación de búsqueda

ALGORITMO

Algoritmo

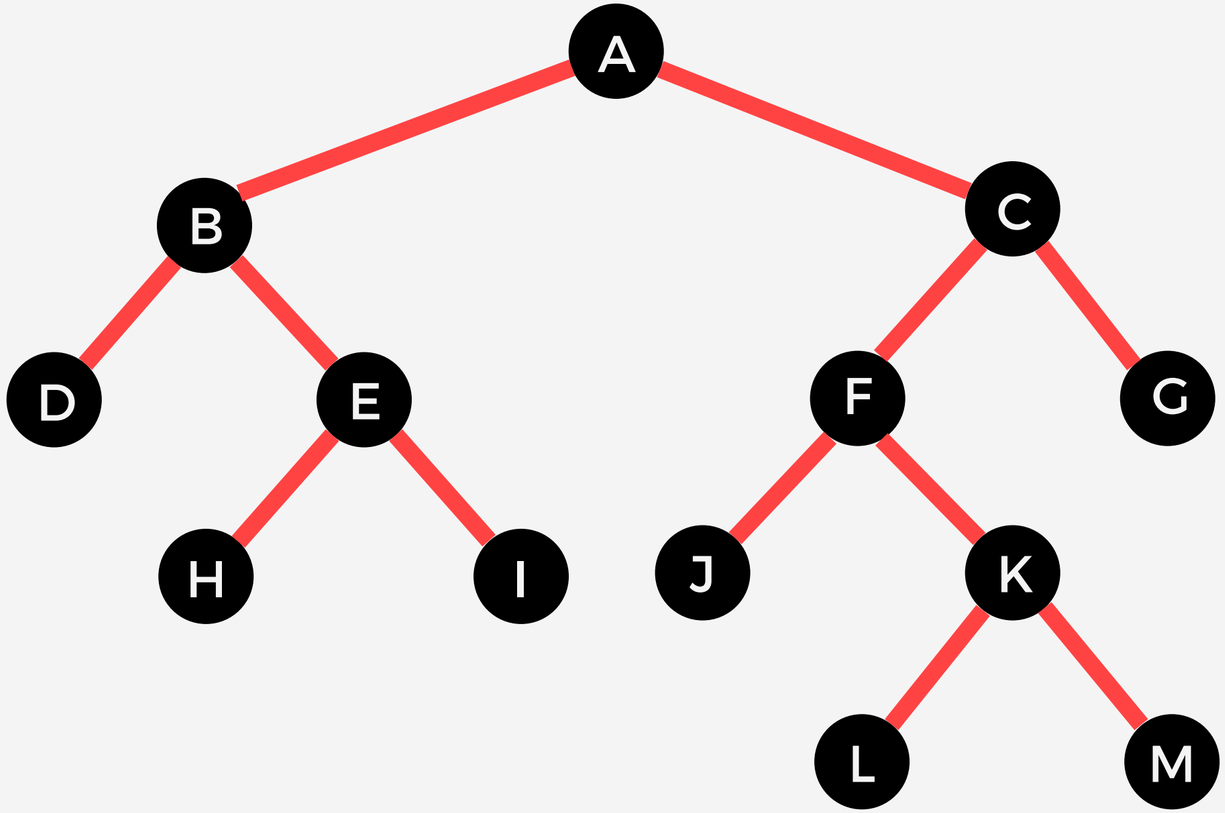
```
If root == NULL
    return NULL;
If number == root->data
    return root->data;
If number < root->data
    return search(root->left)
If number > root->data
    return search(root->right)
```

ENCONTRAR EL 4

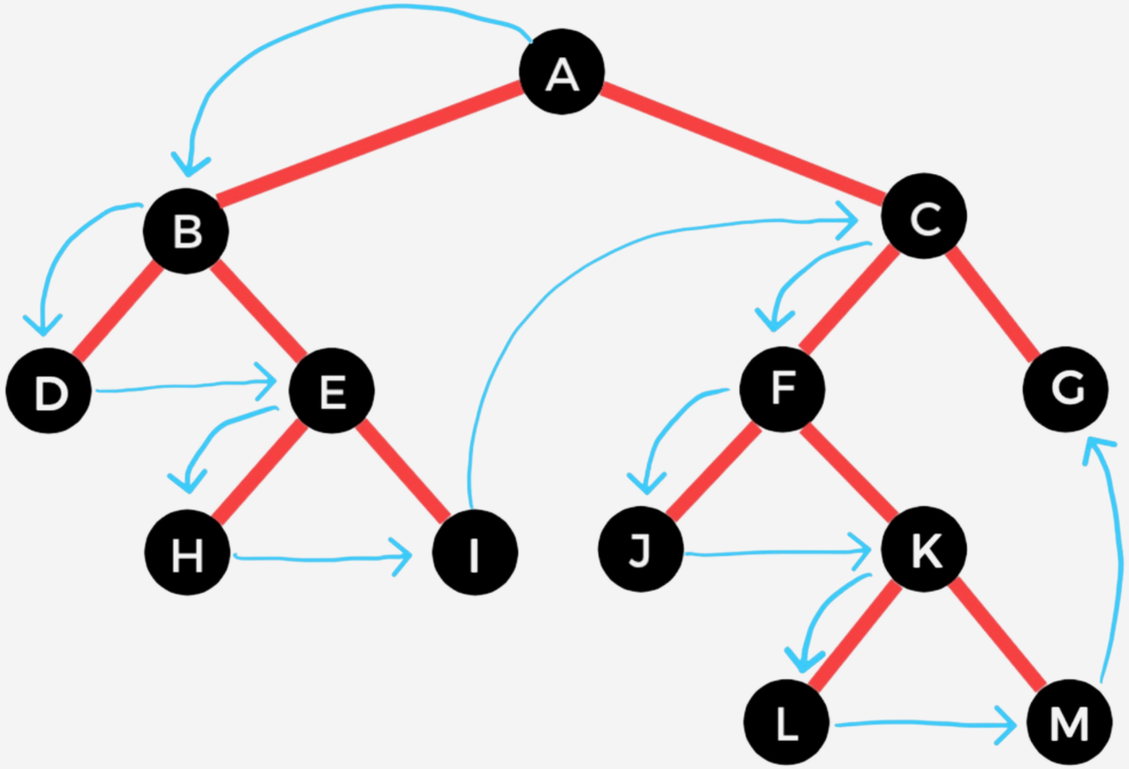


Recorrido de árboles

ACTIVIDAD



- Preorden
- Inorden
- Postorden

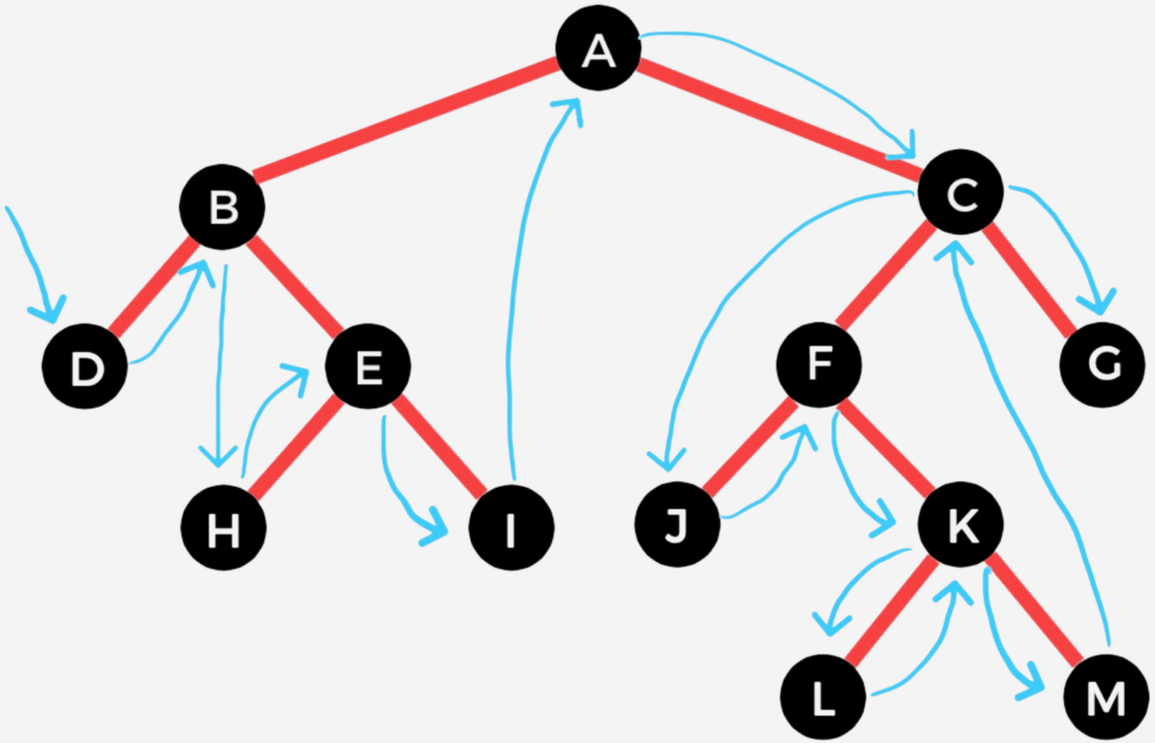


Preorden

A, B, D, E, H, I, C, F, J, K, L, M, G

Inorden

Postorden



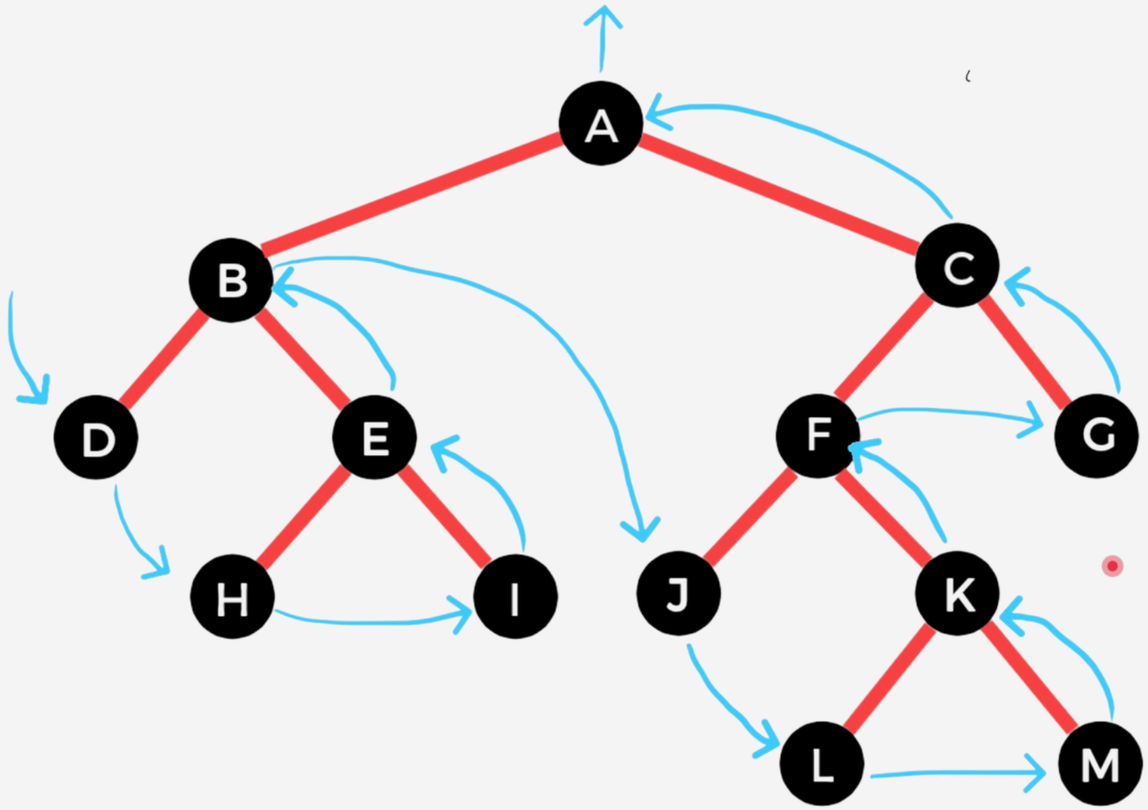
Preorden

A,B,D,E,H,I,C,F,J,K,L,M,G

Inorden

D,B,H,E,I,A,J,F,L,K,M,C,G

Postord



Preorden

A, B, D, E, H, I, C, F, J, K, L, M, G

Inorden

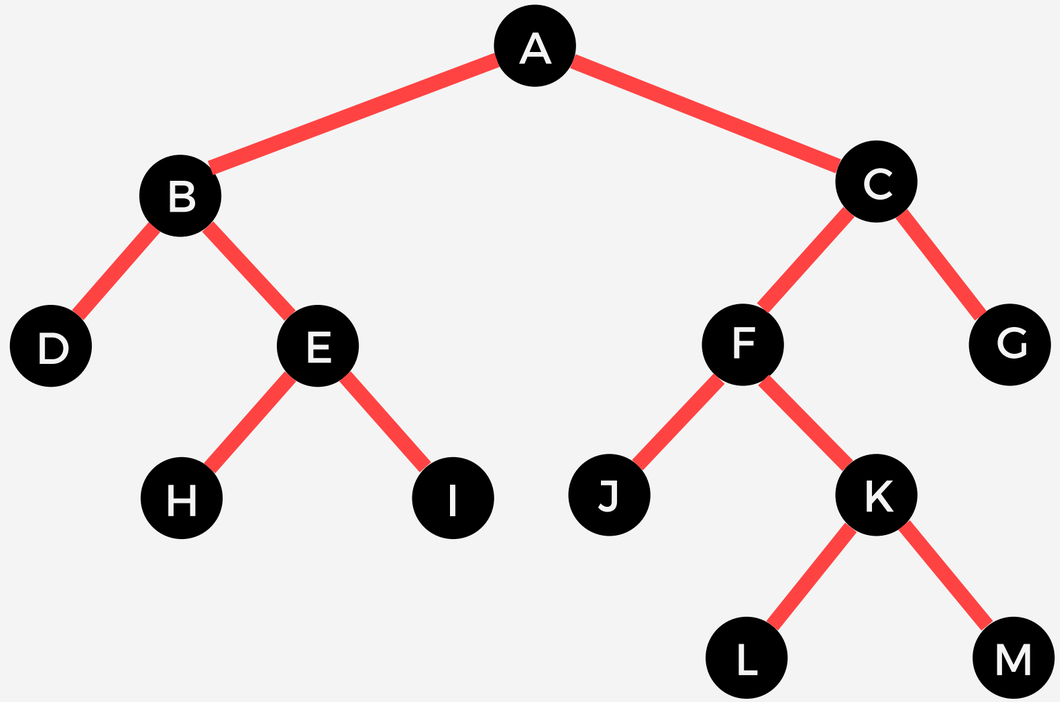
D, B, H, E, I, A, J, F, L, K, M, C, G

Postorden

D, H, I, E, B, J, L, M, K, F, G, C, A

TAREA

- Remplaza las letras del siguiente árbol, por números. El resultado deberá ser un árbol binario de búsqueda.
- Crea el mismo árbol en Python y ejecuta los 3 recorridos en profundidad.
- Comprueba los resultados.
- ¿Qué recorrido imprimió los nodos del árbol en orden?



Operación de inserción

Insertar datos en un árbol binario de búsqueda debe seguir la regla de que los datos del árbol están ordenados.
¿Qué pasa si el dato a insertar ya se encuentra en el árbol (duplicado)?

- **Si el árbol está vacío**
  - Insertamos en la raíz
- **Si la raíz del árbol es igual al elemento a insertar**
  - ¿?
- **Si la raíz del árbol es mayor al elemento a insertar**
  - Insertamos en el subárbol izquierdo
- **Si la raíz del árbol es menor al elemento a insertar:**
  - Insertamos en el subárbol derecho

Operación de inserción

ALGORITMO

Algoritmo

```
If node == NULL
    return createNode(data)
if (data < node->data)
    node->left  = insert(node->left, data);
else if (data > node->data)
    node->right = insert(node->right, data);
return node;
```

INSERTAR EL 4