5. Árboles (Algoritmos)

Carlos Zerón Martínez

Universidad Nacional Autónoma de México

zeron@ciencias.unam.mx

Jueves 19 de Noviembre de 2020

TDA Arbol

La especificación completa de TDA con implementación se hará hasta llegar al caso específico de árboles binarios, en los cuales cada nodo tiene a lo más dos hijos.

Datos

Un conjunto posiblemente vacío de objetos de tipo genérico. (Cada objeto se almacena en un nodo)

Operaciones

- root():NodoArbol. Devuelve el nodo raíz.
- parent(v):NodoArbol. Devuelve el nodo padre del nodo v.
- children(v):Lista$\langle NodoArbol \rangle$. Devuelve una lista de nodos que son hijos del nodo v.
- size() : $\mathbb{N} \cup \{0\}$. Devuelve el tamaño.
- isInternal(v):$\{V, F\}$. Indica si el nodo v es interno o no.
- isLeaf(v):$\{V, F\}$. Indica si el nodo v es hoja.
- isRoot(v):$\{V, F\}$. Indica si el nodo v es raíz.

Algoritmo para calcular la profundidad de un nodo

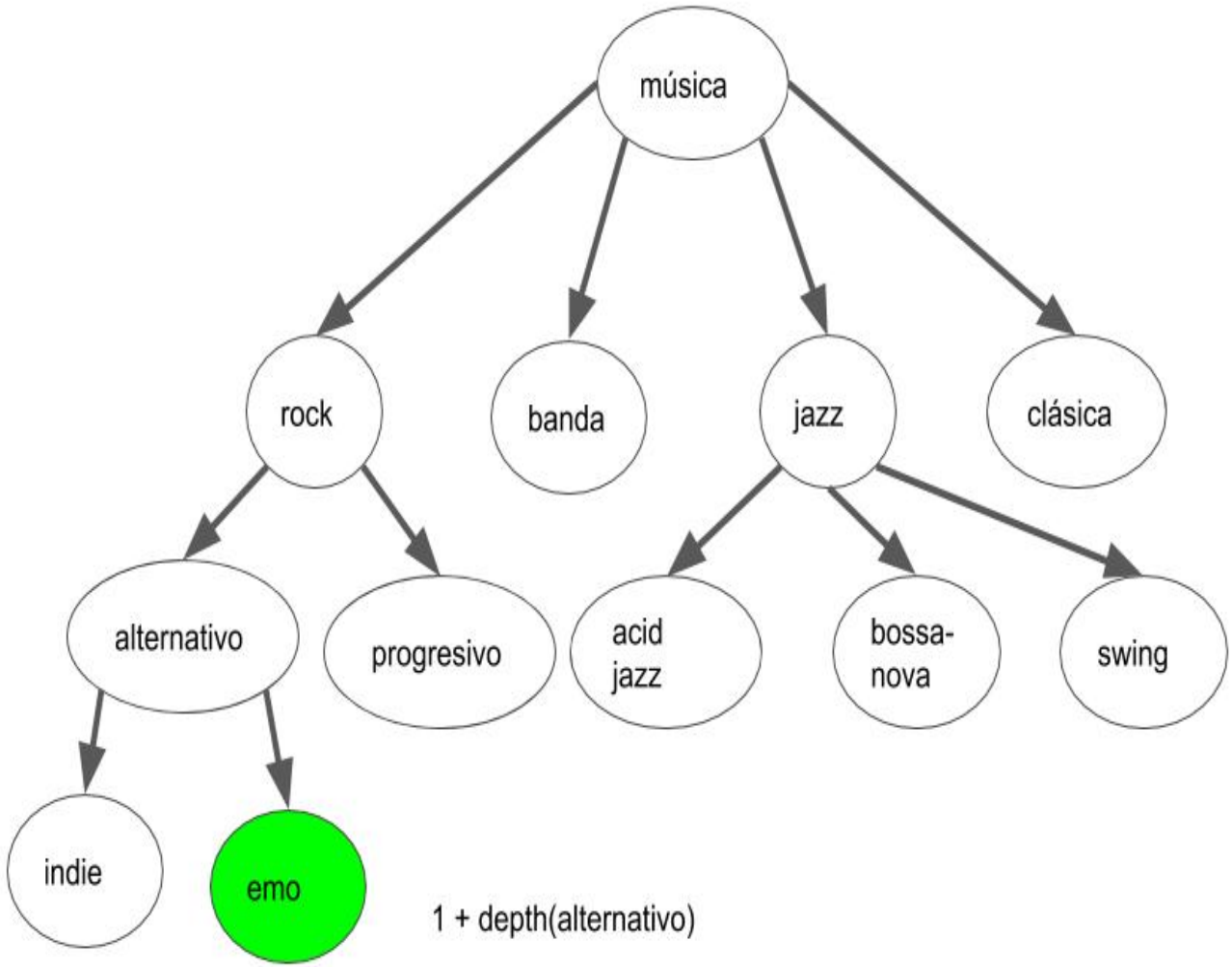
Listado 1 Algoritmo para calcular la profundidad de un nodo en un árbol

Precondiciones: v es un nodo de un árbol t.

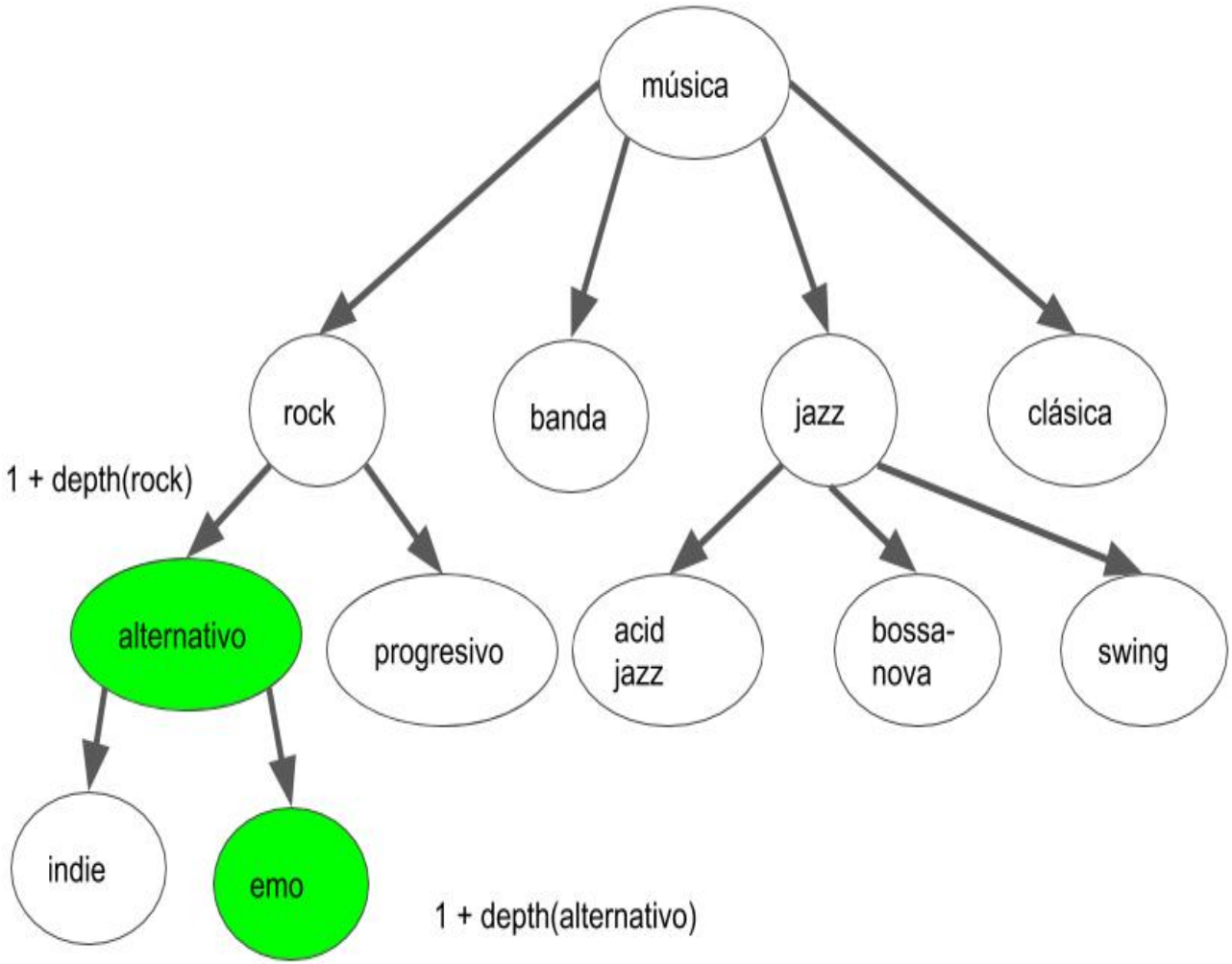
Poscondiciones: Devuelve la profundidad de v en el árbol t.

```
depth(v){
 1: if t.isRoot(v) then
 2:     return 0;
 3: else
 4:     return 1 + t.depth(t.parent(v));
 5: end if
}
```

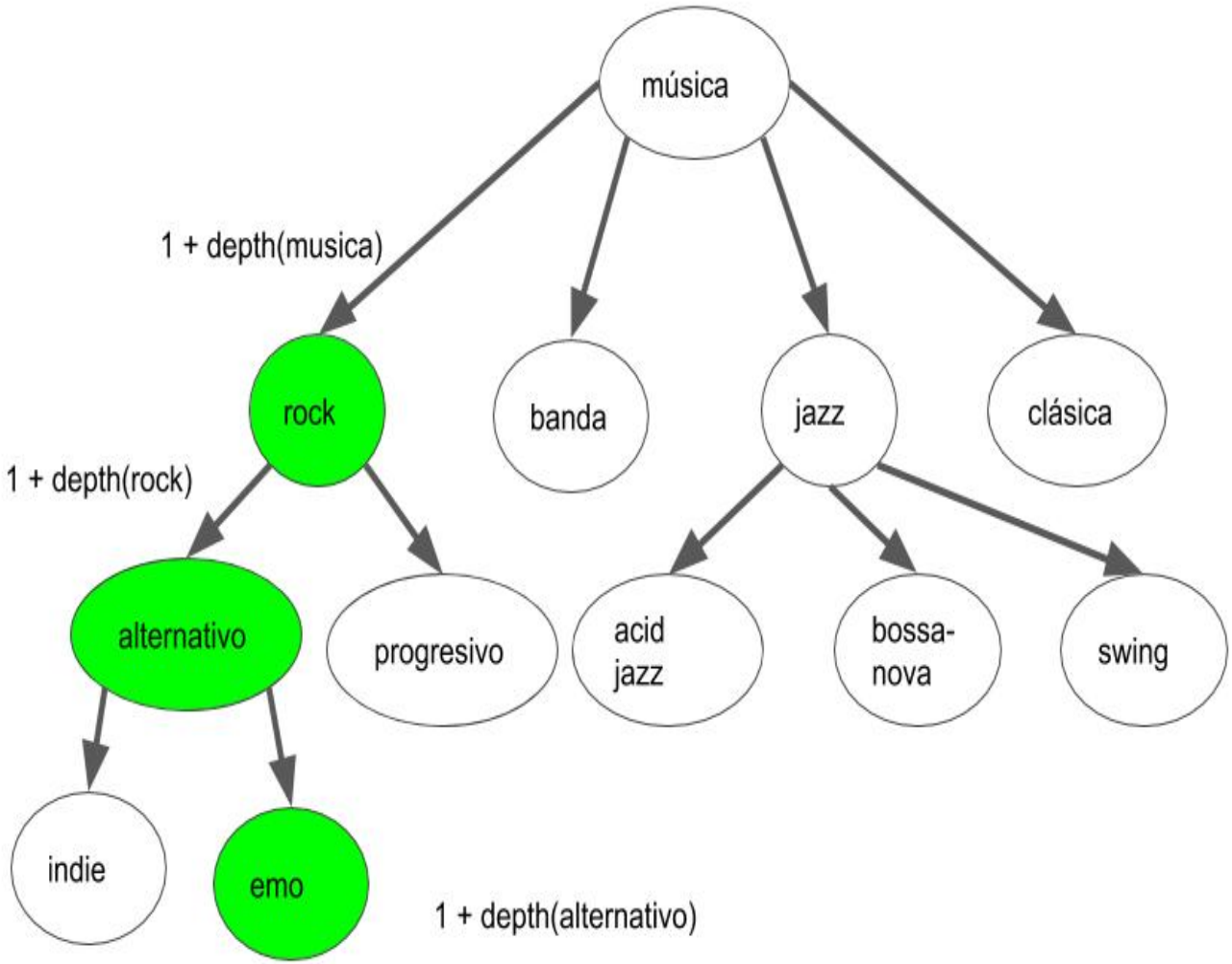
Algoritmo para calcular la profundidad de un nodo



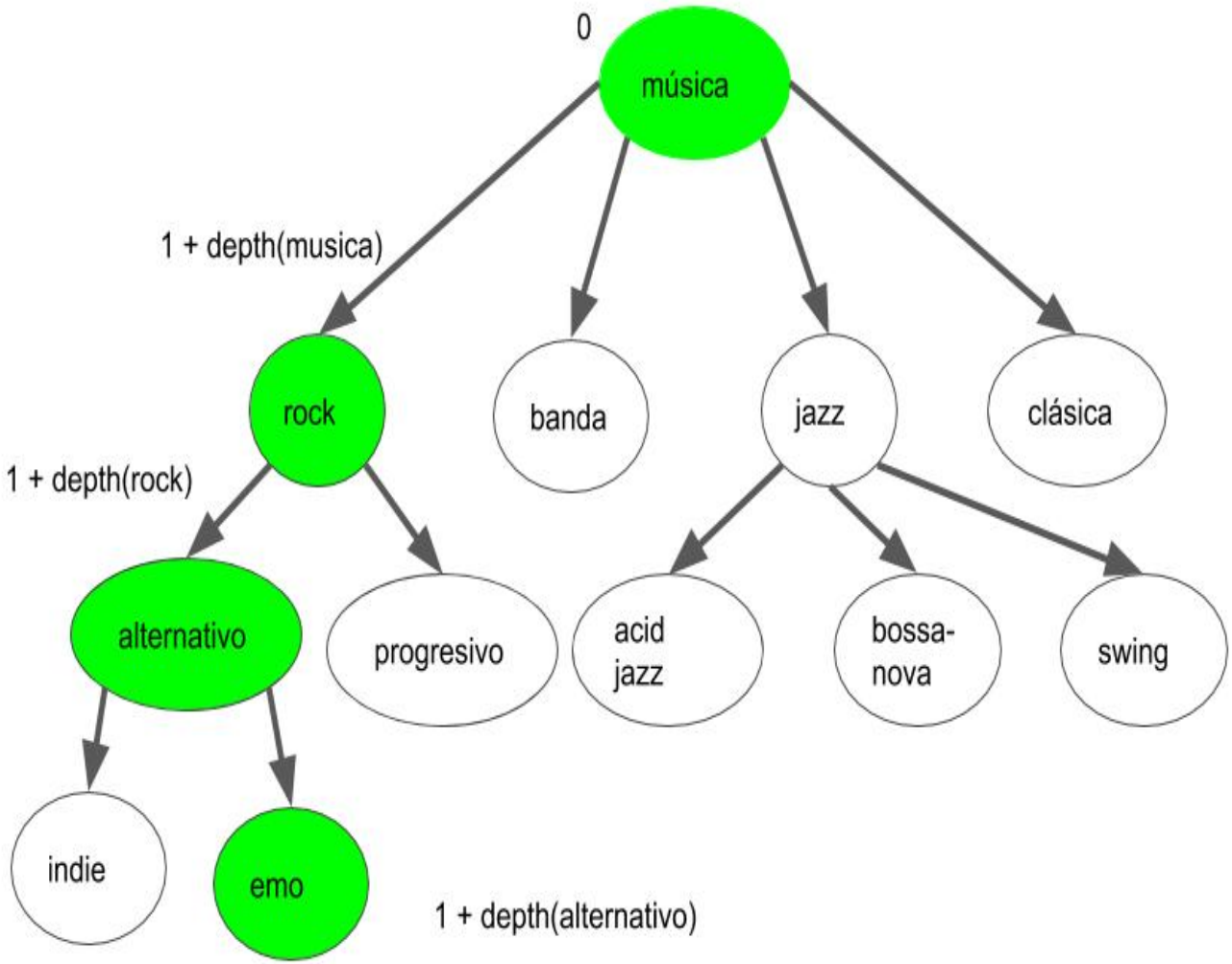
Algoritmo para calcular la profundidad de un nodo



Algoritmo para calcular la profundidad de un nodo

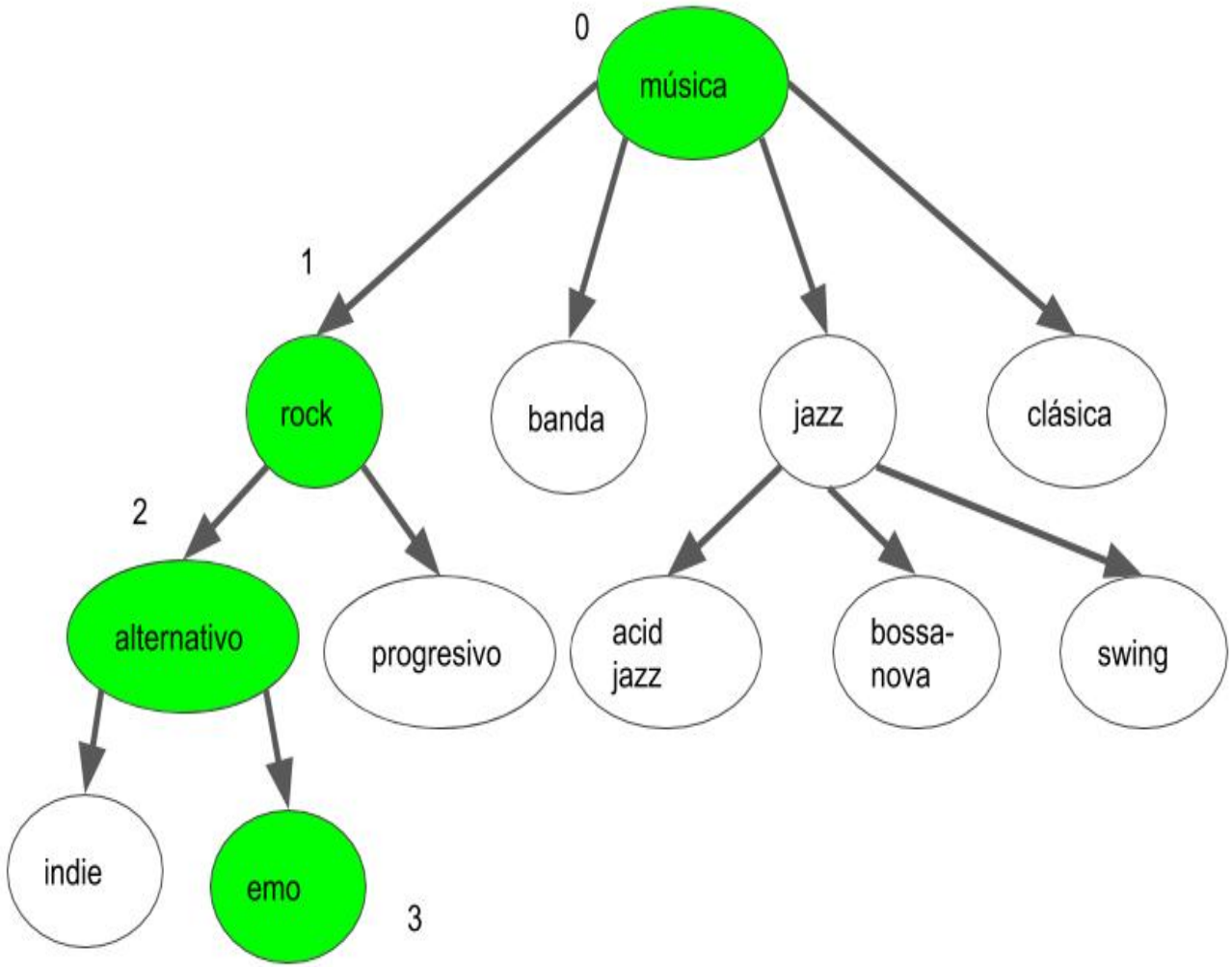


Algoritmo para calcular la profundidad de un nodo



Algoritmo para calcular la profundidad de un nodo

Regreso de las llamadas recursivas para calcular el resultado:



Algoritmo para calcular la altura de un nodo

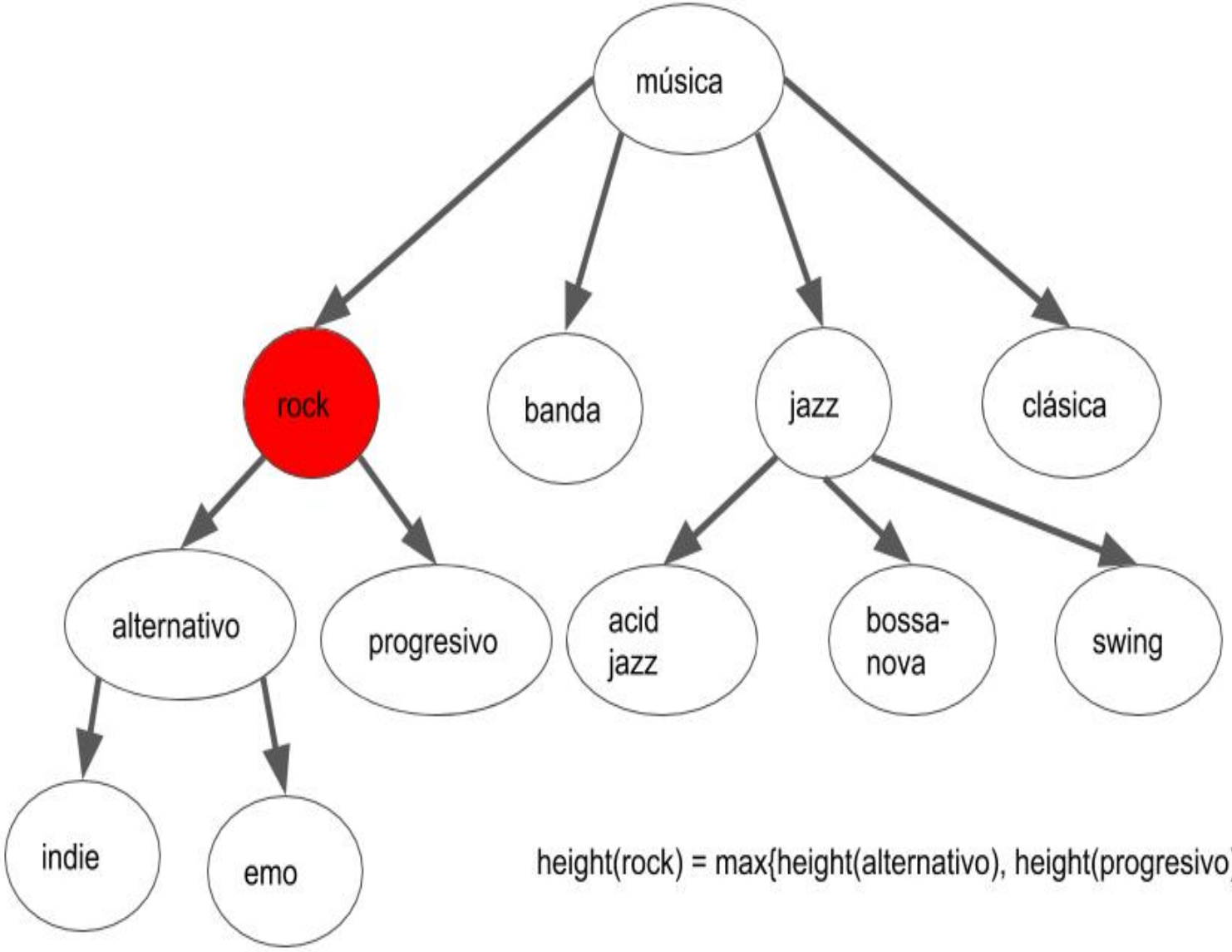
Listado 2 Algoritmo para determinar la altura de un nodo en un árbol

Precondiciones: v es un nodo de un árbol t.

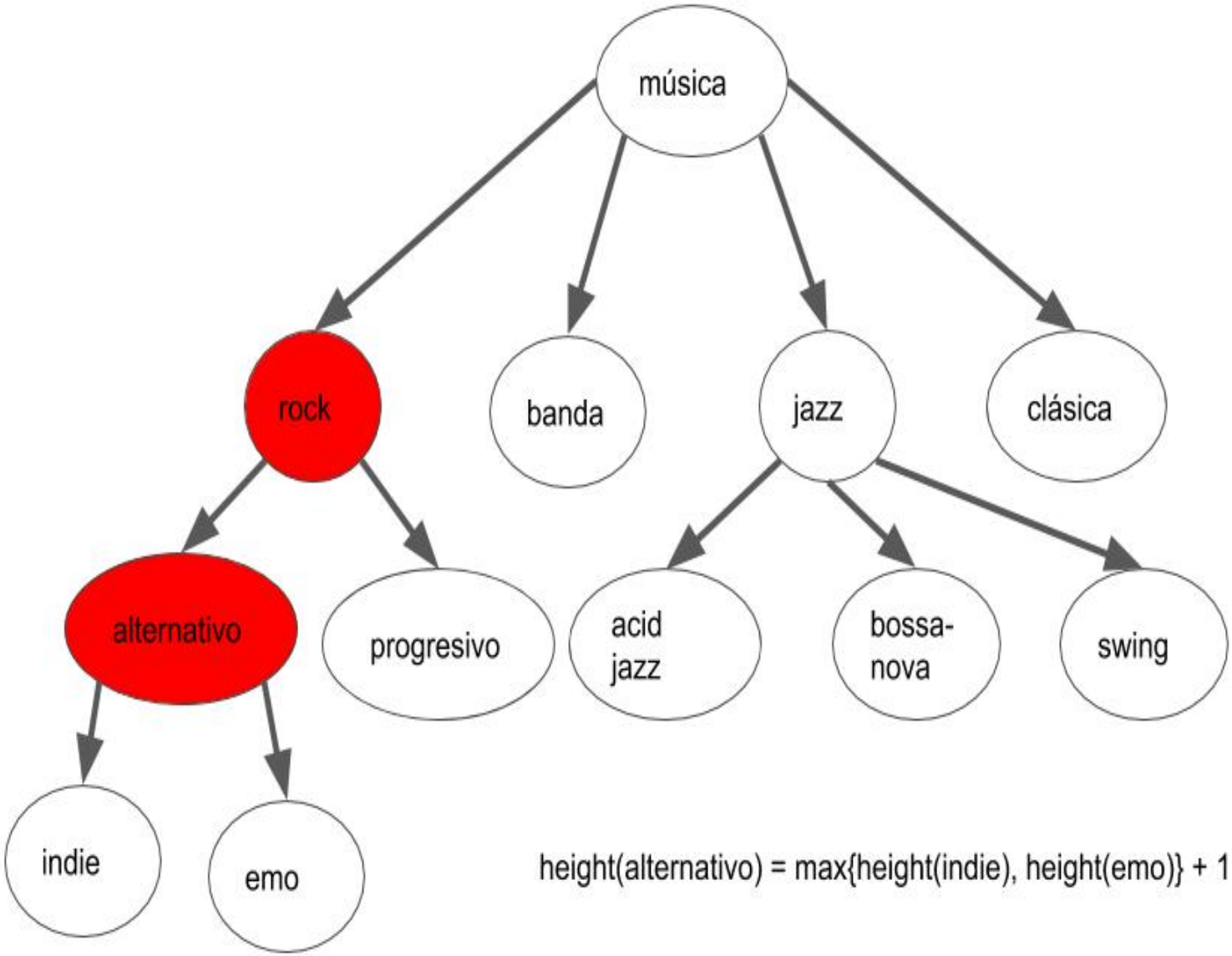
Poscondiciones: Devuelve la altura de v en el árbol t

```
height(v){
 1: if t.isLeaf(v) then
 2:     return 0;
 3: else
 4:     h := 0;
 5:     for all w ∈ children(v) do
 6:         h := max{h, height(w)};
 7:     end for
 8:     return 1 + h;
 9: end if
}
```

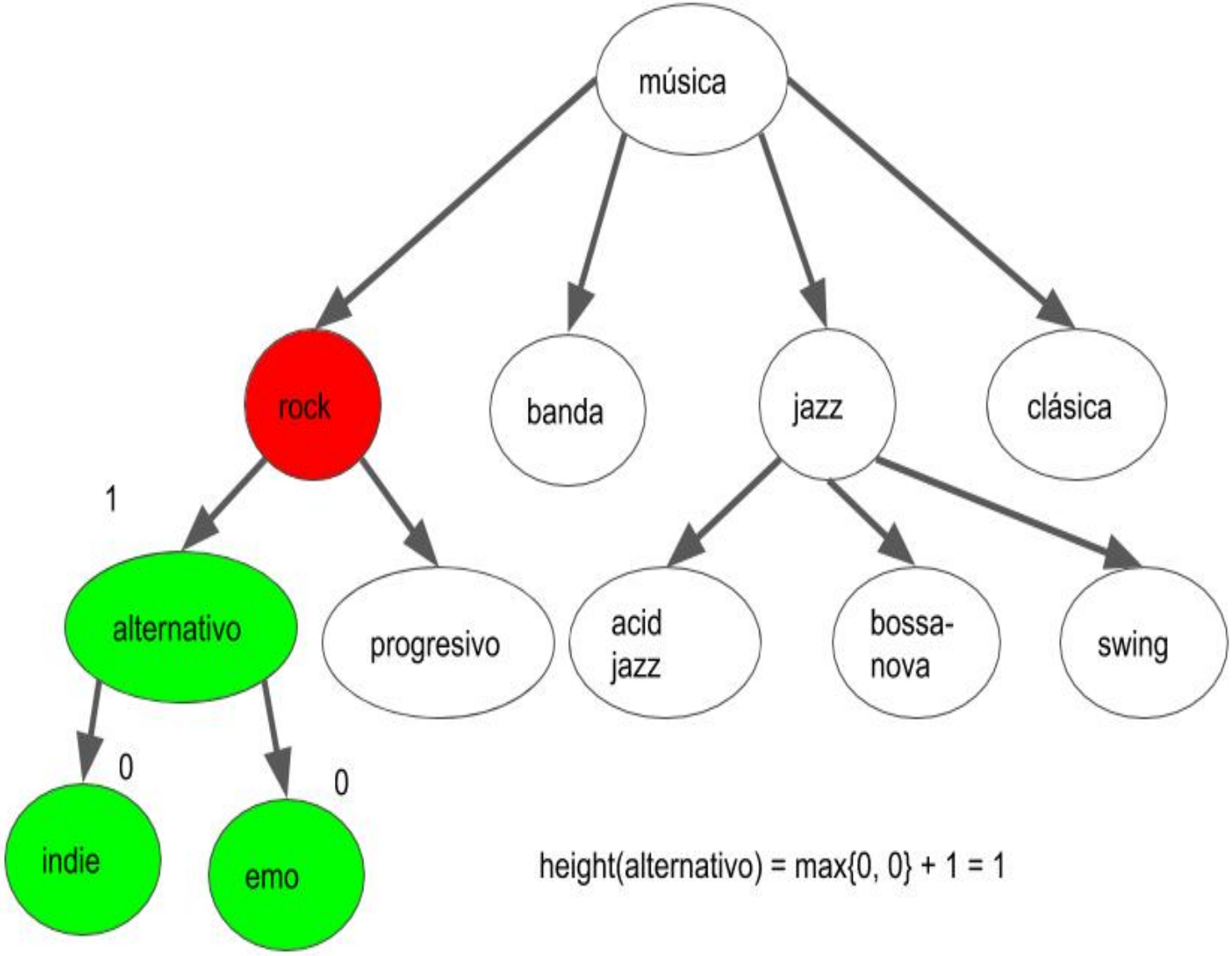
Algoritmo para calcular la altura de un nodo



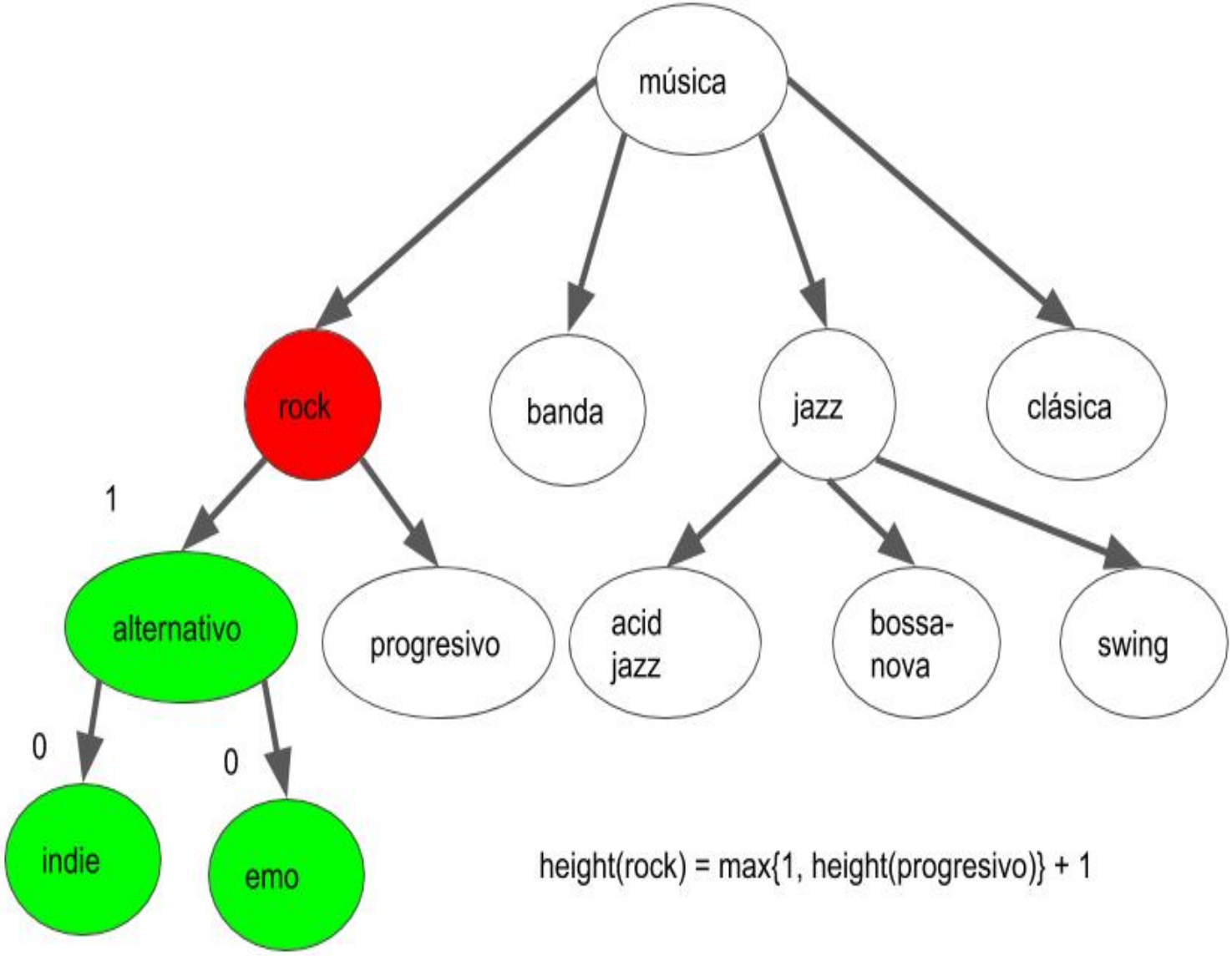
Algoritmo para calcular la altura de un nodo



Algoritmo para calcular la altura de un nodo

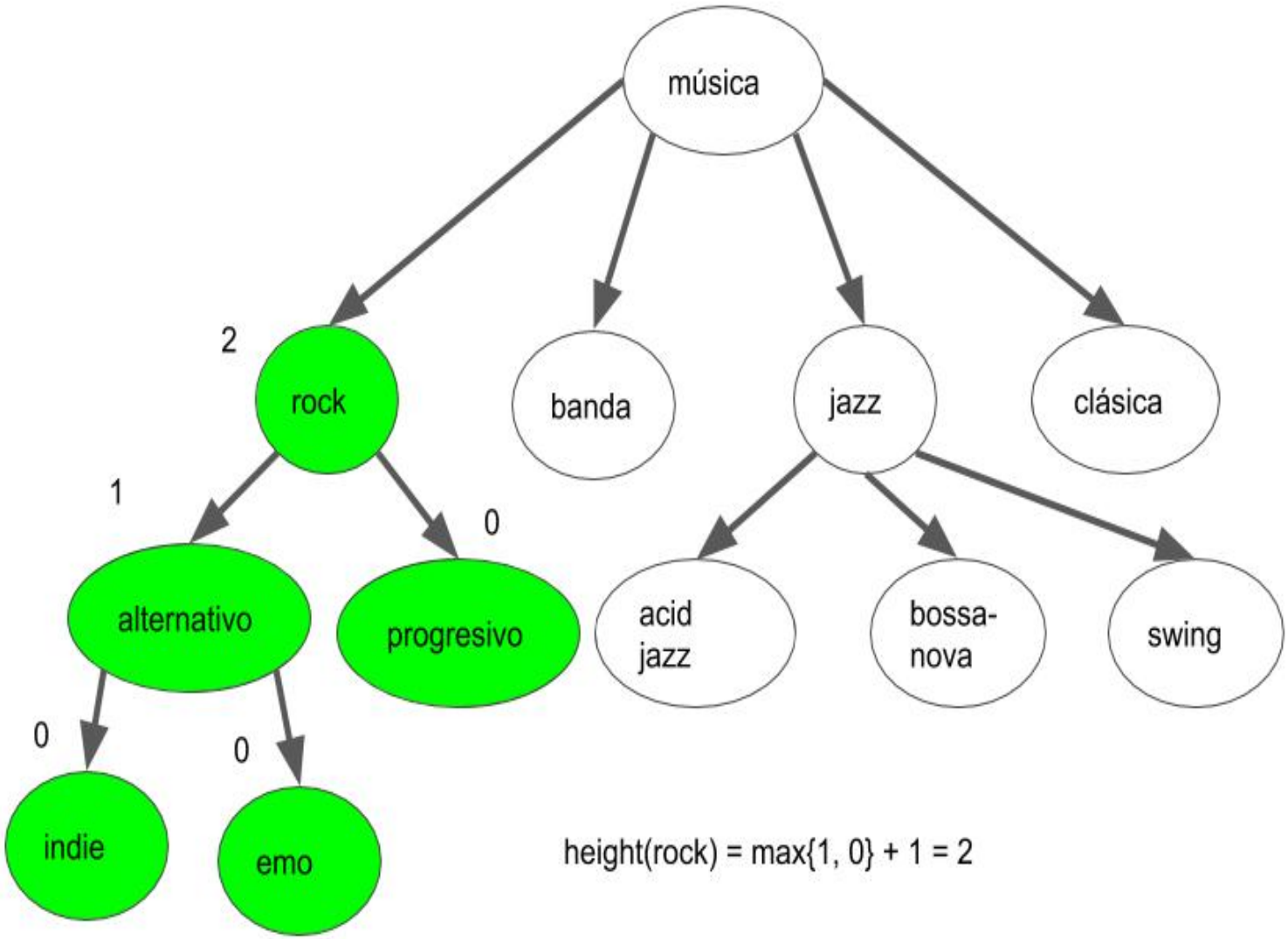


Algoritmo para calcular la altura de un nodo



Algoritmo para calcular la altura de un nodo

Se calcula la altura del nodo que contiene "rock"



Recorridos en árboles

Formas sistemáticas de acceder o visitar todos los nodos de un árbol. Los recorridos que pueden aplicarse a cualquier árbol en general:

- Recorrido en preorden
- Recorrido en postorden

Es importante establecer el concepto de posición en los hijos de cada nodo. El orden de los hijos se manifiesta en una estructura lineal (por ejemplo, una lista).

Recorrido en preorden

Se *visita* primero la raíz del árbol T y a continuación se visitan recursivamente los subárboles enraizados en sus hijos.

El recorrido se define de forma genérica: la visita puede representar algo distinto dependiendo de para qué se desee aplicar. La invocación inicial es preorder(root()).

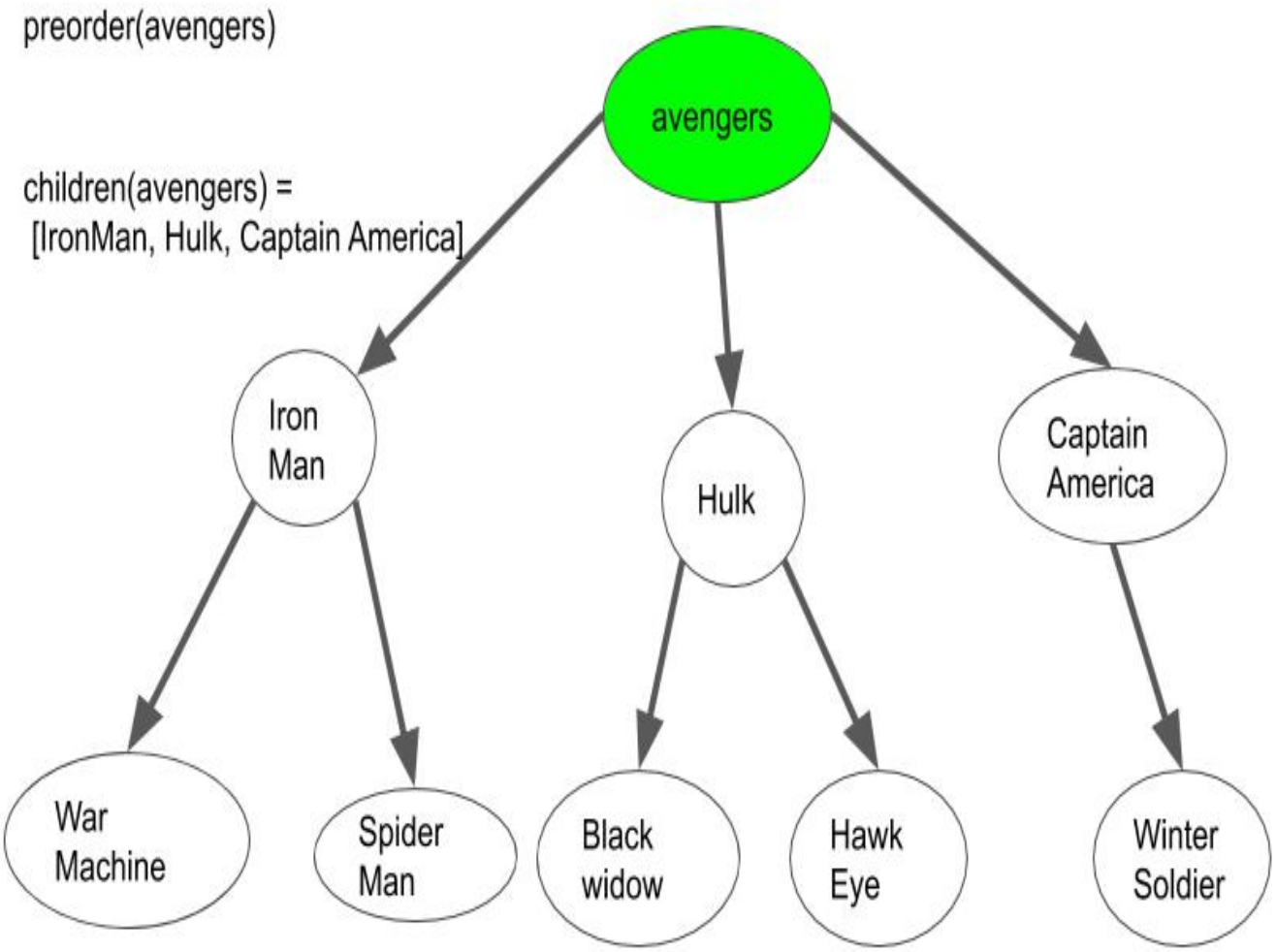
Listado 3 Algoritmo para recorrer un árbol en preorden

Precondiciones: v es un nodo de un árbol t.

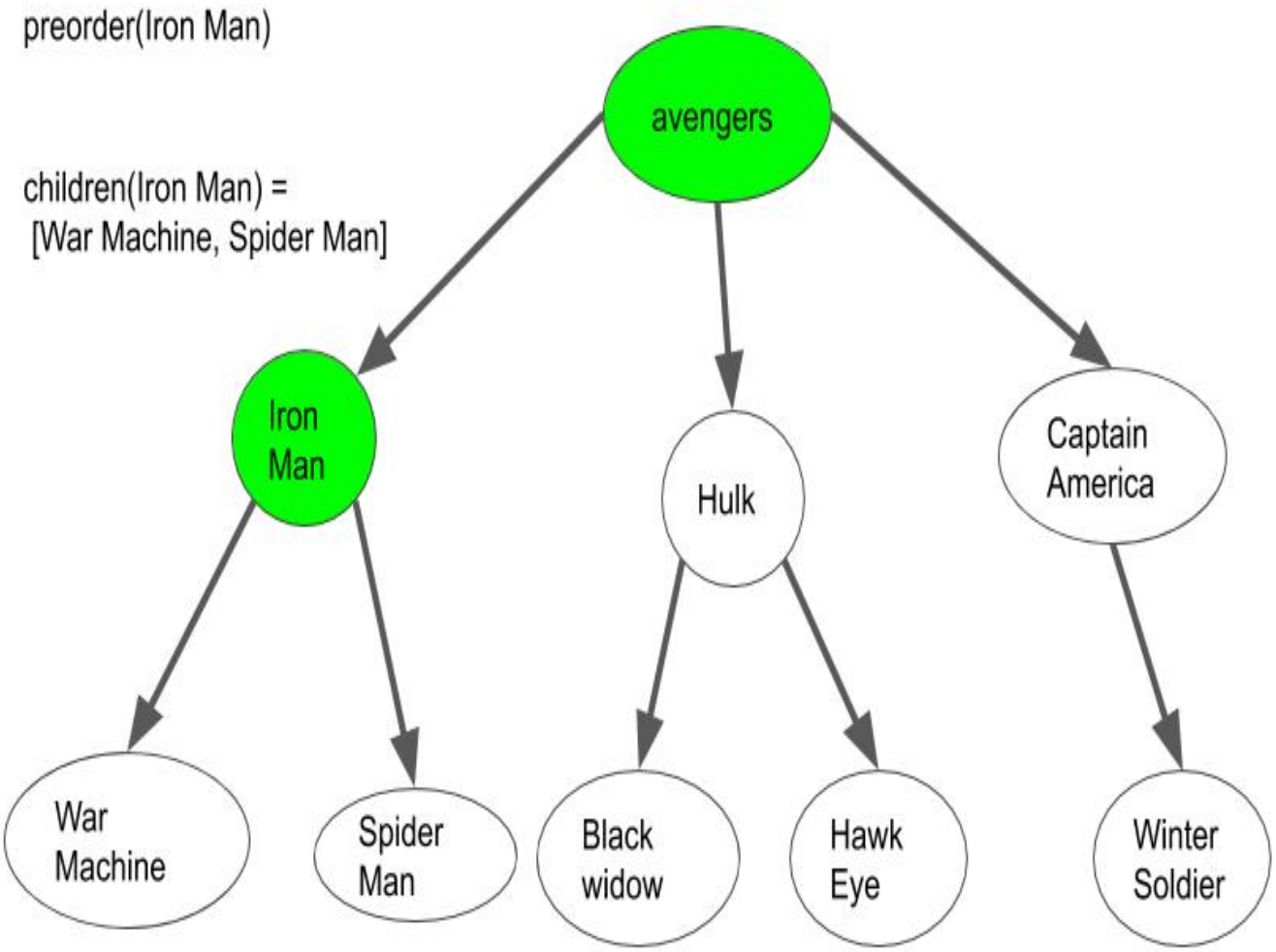
Poscondiciones: Recorre t en preorden en t de forma genérica.

```
preorder(v){
 1: visit(v);
 2: for all w ∈ children(v) do
 3:     preorder(w);
 4: end for
}
```

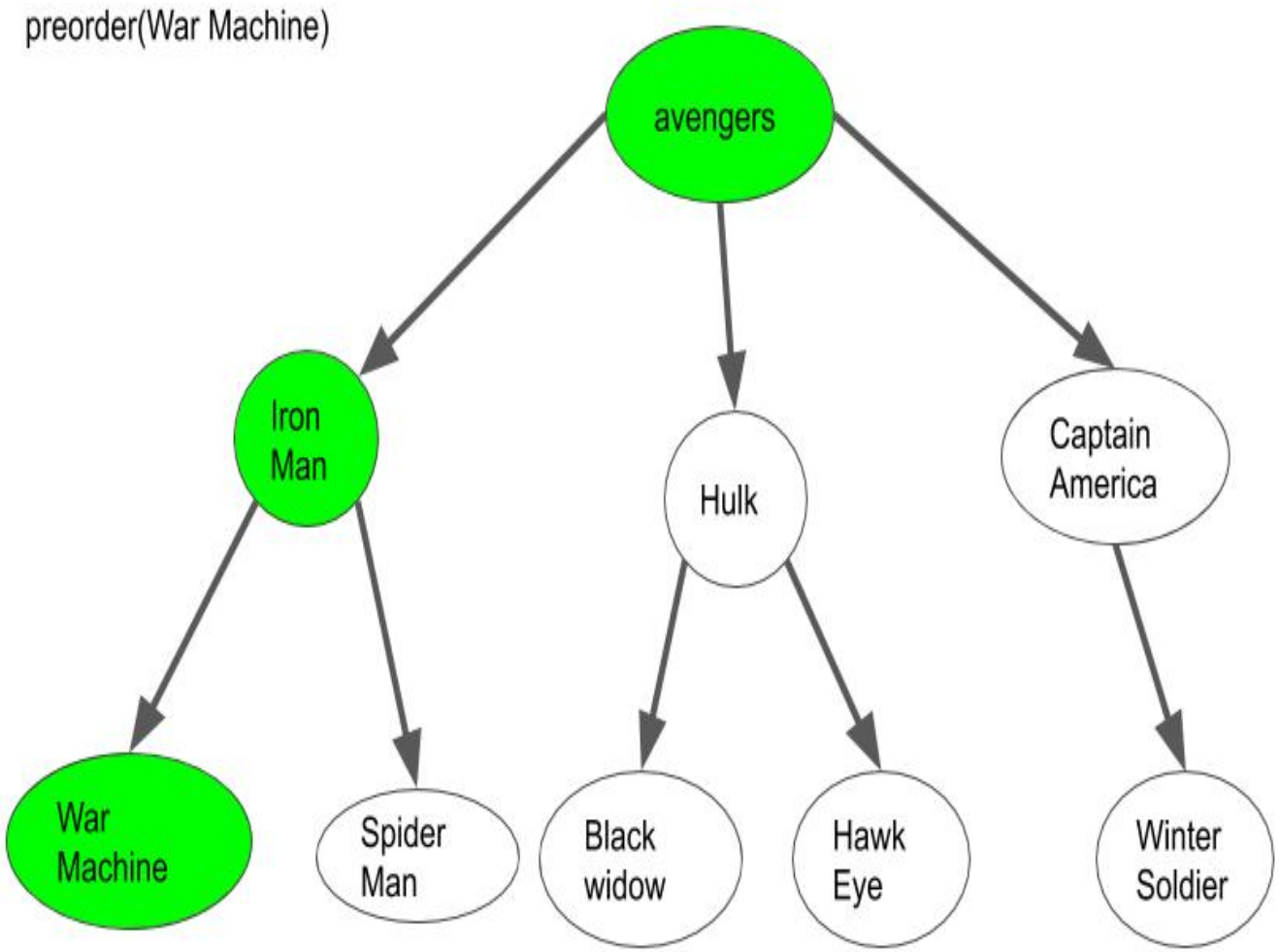
Recorrido en preorden



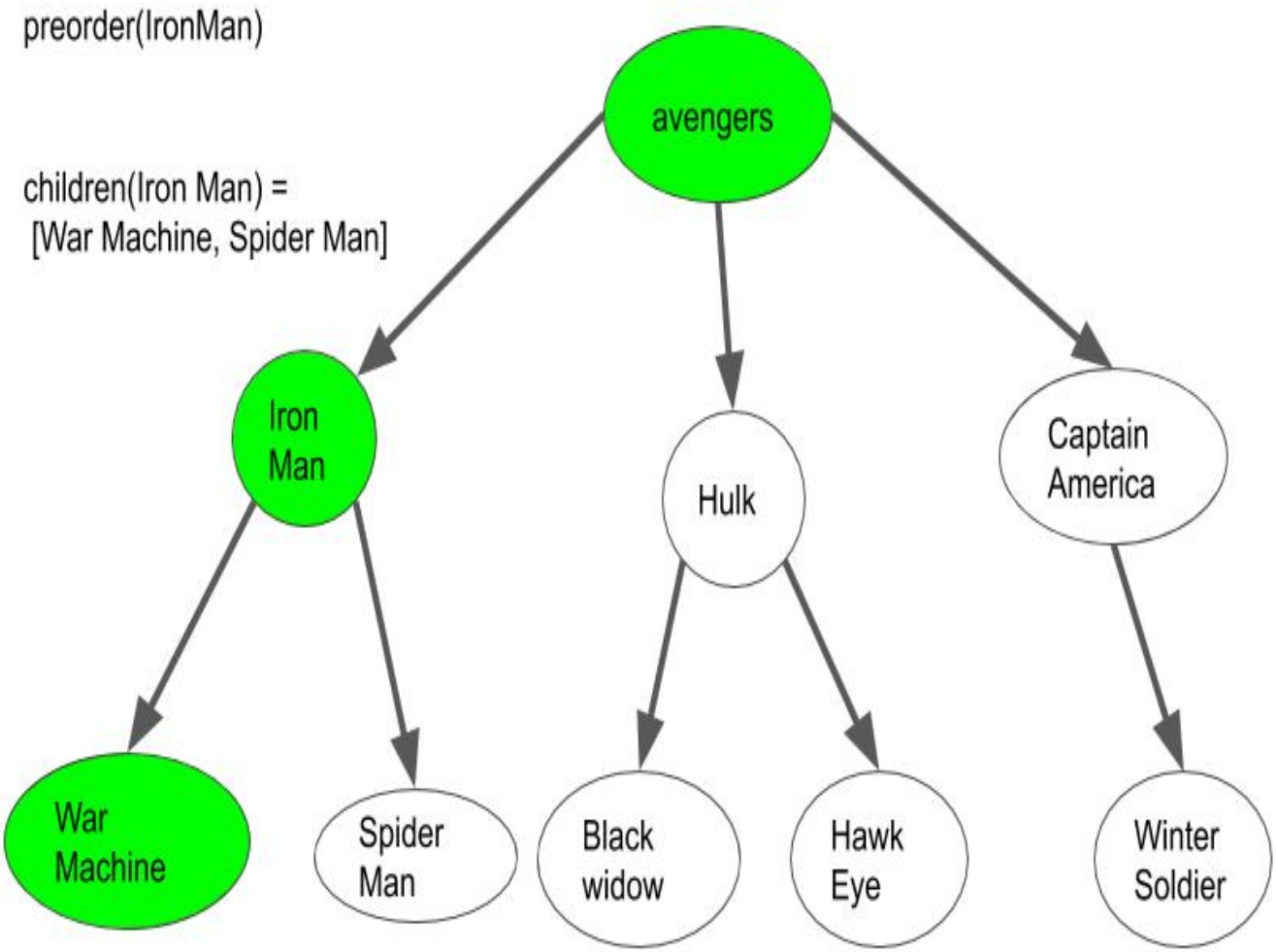
Recorrido en preorden



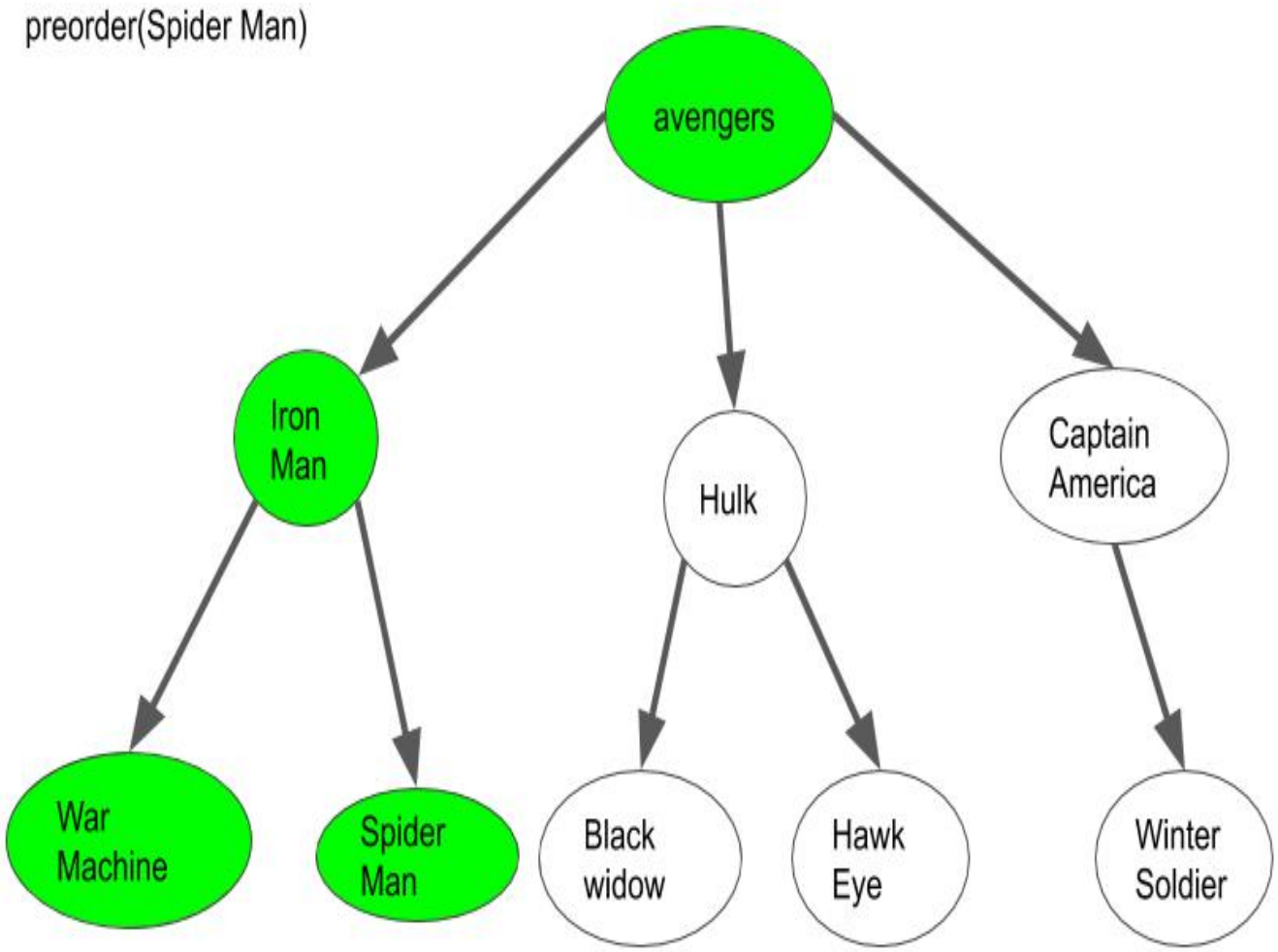
Recorrido en preorden



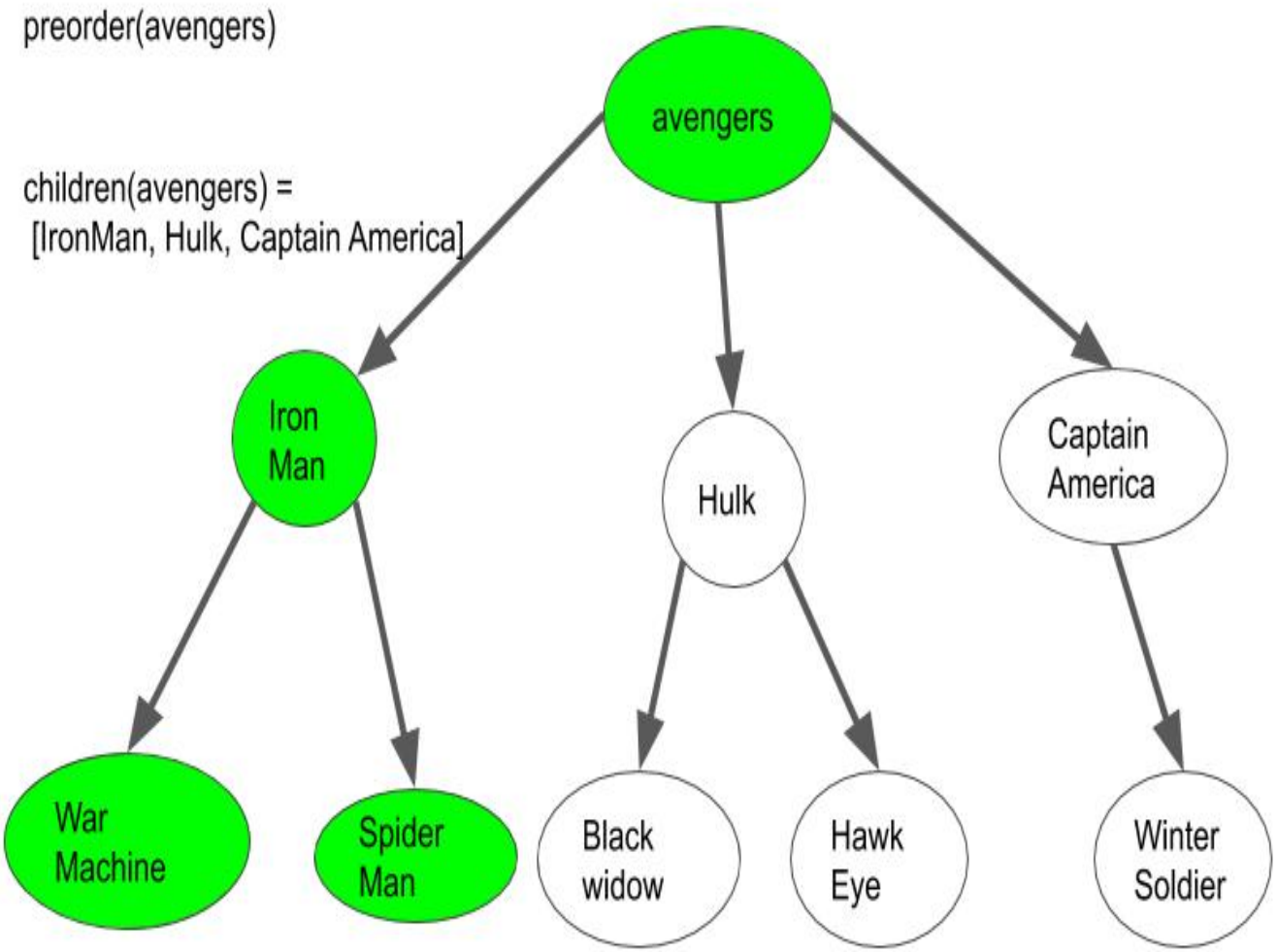
Recorrido en preorden



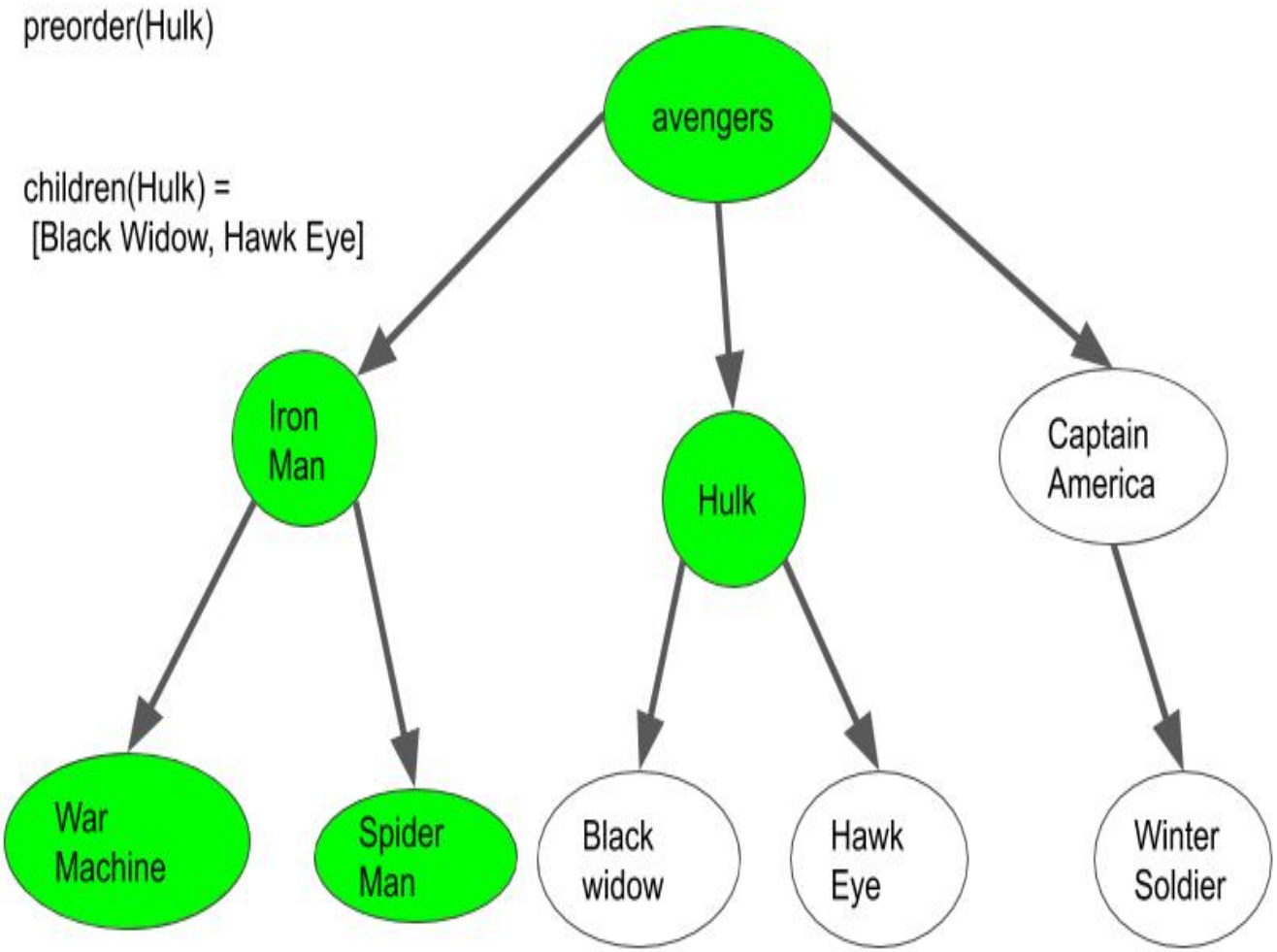
Recorrido en preorden



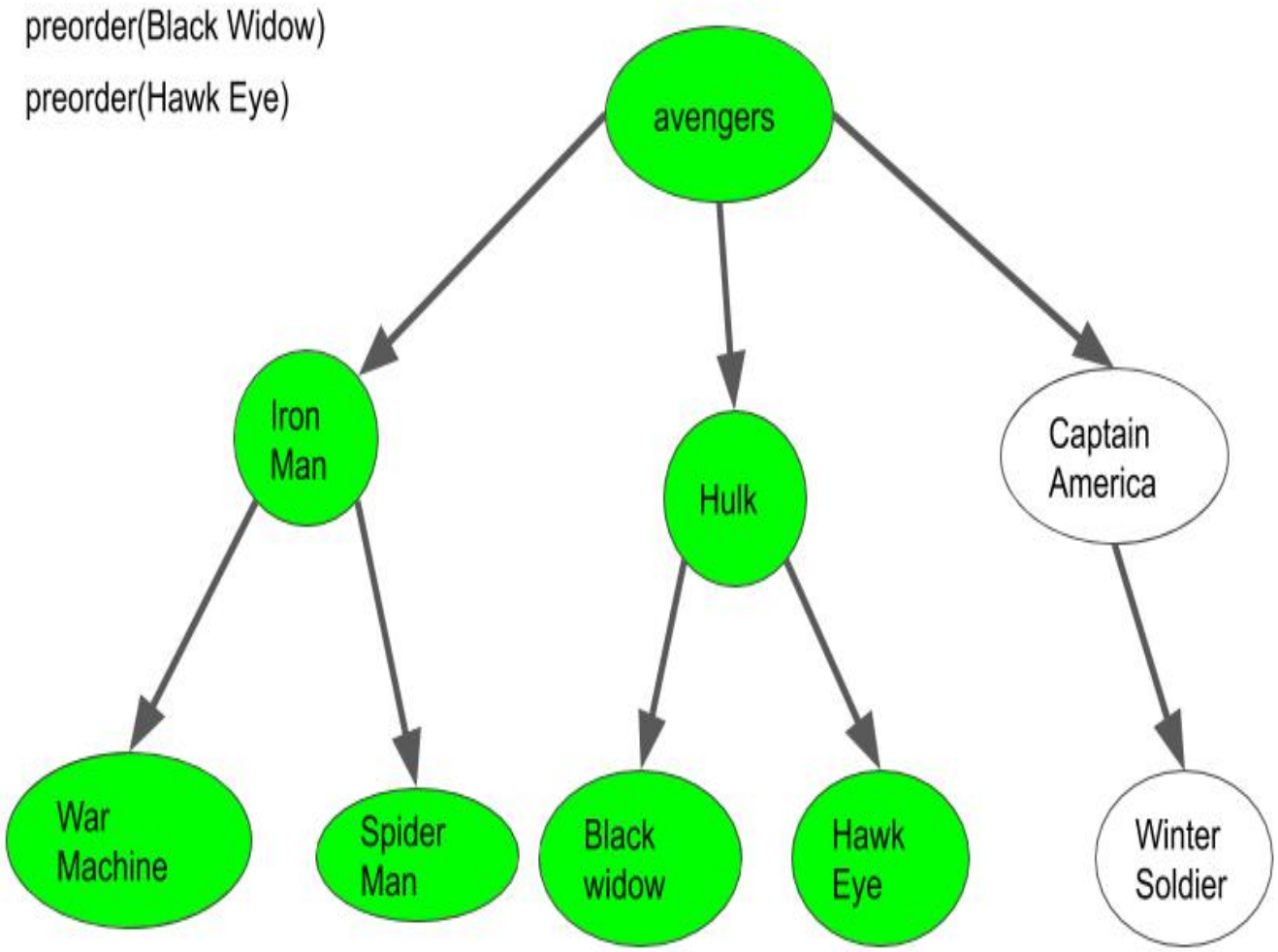
Recorrido en preorden



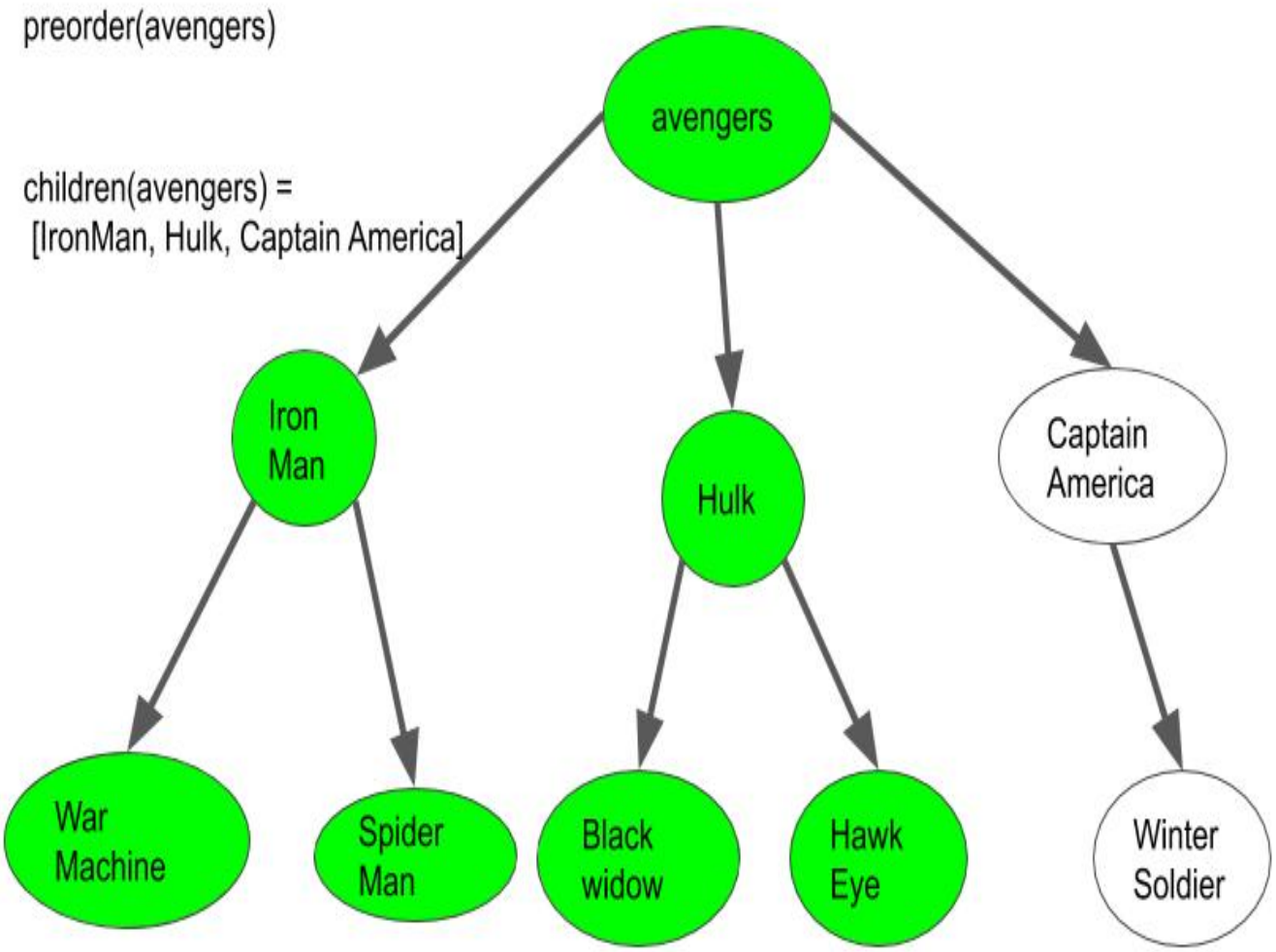
Recorrido en preorden



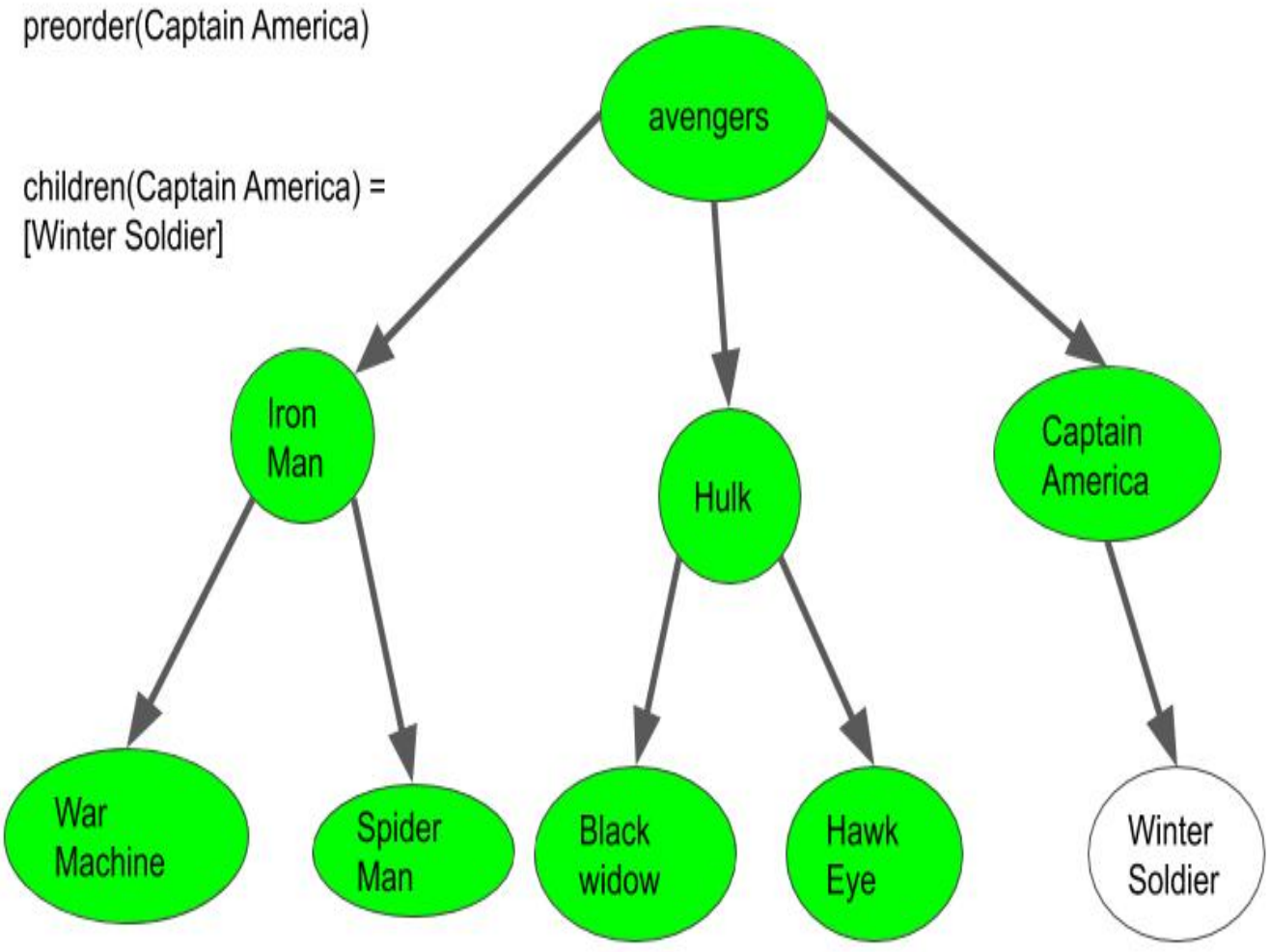
Recorrido en preorden



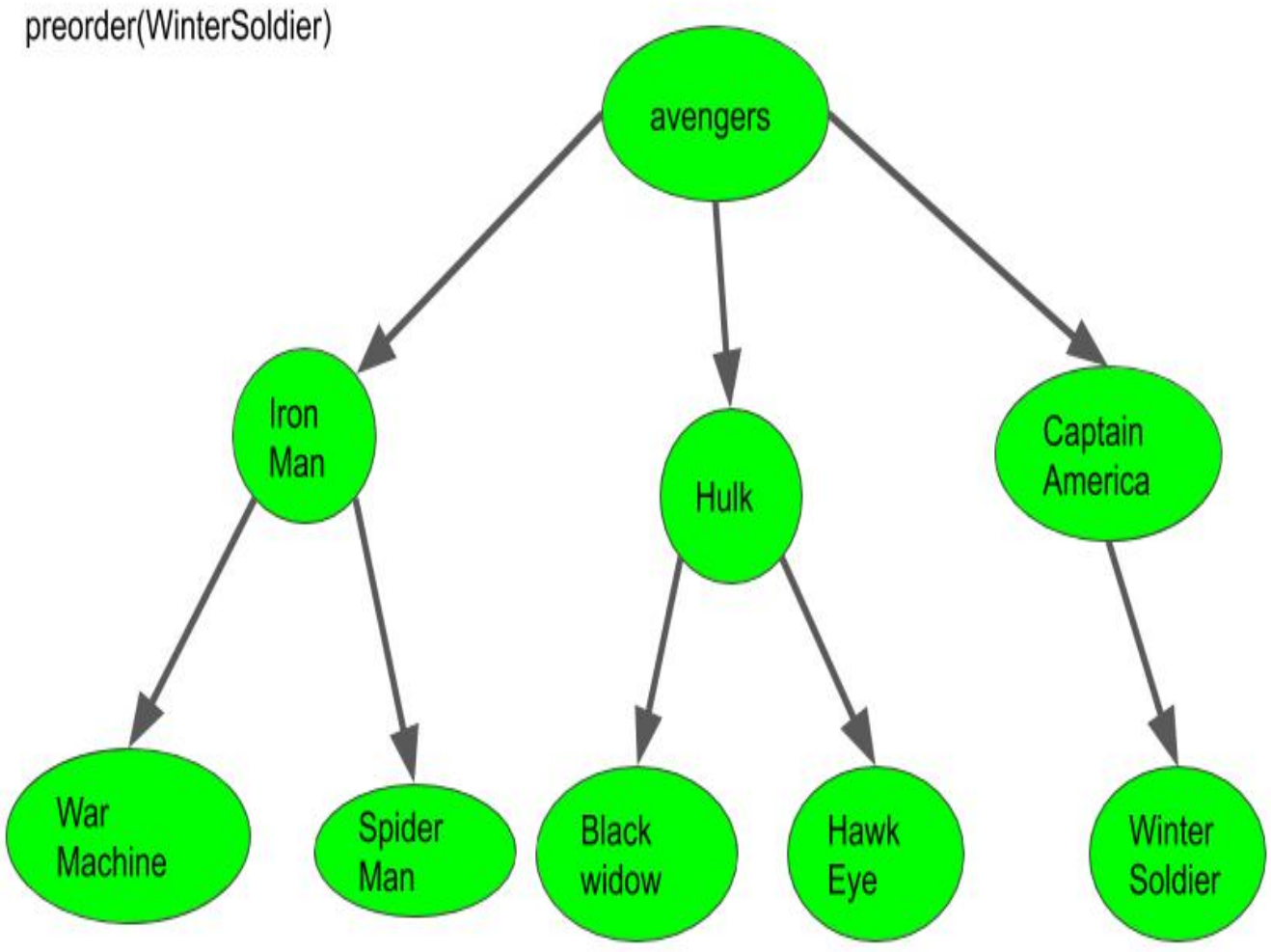
Recorrido en preorden



Recorrido en preorden



Recorrido en preorden



Recorrido en postorden

Trabaja de forma contraria al recorrido en preorden, pues aquí se recorren primero recursivamente los subárboles enraizados en los hijos del nodo raíz y después se marca como visitado este nodo. La invocación inicial es postorder(root()).

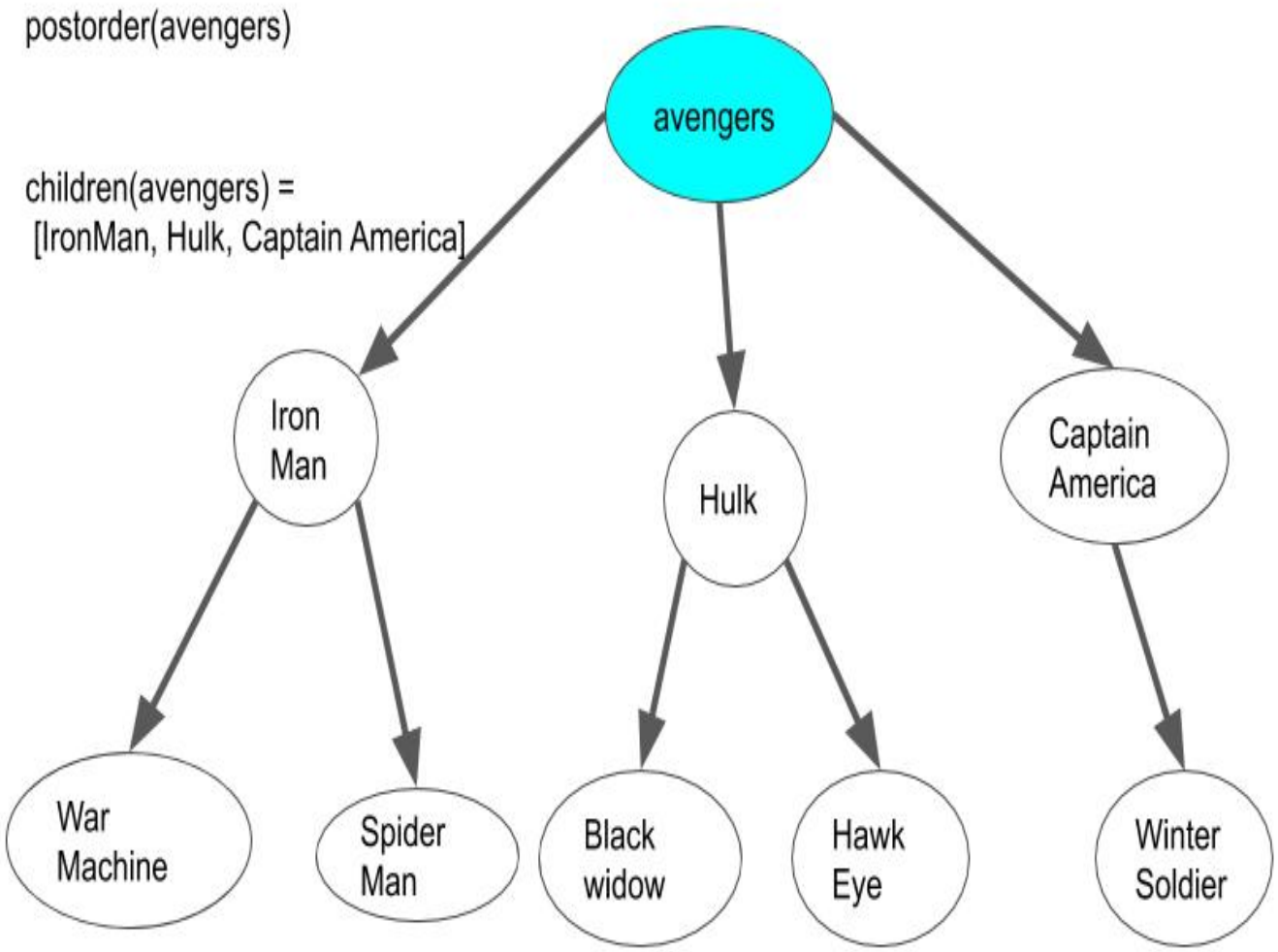
Listado 4 Algoritmo para recorrer un árbol en postorden

Precondiciones: v es un nodo de un árbol t.

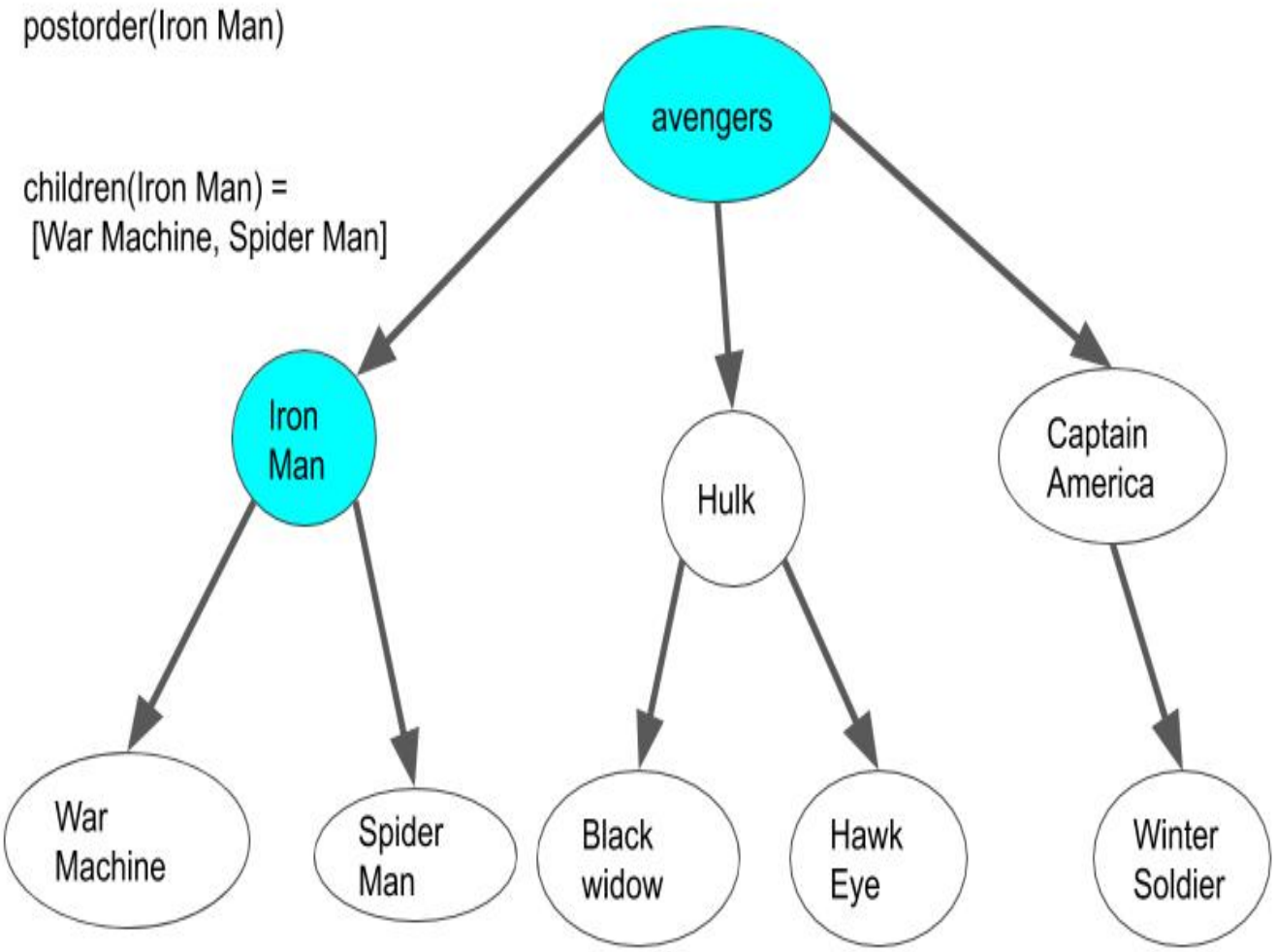
Poscondiciones: Recorre t en postorden de forma genérica.

```
postorder(v){
 1: for all w ∈ children(v) do
 2:     postorder(w);
 3: end for
 4: visit(v);
}
```

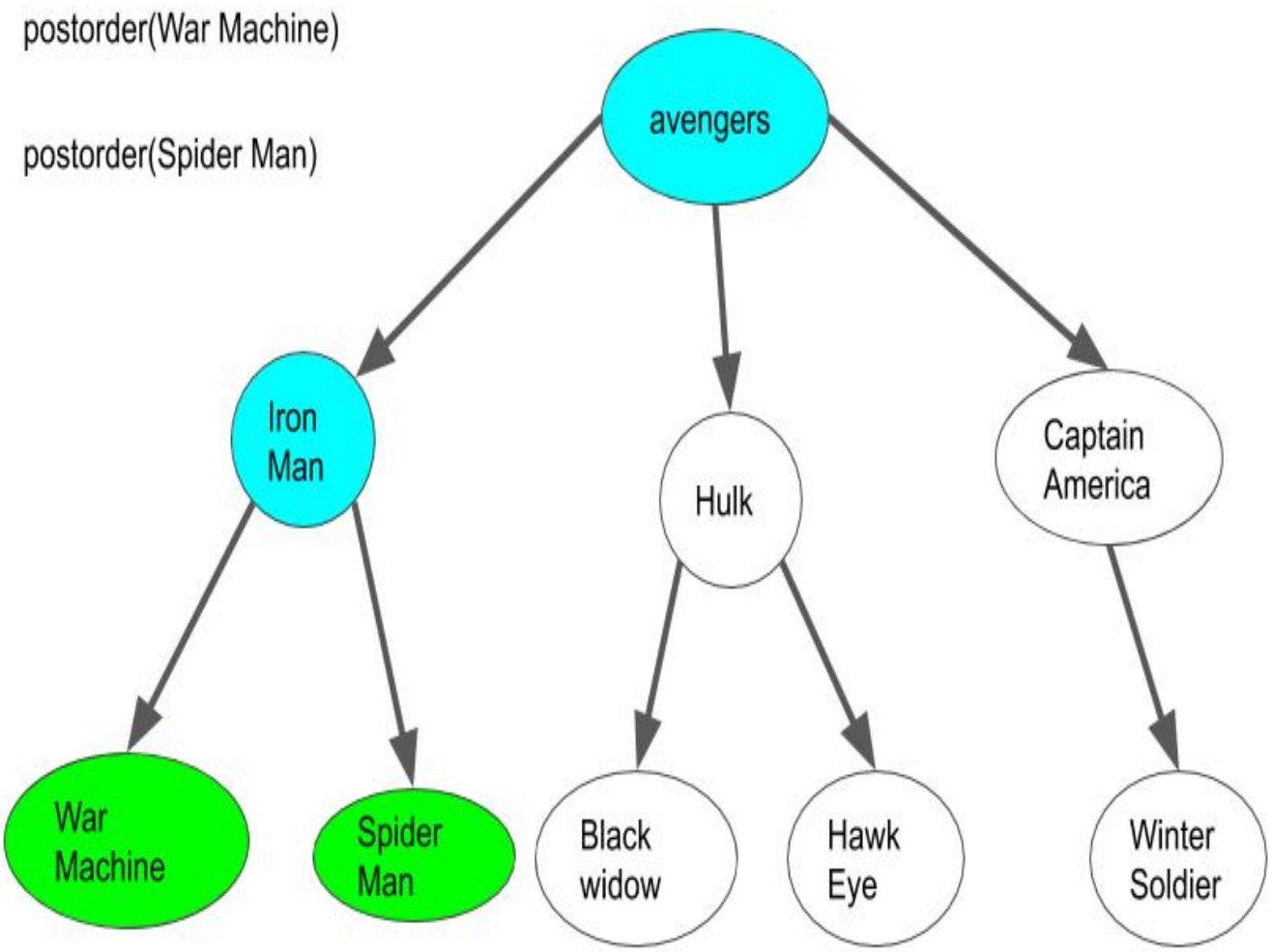
Recorrido en postorden



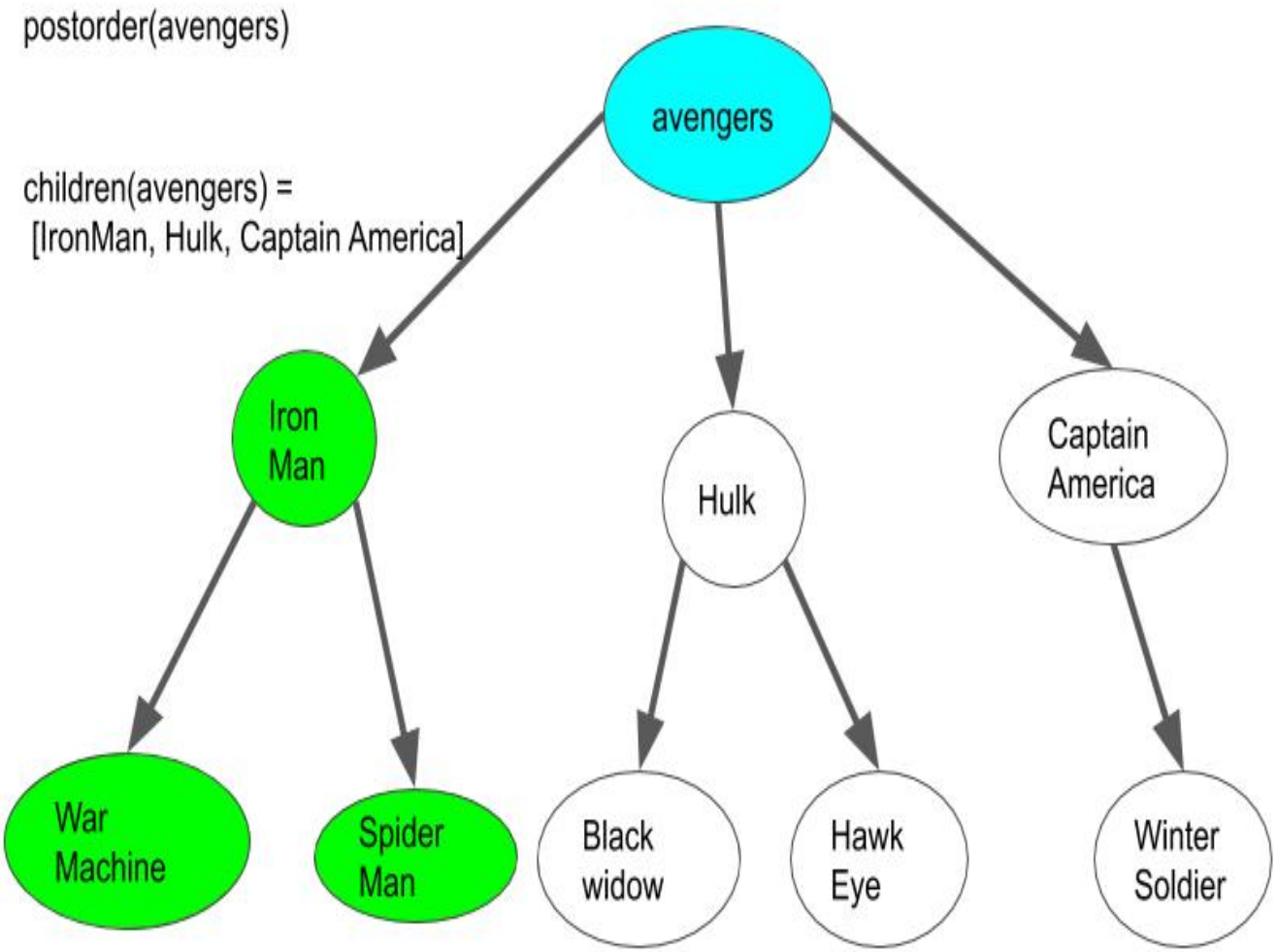
Recorrido en postorden



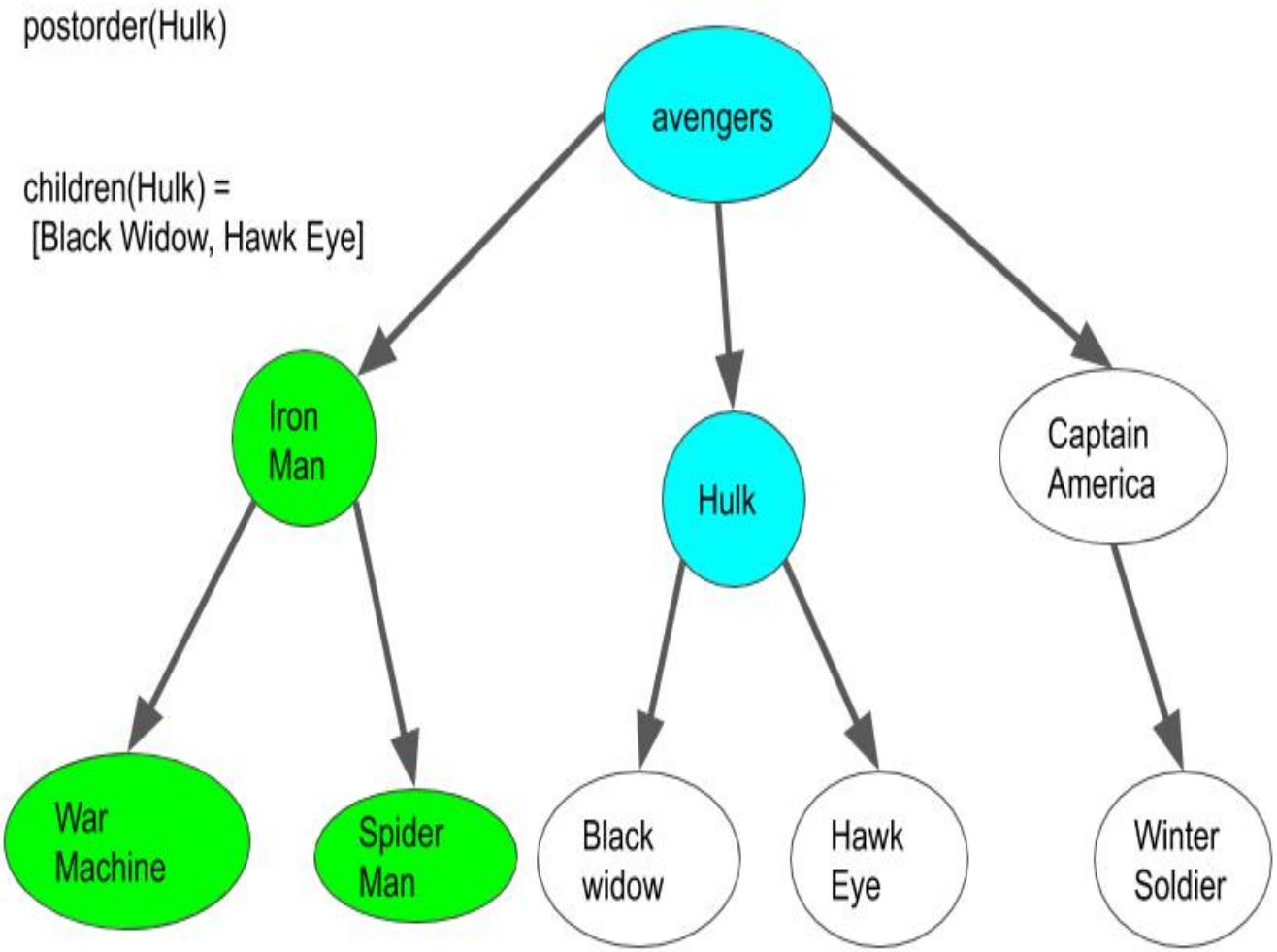
Recorrido en postorden



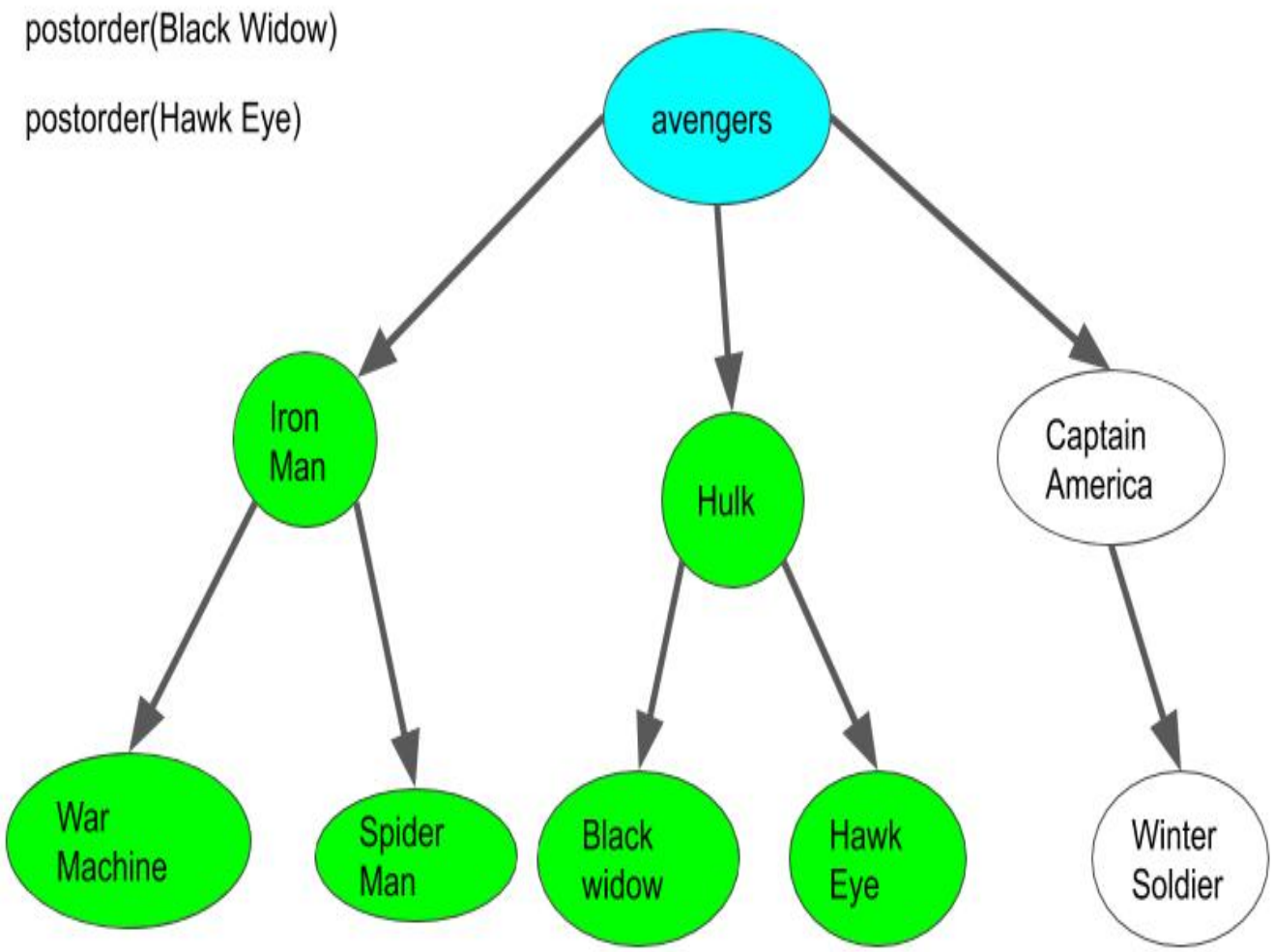
Recorrido en postorden



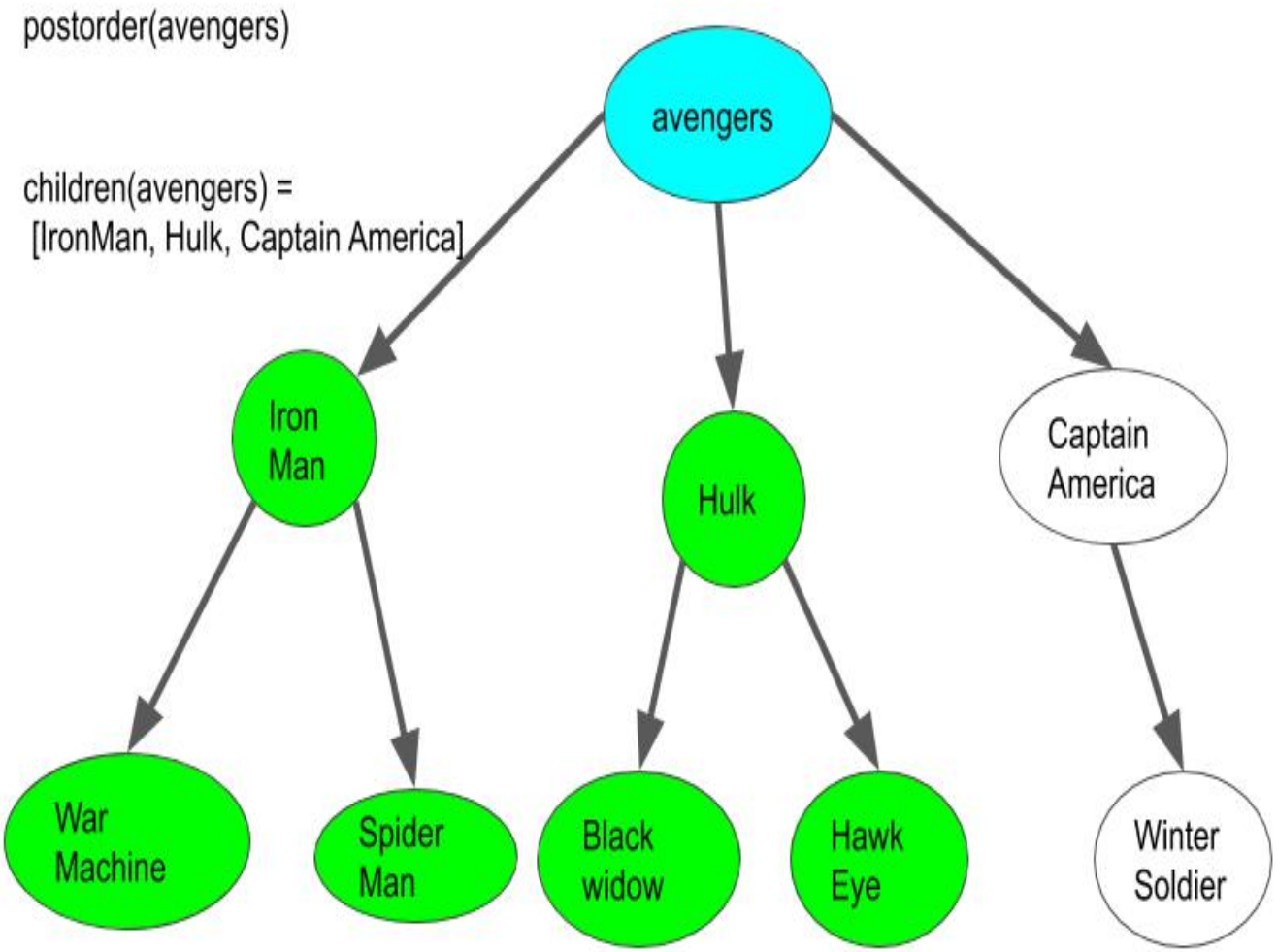
Recorrido en postorden



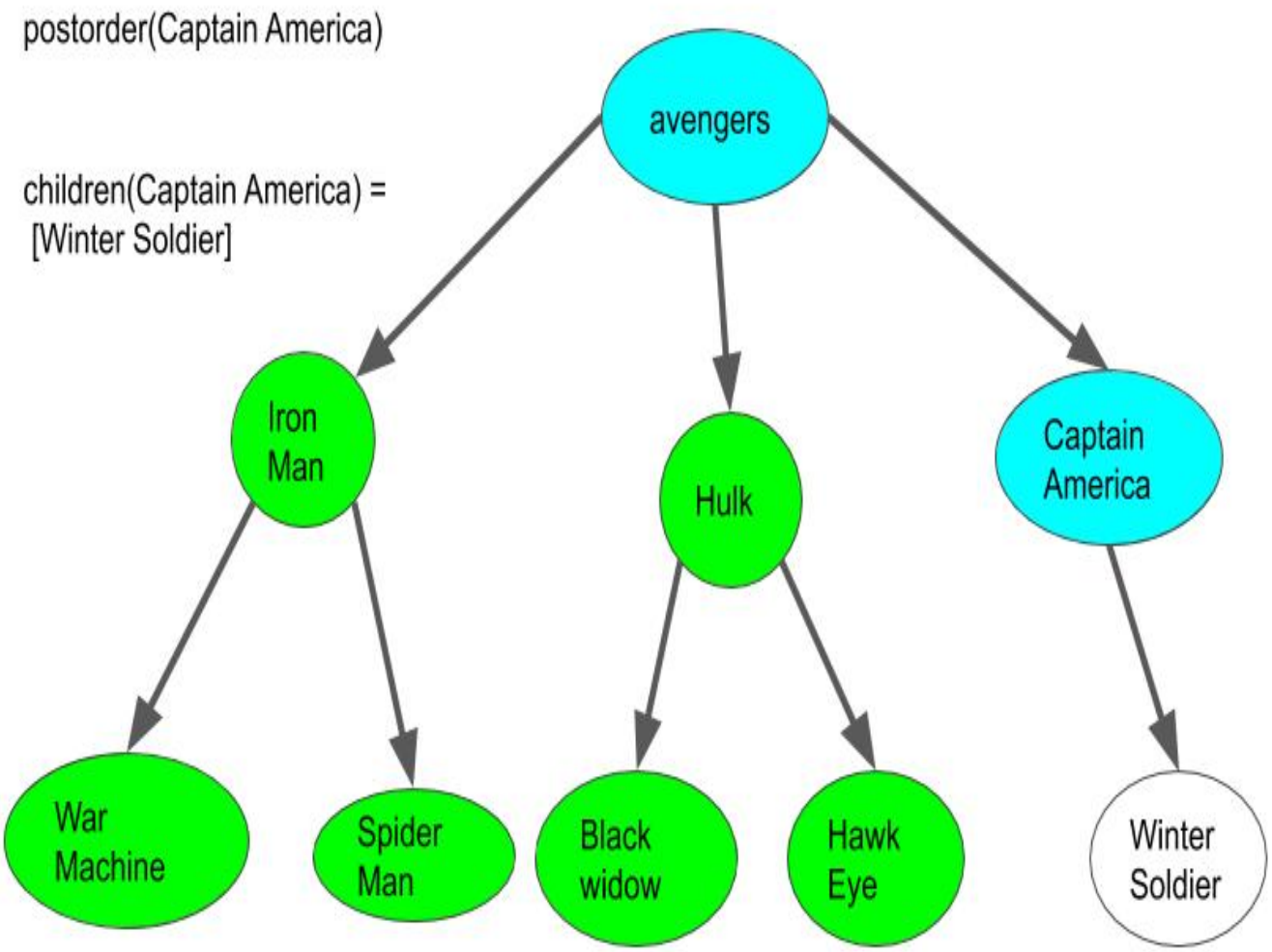
Recorrido en postorden



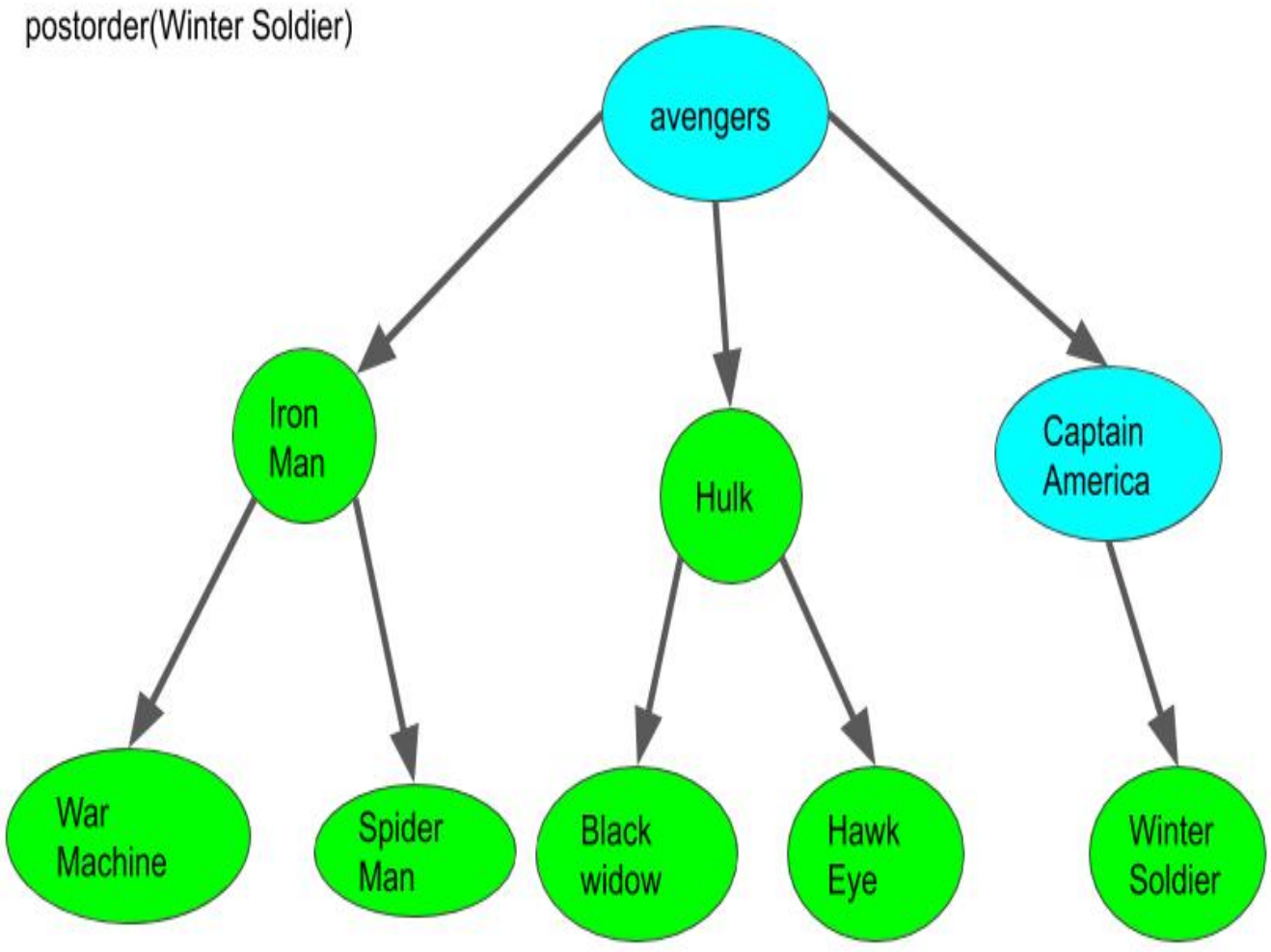
Recorrido en postorden



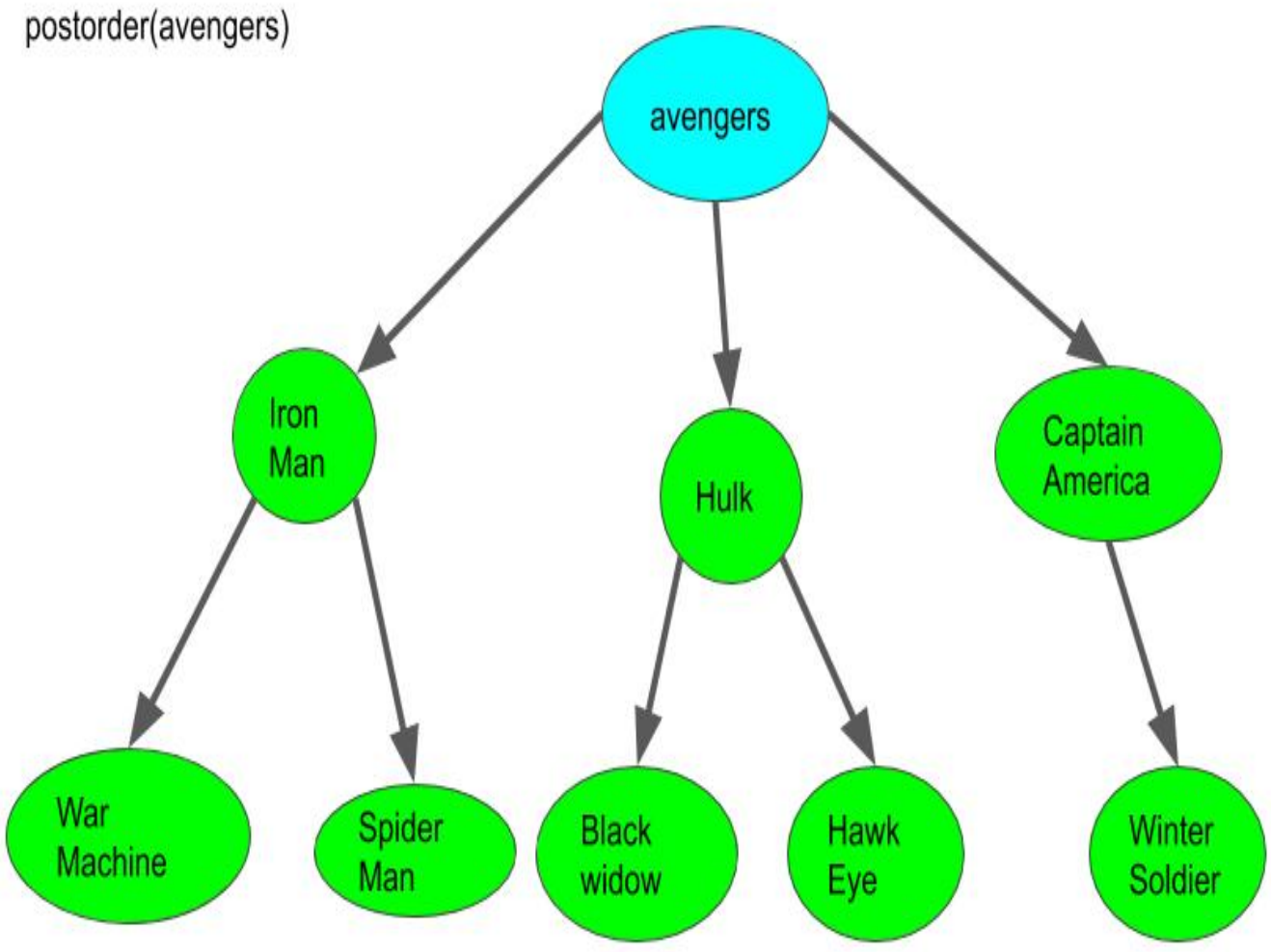
Recorrido en postorden



Recorrido en postorden



Recorrido en postorden



Recorrido en postorden