INF 263 – Algoritmia

Aplicaciones de Estructuras de Datos Árboles

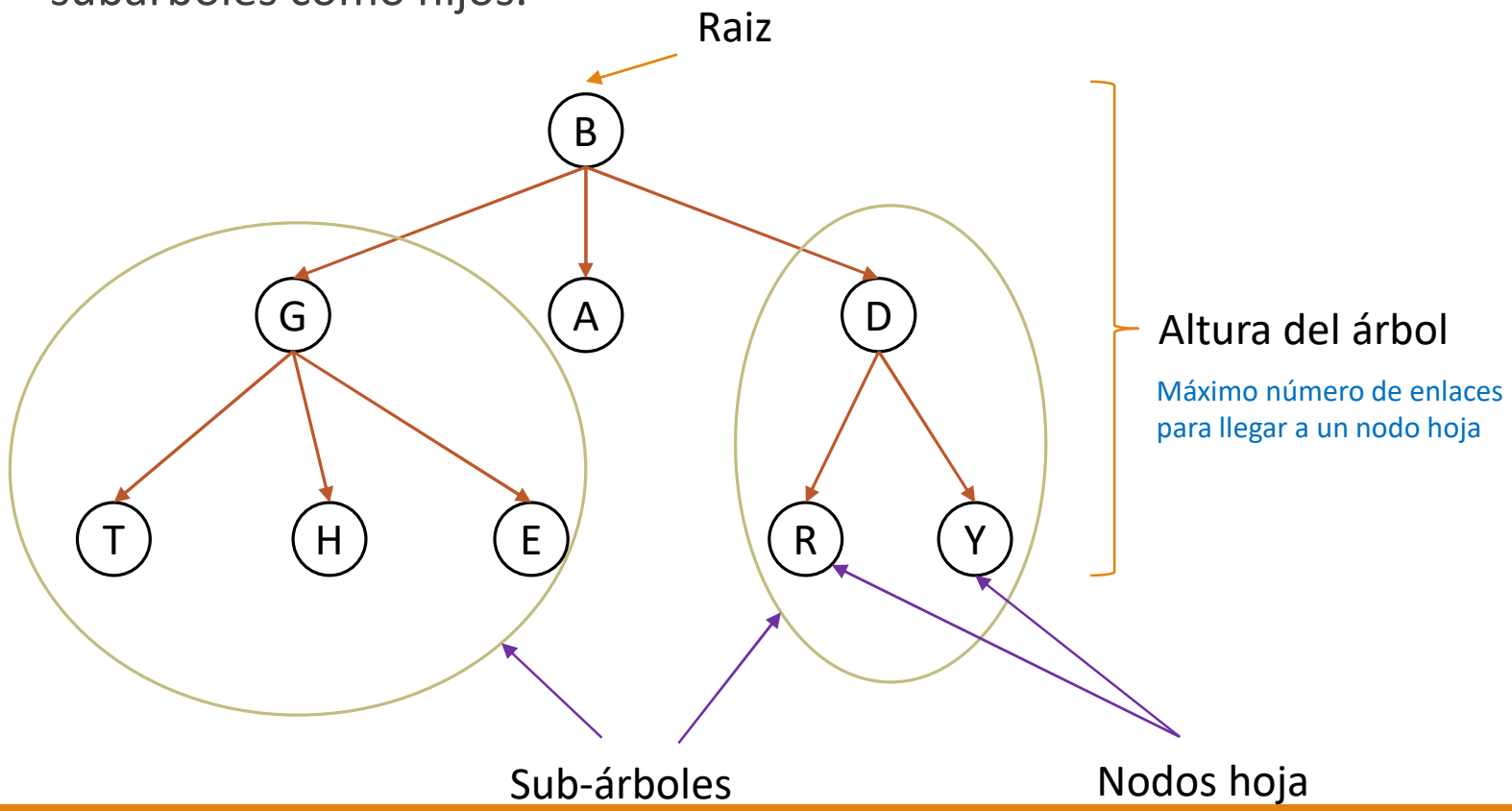
MG. JOHAN BALDEON

MG. RONY CUEVA MOSCOSO

2021-1

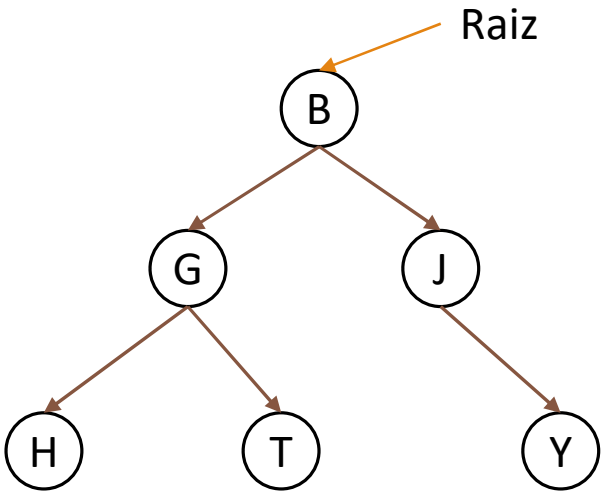
Definición

- Es una estructura de dato jerárquica que tiene un nodo raíz y subárboles como hijos.



Árboles Binarios

- Son árboles en donde cada nodo tiene a los más dos hijos.



Terminología

- Hijo izquierdo
- Hijo derecho
- Nodo padre

Operaciones sobre árboles binarios

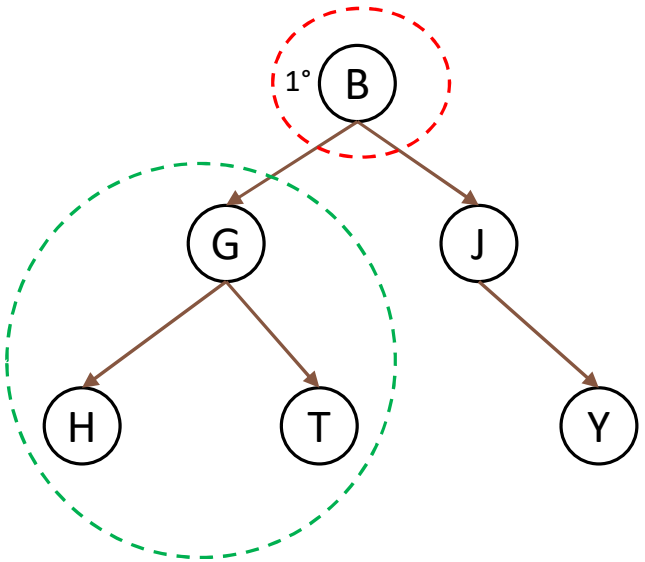
- Crear árbol
- Verificar si el árbol está vacío
- Imprimir elementos del árbol
- Determinar la altura del árbol
- Buscar un elemento
- Buscar el padre de un elemento
- Insertar un elemento a la izquierda de otro elemento
- Insertar un elemento a la derecha de otro elemento
- Finalizar un árbol

Recorridos en árboles binarios

- Muchas operaciones se basan en recorrer los elementos de un árbol.
- Como no es una estructura lineal (como los arreglos o las listas), se necesita de algoritmos diferentes para recorrerlos.
- Tres formas de recorrer un árbol
  - Pre-orden
  - En-orden
  - Post-orden

Recorridos en árboles binarios

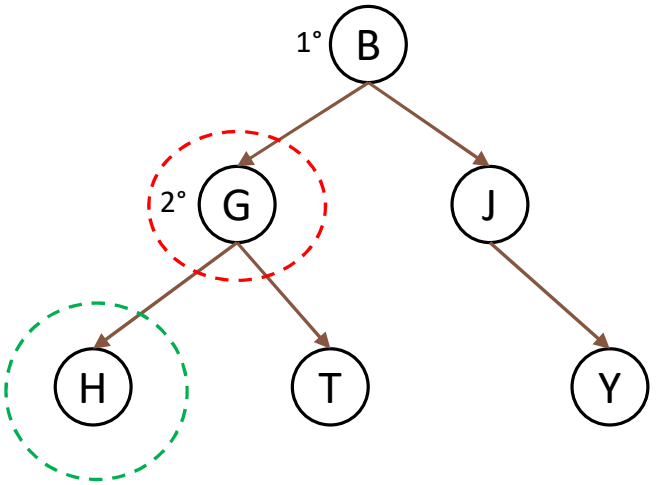
- **Pre-orden**: se visita primero el nodo raíz y después los subárboles izquierdo en *pre-orden* y derecho en *pre-orden*.



Orden de visita: B

Recorridos en árboles binarios

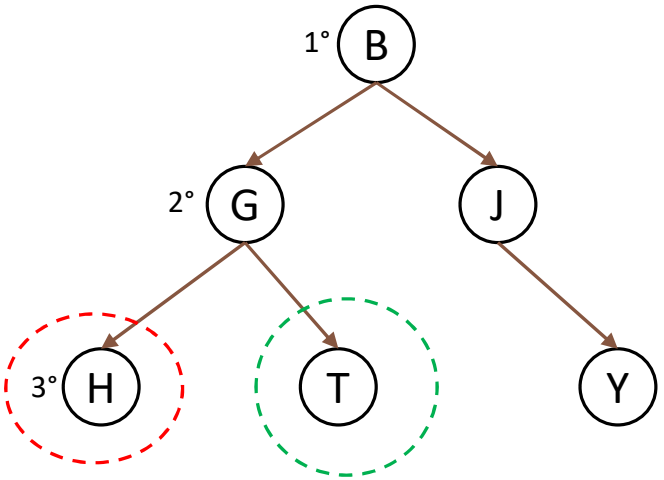
- **Pre-orden**: se visita primero el nodo raíz y después los subárboles izquierdo en *pre-orden* y derecho en *pre-orden*.



Orden de visita: B G

Recorridos en árboles binarios

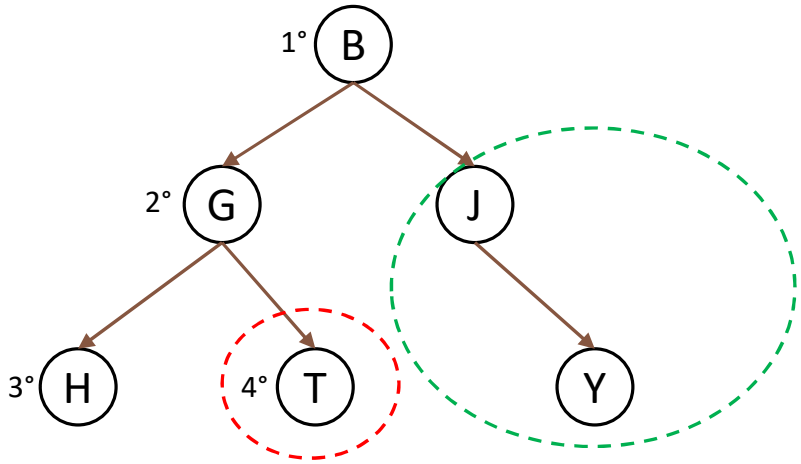
- **Pre-orden**: se visita primero el nodo raíz y después los subárboles izquierdo en *pre-orden* y derecho en *pre-orden*.



Orden de visita: B G H

Recorridos en árboles binarios

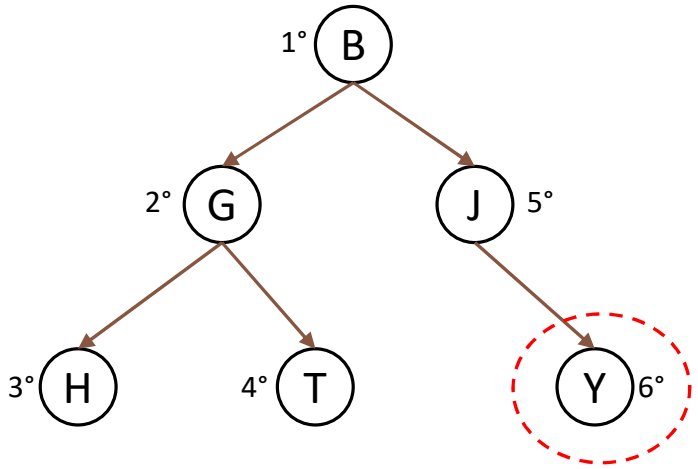
- **Pre-orden**: se visita primero el nodo raíz y después los subárboles izquierdo en *pre-orden* y derecho en *pre-orden*.



Orden de visita: B G H T

Recorridos en árboles binarios

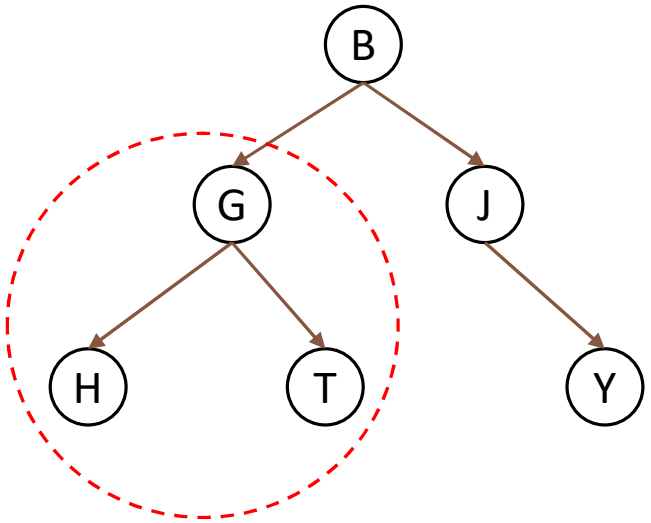
- **Pre-orden**: se visita primero el nodo raíz y después los subárboles izquierdo en *pre-orden* y derecho en *pre-orden*.



Orden de visita: B G H T J Y

Recorridos en árboles binarios

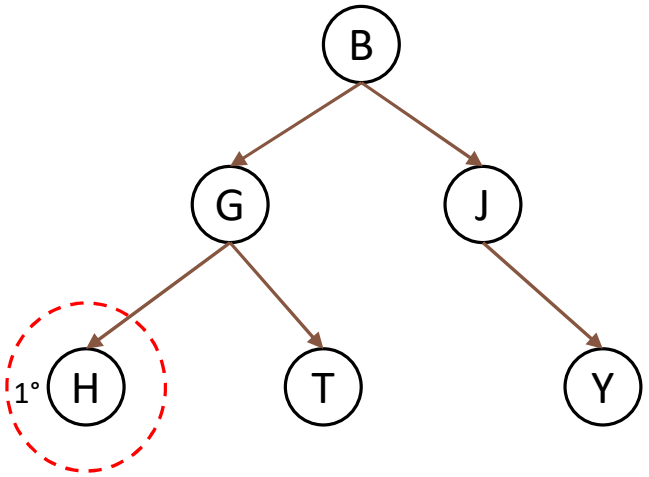
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita:

Recorridos en árboles binarios

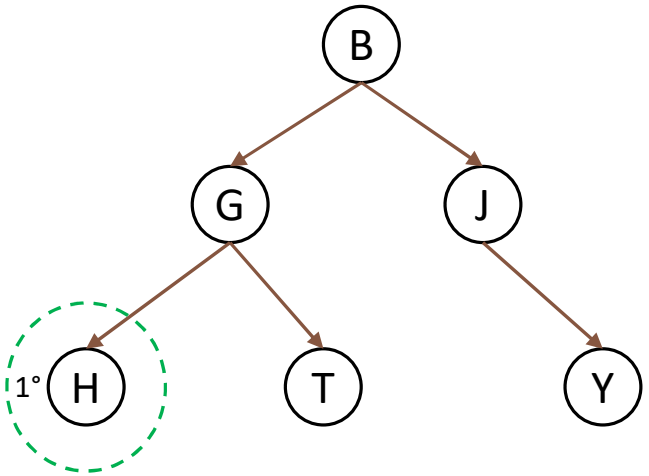
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita:

Recorridos en árboles binarios

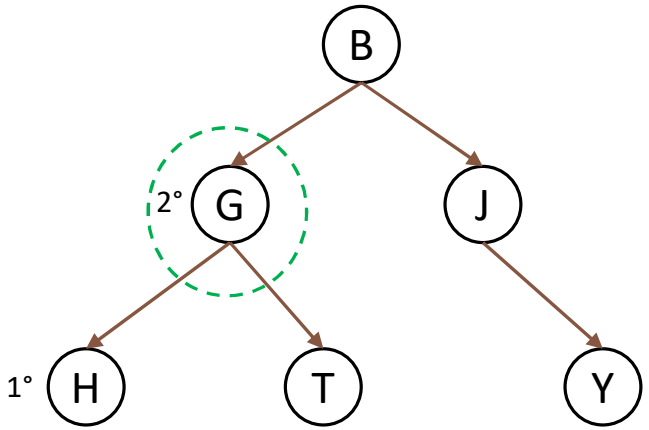
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita: H

Recorridos en árboles binarios

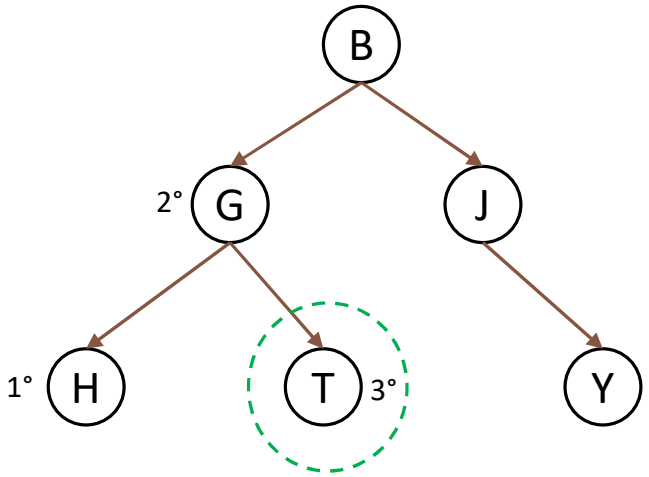
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita: H G

Recorridos en árboles binarios

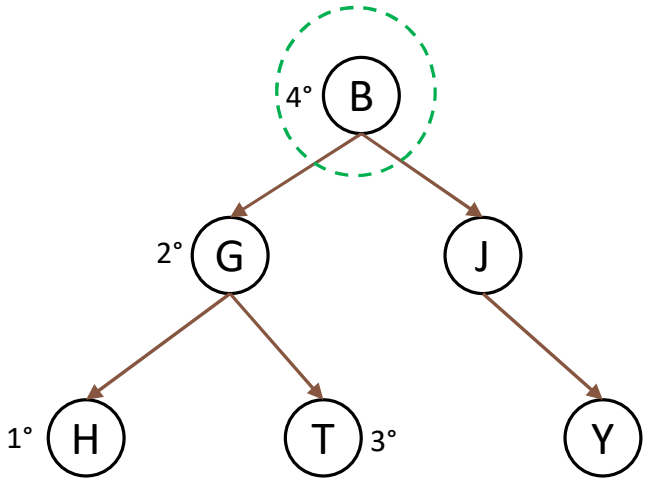
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita: H G T

Recorridos en árboles binarios

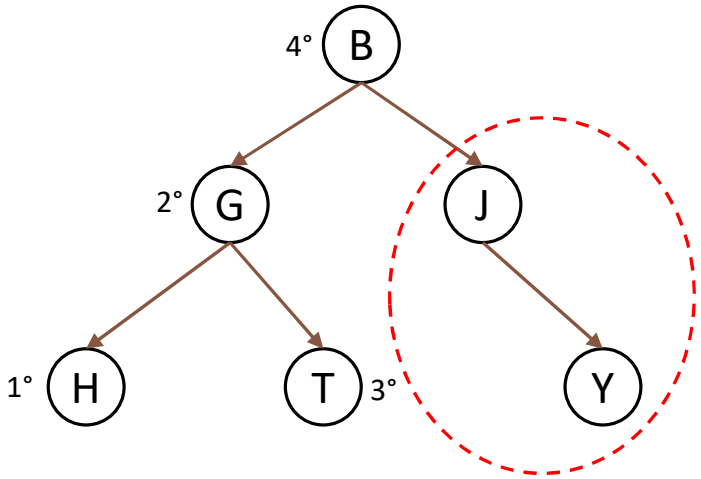
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita: H G T B

Recorridos en árboles binarios

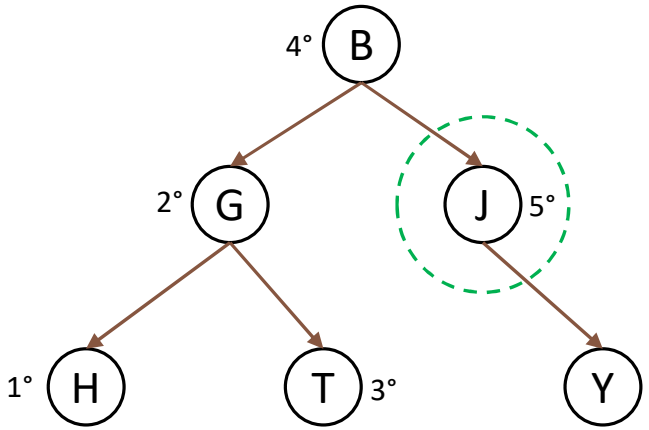
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita: H G T B

Recorridos en árboles binarios

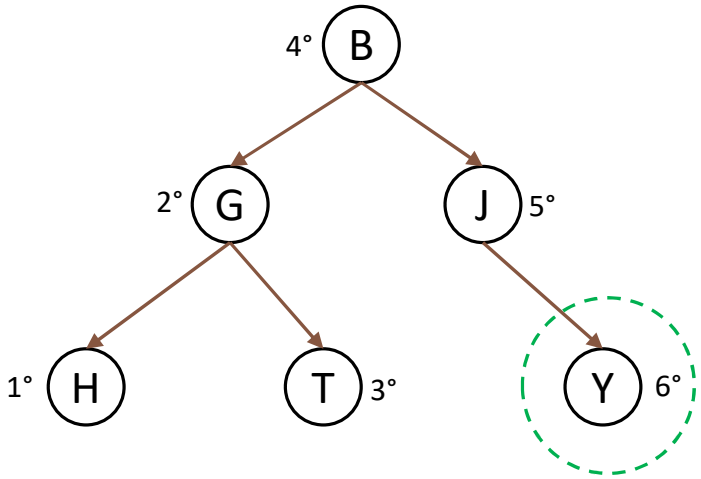
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita: H G T B J

Recorridos en árboles binarios

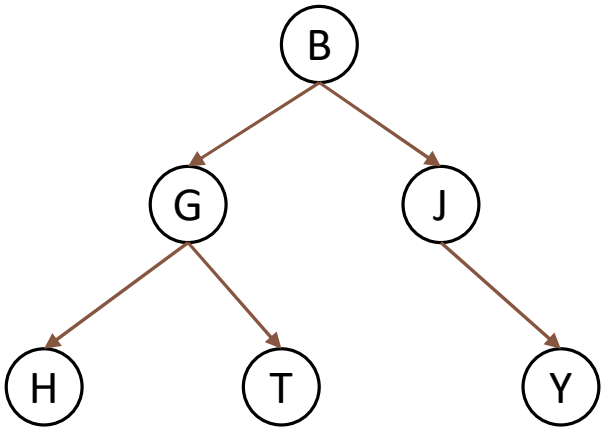
- **En-orden**: se visita primero subárbol izquierdo en en-orden, el nodo raíz y el subárbol derecho en en-orden.



Orden de visita: H G T B J Y

Recorridos en árboles binarios

- **Post-orden**: se visita primero subárbol izquierdo en post-orden, el subárbol derecho en post-orden y el nodo raíz

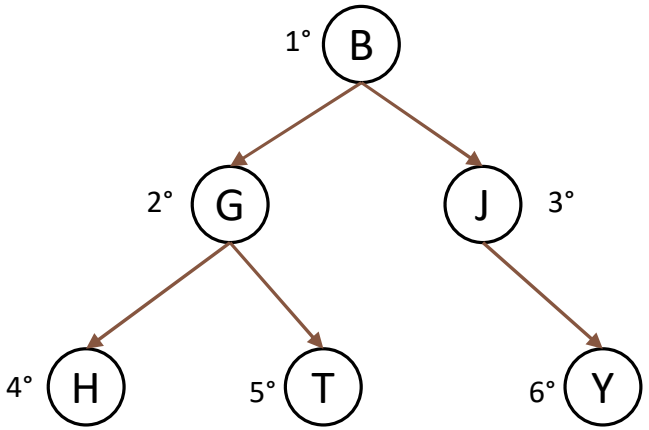


Cuál es el orden de visita?

H T G Y J B

Recorridos en árboles binarios

- **Otros recorridos:**
  - **Recorrido en amplitud:** Recorrer el árbol nivel por nivel



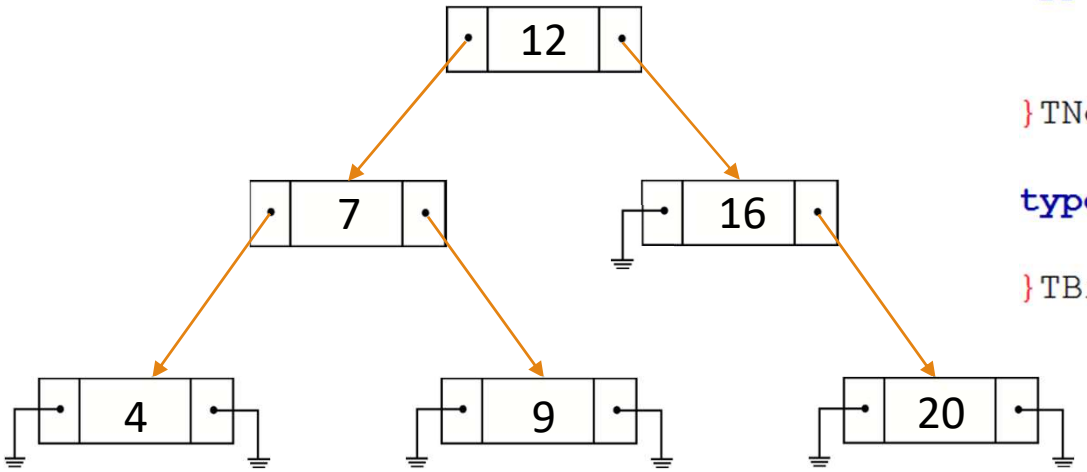
Orden de visita: B G J H T Y

  - **Recorrido en profundidad**: En - orden

ABB

- Son estructuras de datos muy importantes en computación
- Podemos realizar búsquedas muy rápidas de elementos (tiene mucho que ver con la búsqueda binaria).
- Para buscar un elemento:
  - Se compara el elemento buscado con el nodo actual
    - Si es menor, se recorre el subárbol izquierdo
    - Si es mayor, se recorre el subárbol derecho
  - La búsqueda termina en dos casos
    - Cuando encontramos el elemento buscado
    - Cuando llegamos a un nodo hoja y no encontramos el elemento buscado
- La cantidad de comparaciones es proporcional a la altura del árbol.
- Si los elementos están bien distribuidos en el árbol, la altura será log n. Por lo tanto la complejidad de la búsqueda será de O(log n)

Y cómo los implementamos?




```
typedef int TInfo;

typedef struct node{
    TInfo info;
    struct node *left, *right;
}TNode;

typedef struct tree{
    TNode* root;
}TBinaryTree;
```

Operaciones sobre ABB

- Hay que tener en cuenta el orden de los elementos
- Por ejemplo:
  - Cuando se inserta un elemento nuevo, hay que buscar la posición correcta dónde insertar el elemento.
- Ejercicio:
  - Insertar los siguientes elementos en un ABB vacío

K, E, C, P, G, F, A, T, M, U, V, X, Z

Operaciones básicas

- Buscar
- Insertar un nuevo elemento
- Eliminar un elemento

Ejercicios

- Dado un ABB, cuente el número de nodos del árbol
- Dado un ABB, calcule la máxima profundidad del árbol; es decir, el número de nodos en el camino más largo desde la raíz hasta la hoja más profunda.
- Dado un árbol binario, implemente una función que indique si es un ABB o no.