4.2.2 Eliminación en Arboles Binarios de Búsqueda

Es evidente que es más complejo eliminar un nodo con 2 hijos que una hoja

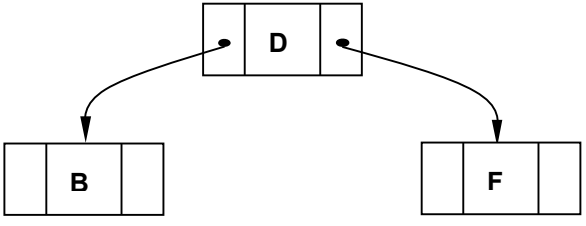
Por lo que conviene desglosar el problema en 3 casos:

1. **Eliminar una hoja**
2. **Eliminar un nodo que tiene solo un hijo**
3. **Eliminar un nodo que tiene 2 hijos**

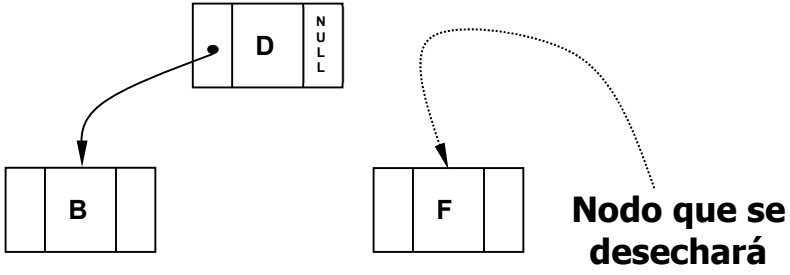
1. Supresión de una hoja

- **Procedimiento muy simple.**
- **Hay que colocar el apuntador de su padre en FIN DE LISTA.**
- **Deshacerse del nodo ya eliminado.**

Estado original de cierto <u>ARBOL</u> o <u>SUB-ARBOL</u>:



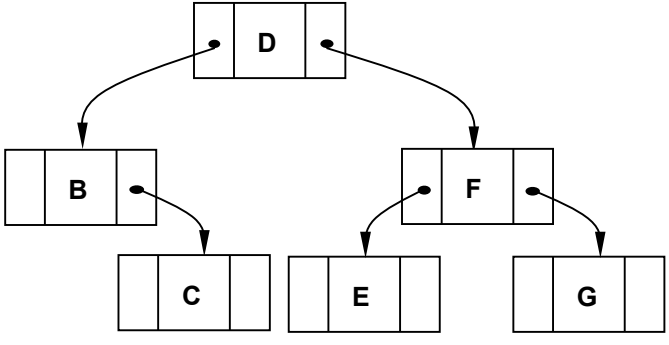
Al eliminar la F, el sub-árbol resultante será:


D
NULL
B
F
Nodo que se desechará

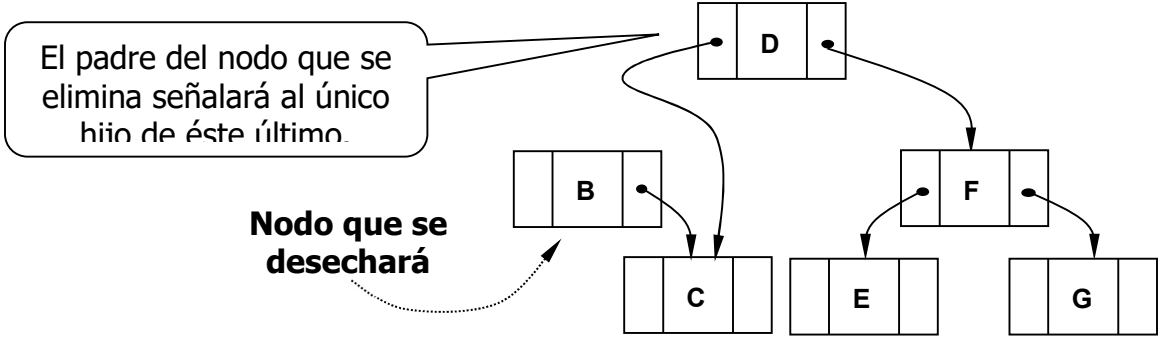
2. Supresión de un nodo con un hijo

No se puede usar el algoritmo anterior porque se perderían los descendientes del nodo que se elimina.

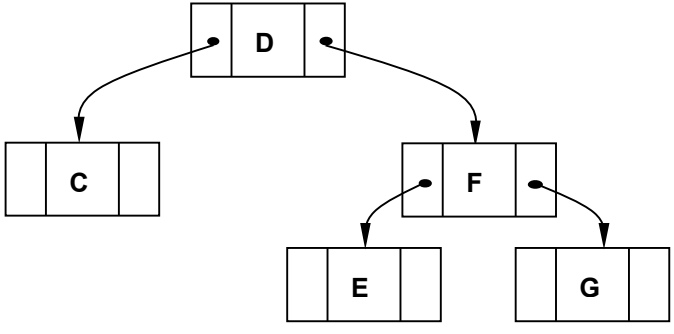
Estado original de cierto <u>ARBOL</u> o <u>SUB-ARBOL</u>:



¿Cómo quedará el árbol al eliminar la B?



El sub-árbol con la raíz C sube un nivel.



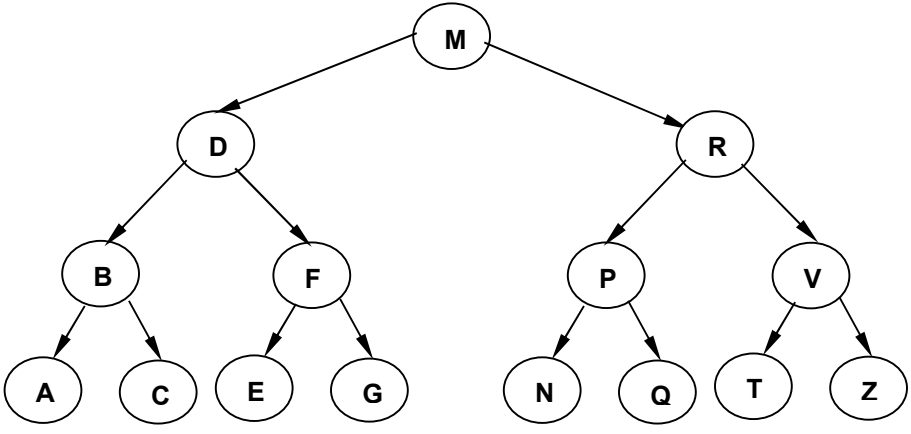
3. Supresión de un nodo con dos hijos

- **Caso más complejo que el anterior porque no se puede hacer que un nodo apunte a otros dos por la izquierda o la derecha.**
- **El método más común para resolver este caso es sustituir información.**
- **Se reemplazará el dato que se quiere eliminar con el dato más cercano del subárbol izquierdo (equivalente al predecesor inmediato).**

¿Cómo se encuentra el predecesor inmediato?

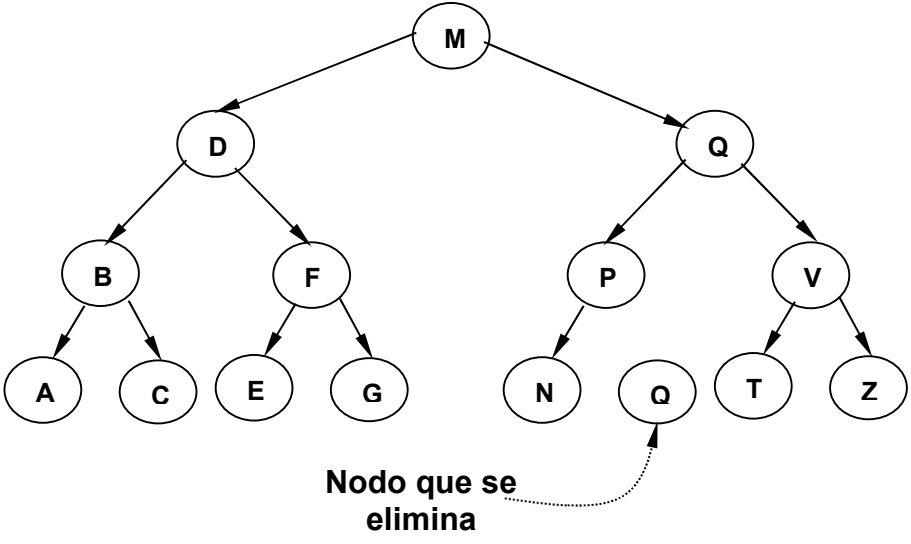
Se viaja (a partir del nodo a eliminar) una vez a la izquierda y luego tantas veces como se pueda hacia la derecha.

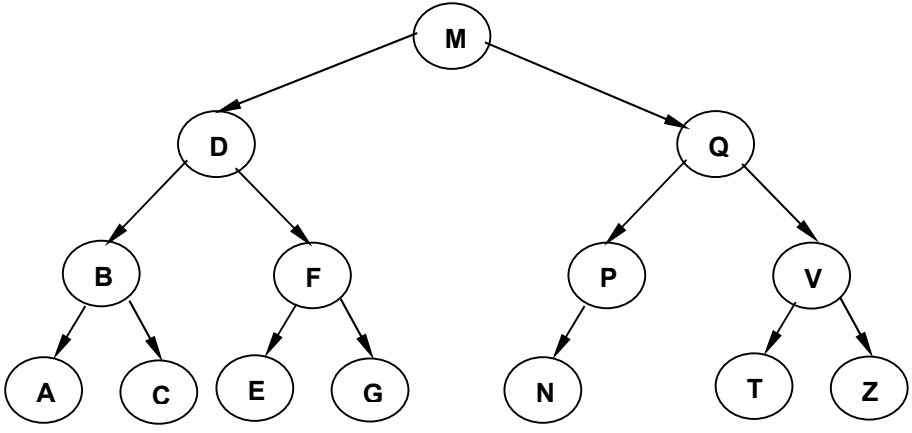
Ya encontrado el dato sustituto, se coloca en el lugar del nodo a eliminar, y el nodo del dato sustituto se elimina (solo tendrá un hijo como máximo).



Ejemplo: Eliminar la R

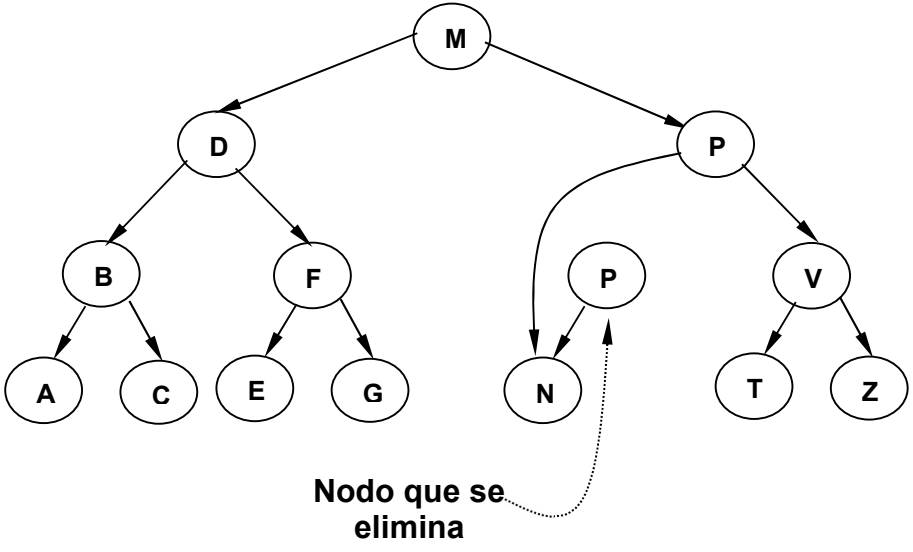
Se reemplazará la información del nodo que contiene la **R** por el predecesor inmediato, es decir la **Q**, luego el nodo que contenía la Q deberá ser eliminado.

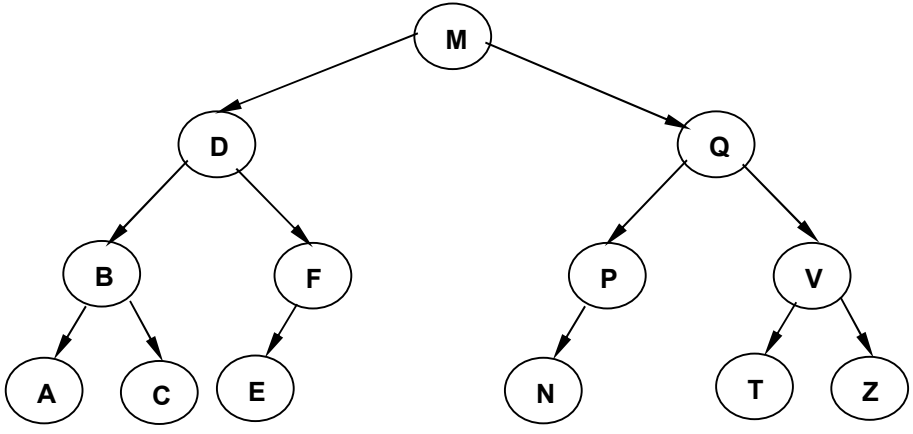

M
D
Q
B
F
P
V
A
C
E
G
N
Q
T
Z
Nodo que se elimina



Ejemplo: Eliminar la Q

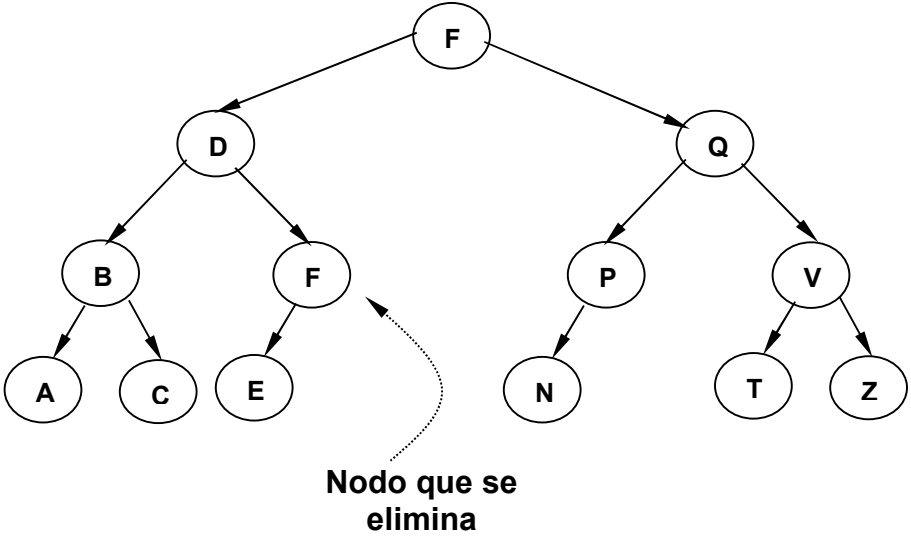
Se reemplazará la información del nodo que contiene la **Q** por el predecesor inmediato, es decir la **P**, luego el nodo que **contenía** la P deberá ser eliminado.

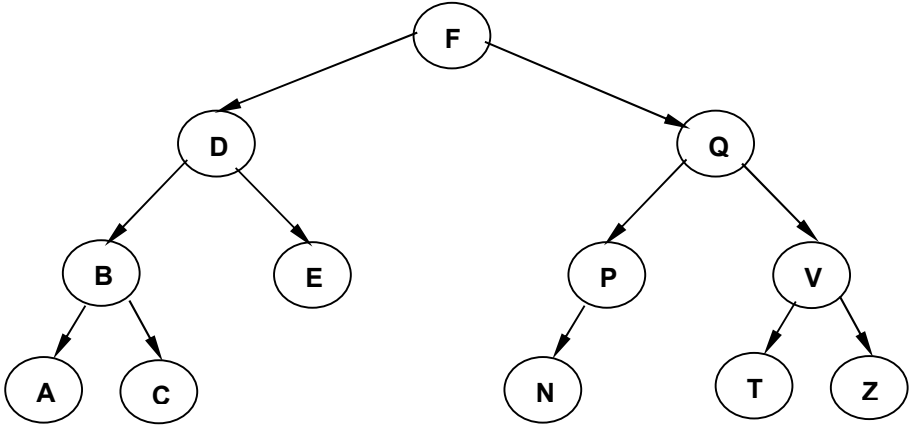

M
D
P
B
F
P
V
A
C
E
G
N
T
Z
Nodo que se elimina



Ejemplo: Eliminar la M

Se reemplazará la información del nodo que contiene la **M** por el predecesor inmediato, es decir la **F**, luego el nodo que **contenía** la **F** deberá ser eliminado.


F
D
Q
B
F
P
V
A
C
E
N
T
Z
Nodo que se elimina


F
D
Q
B
E
P
V
A
C
N
T
Z

Algoritmos:

P = dirección del nodo a eliminar.
PADRE = dirección del padre del nodo a eliminar.

PROCEDIMIENTO SUPRIME_NODO(P)

```
SI IZQ(P)=FIN Y DER(P)=FIN
        EJECUTA PROCEDIMIENTO CASO_1
DE LO CONTRARIO
        SI IZQ(P)<>FIN Y DER(P)<>FIN
                EJECUTA PROCEDIMIENTO CASO_3
        DE LO CONTRARIO
                EJECUTA PROCEDIMIENTO CASO_2
        FIN SI
FIN SI
```

Una vez ejecutado el algoritmo anterior, se sabrá cuantos hijos tiene el nodo a eliminar ....

CASO 1

Hay que determinar si el nodo a eliminar es hijo izquierdo o derecho de su padre.

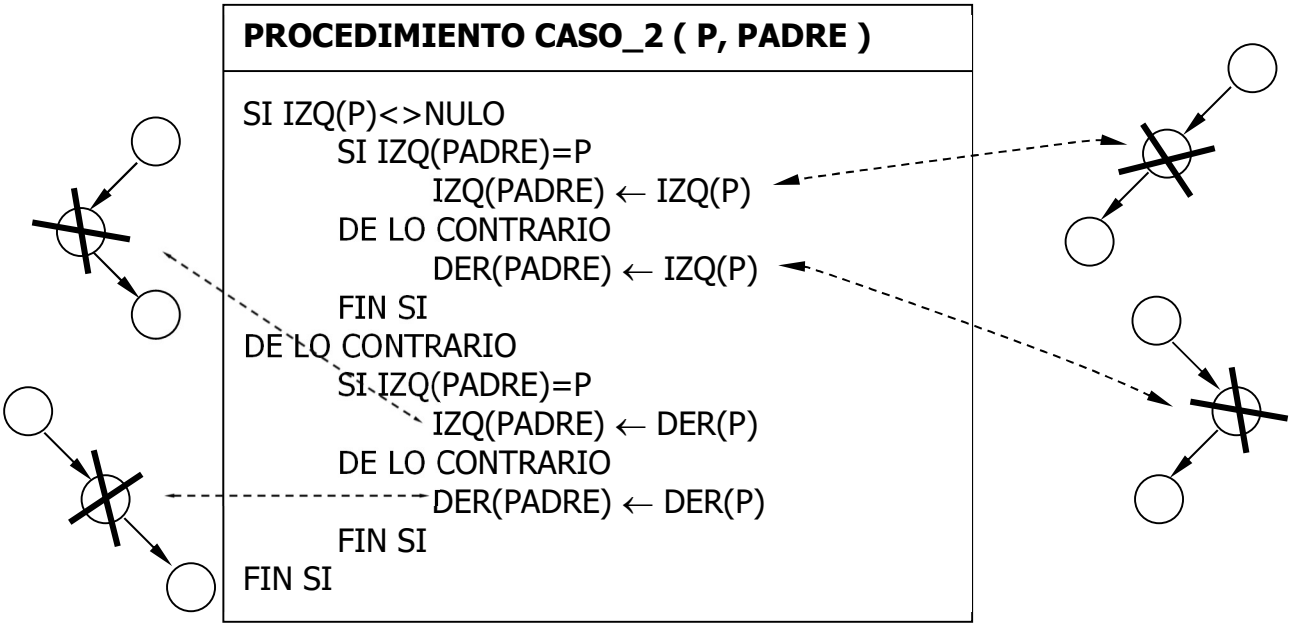
PROCEDIMIENTO CASO_1(P, PADRE)

```
SI IZQ(PADRE)=P
        IZQ(PADRE) ← NULO
DE LO CONTRARIO
        DER(PADRE) ← NULO
FIN SI
```

Adecue este algoritmo para verificar si el nodo a eliminar es la RAIZ, porque en tal caso no hay un nodo PADRE **(RAIZ ← NULO)**

CASO 2

Hay que determinar si el nodo a eliminar es hijo izquierdo o derecho de su padre y si el único hijo que tiene es izquierdo o derecho



```
PROCEDIMIENTO CASO_2 ( P, PADRE )

SI IZQ(P)<>NULO
        SI IZQ(PADRE)=P
                IZQ(PADRE) ← IZQ(P)
        DE LO CONTRARIO
                DER(PADRE) ← IZQ(P)
        FIN SI
DE LO CONTRARIO
        SI IZQ(PADRE)=P
                IZQ(PADRE) ← DER(P)
        DE LO CONTRARIO
                DER(PADRE) ← DER(P)
        FIN SI
FIN SI
```

Se debe adecuar este algoritmo para verificar si el nodo a eliminar es la RAIZ, porque en tal caso no hay un nodo PADRE.

CASO 3

P_SUST = Dirección del nodo sustituto
PADRE_SUST = Dirección del Padre del Nodo Sustituto.

PROCEDIMIENTO CASO_3(P)

```
PADRE_SUST ← P
P_SUST ← IZQ(P)
MIENTRAS DER(P_SUST) <> NULO
        PADRE_SUST ← P_SUST
        P_SUST ← DER(P_SUST)
FIN MIENTRAS

INFO(P) ← INFO(P_SUST)

SI PADRE_SUST = P
        IZQ(PADRE_SUST) ← IZQ(P_SUST)
DE LO CONTRARIO
        DER(PADRE_SUST) ← IZQ(P_SUST)
FIN SI
```

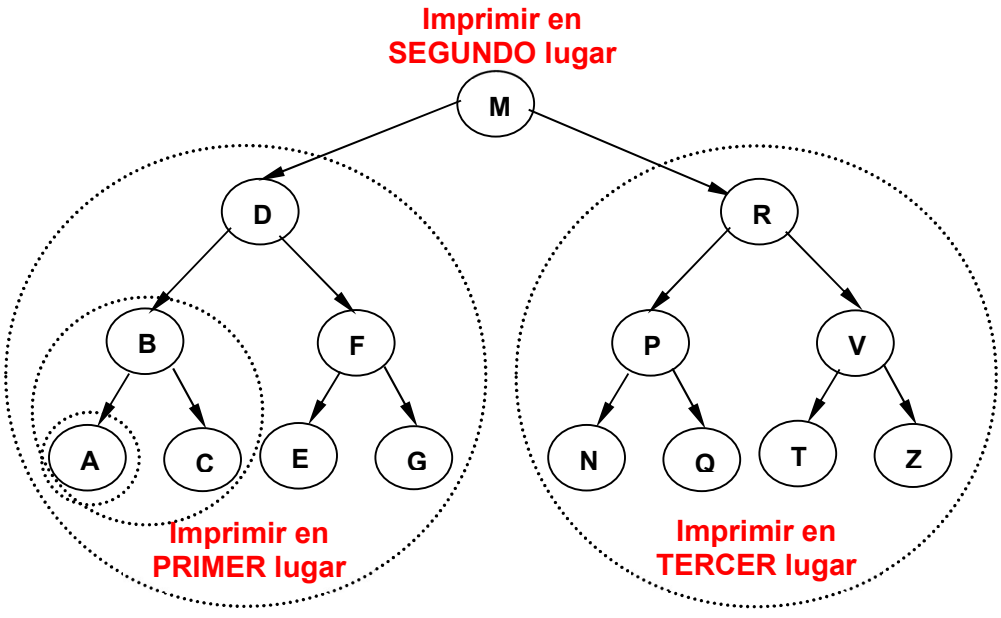
4.2.3 Recorridos

Recorrer significa "visitar todos los nodos del árbol".

En las listas lineales, solo hay un camino. En los Árboles Binarios no.

Estando en un nodo ¿hacia donde ir? …. ¿izquierda ... derecha?

Supongamos que se quiere imprimir los datos de un Arbol Binario de Búsqueda ordenados de menor a mayor:



Para imprimir cada sub-árbol se repetirá el mismo proceso.

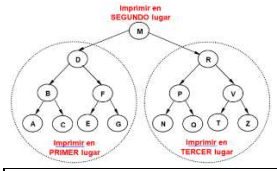
Este recorrido, que en este caso se sugiere su uso para impresión, se conoce como **INORDEN** (IN=entre, es decir, la raíz ENTRE ambos subárboles).

La definición recursiva de la impresión en Inorden de un árbol con raíz en el **nodo P** sería la siguiente:

Imprime en Inorden los datos del sub-árbol con raíz en el **hijo izquierdo de P**, luego imprime el dato del **nodo P** y finalmente, en Inorden, los datos del sub-árbol con raíz en el **hijo derecho de P**.

La tabla de la página siguiente describe una metodología para de conocer el resultado de la impresión Recursiva en Inorden.

Los recuadros vacíos muestran lugares reservados para datos que se sabe estarán allí pero no se conoce aún el orden.



M

D M

B D M

A B D M

A B C D M

A B C D F M

A B C D E F M

A B C D E F G M

A B C D E F G M R

Algoritmo Recursivo:

```
PROCEDIMIENTO INORDEN(P)

SI P<>NULO
        EJECUTAR INORDEN(IZQ(P))
        IMPRIMIR INFO(P)
        EJECUTAR INORDEN(DER(P))
FIN SI
```

Llamada Inicial: **INORDEN(RAIZ).**

Algoritmo Iterativo:

```
P ← RAIZ
HASTA P=FIN Y PILAVACIA
        MIENTRAS P<>FIN
                PUSH(P)
                P ← IZQ(P)
        FIN MIENTRAS
        SI NO PILAVACIA
                P ← POP
                IMPRIME INFO(P)
                P ← DER(P)
        FIN SI
FIN HASTA
```

PROCEDIMIENTO PREORDEN(P)

```
SI P<>FIN
        IMPRIMIR INFO(P)
        EJECUTAR PREORDEN(IZQ(P))
        EJECUTAR PREORDEN(DER(P))
FIN SI
```

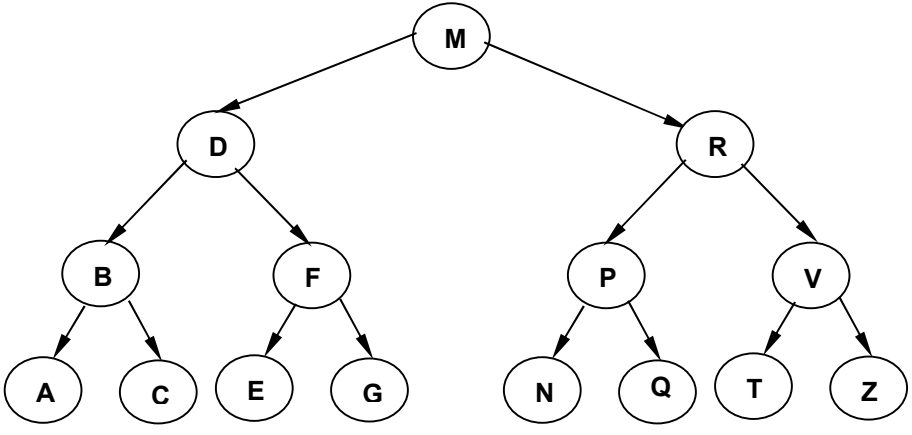
Llamada Inicial: PREORDEN(RAIZ).

PROCEDIMIENTO POSTORDEN(P)

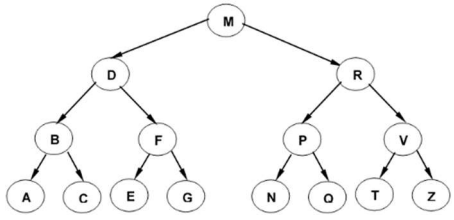
```
SI P<>FIN
        EJECUTAR POSTORDEN(IZQ(P))
        EJECUTAR POSTORDEN(DER(P))
        IMPRIMIR INFO(P)
FIN SI
```

Llamada Inicial: POSTORDEN(RAIZ).

Ejemplo: ¿Cuál será la impresión obtenida para cada uno de los 3 recorridos básicos para el árbol siguiente?



INORDEN: A B C D E F G M N P Q R T V Z
PREORDEN: M D B A C F E G R P N Q V T Z



POSTORDEN (Primero los sub-árboles y al final la raíz)**:**

														M
						D								M
		B				D								M
A	C	B				D								M

. . . . . .
. . . . . .
. . . . . .

A	C	B	E	G	F	D	N	Q	P	T	Z	V	R	M