Gli iteratori

Anna Corazza

aa 2023/24

Sommario

Iteratori

Design pattern

- Behavioral design patter.
- Un'operazione importante su un generico contenitore è la visita di tutti gli elementi che contiene.
- Non vogliamo sapere come sono organizzati i dati all'interno del container, vogliamo solo poterli elencare in modo che nell'elenco ci siano tutti e nessuno sia ripetuto.
- Quindi deve essere il contenitore a gestire la visita e ad offrire all'esterno dei metodi per poterla compiere:
  1. restituisci il prossimo elemento `Hext()`
  2. l'elenco non è ancora finito `HasNext()` (o anche `IsDone()`)
- Vedi Struttura a `https://refactoring.guru/design-patterns/iterator`

Iteratori

La class Iterator

- `Next()` può venir implementato con gli operatori `++` e `()`
- `hasNext` o `isDone()` possono venir implementati con `Terminated()`
- Inoltre
  - restituisci l'elemento corrente`CurrentItem` che può venir implementato con `operator ()`
  - se sono resettable, ripristinare le condizioni iniziali `reset()`

Iteratori

Esempi d'uso

- Ricerca di una certa chiave all'interno di un contenitore.
- Sarà un metodo del contenitore.
- L'inizializzazione è fatta dal costruttore.

```
bool search(SomeData key){
  SomeIterator itr(*this);
  for(; !itr.Terminated(); ++itr)
    if(*itr == key)
      return true;
  return false;
}
```

Esempi d'uso

Copia generica facendo uso di template

- Copia di un contenitore.

```
template<class In, class Out>
Out copy(In first, In last, Out res){
    while( first != last ) {
        *res = *first;
        ++first;
        ++res;
    }
    return res;
}
```

Iteratori esterni o interni

- Dipende da chi chiama l'iteratore:

  esterni (o attivi) Una funzion o altro (client, nella terminlogia di pattern design) che usa esplicitamente l'iteratore, richiamandone le funzionalità.

  Interni (o passivi) L'iteratore stesso: in questo caso chi lo usa vede solo una funzione di visita (come `traverse()` o `map()`) a cui passa l'operazione da eseguire.

Iteratori e cursori

- Il cursore è un iteratore più “leggero”: si limita a salvare lo stato dell’iterazione.
- In questo caso, il client invoca il metodo `next` sull’aggregazione passando il cursore come argomento.
- `next()` andrà ad aggiornare il cursore.

La classe `iterator`

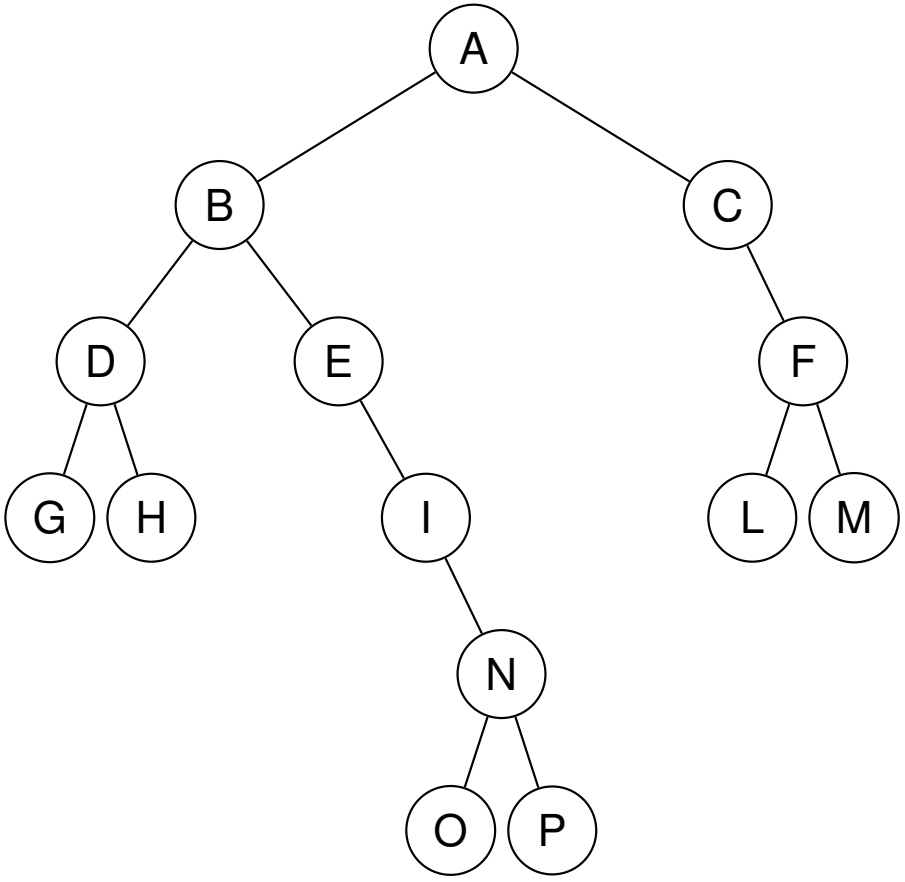
- Classe che implementa questo pattern.
- Nell'esercizio 2 i contenitori sono basati su alberi binari.

Visita in ampiezza

- Serve una **coda** di supporto.
  1. lettura del dato: metodo della classe;
  2. controllo di terminazione: coda vuota;
  3. successore: deque del nodo visitato e enque dei due figli del nodo corrente.

Visita in ampiezza

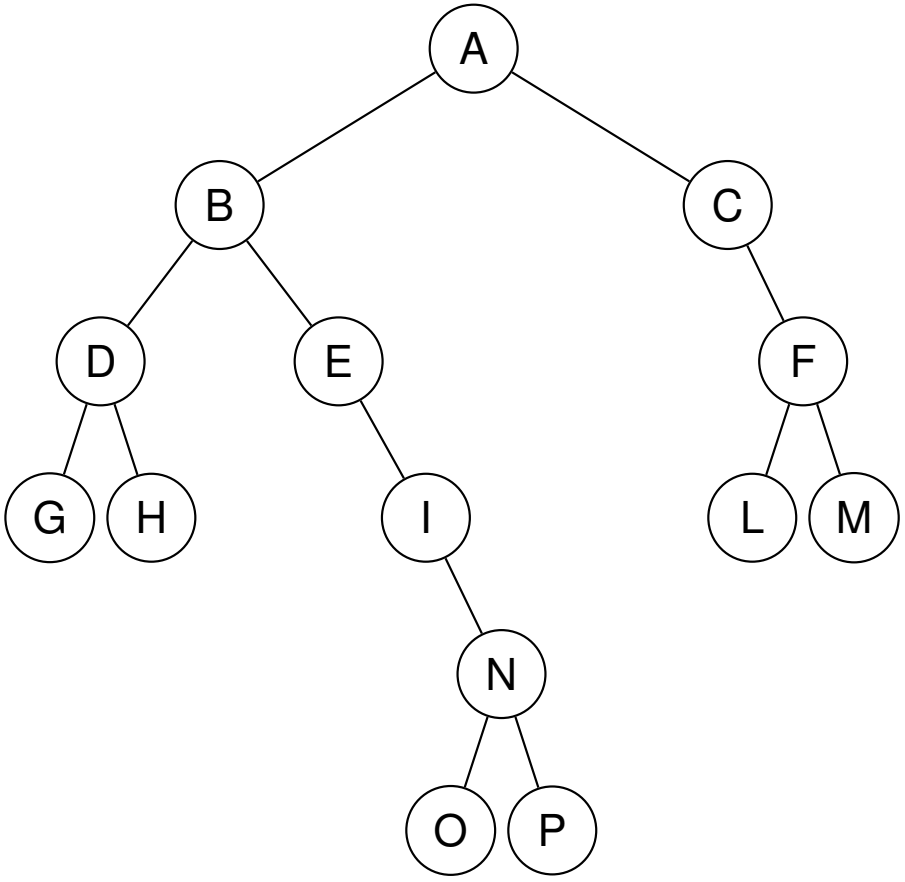
Esempio



- ϵ

Visita in ampiezza

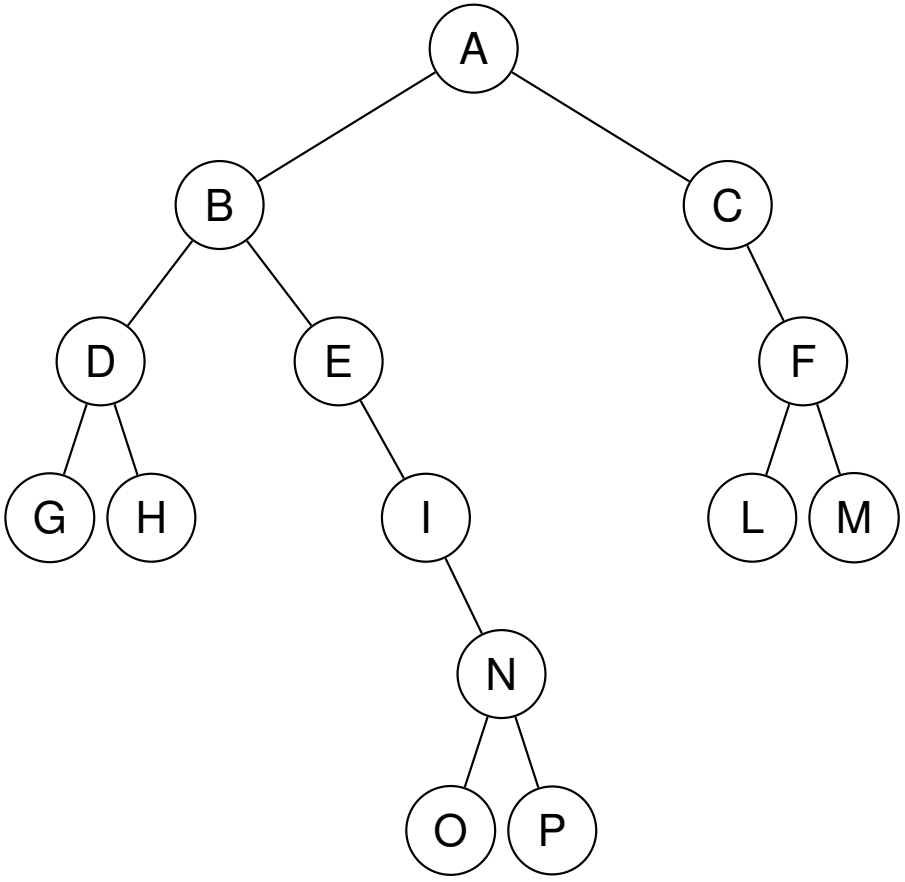
Esempio



- $\boxed{\epsilon}$
- $\boxed{A} \rightarrow A$

Visita in ampiezza

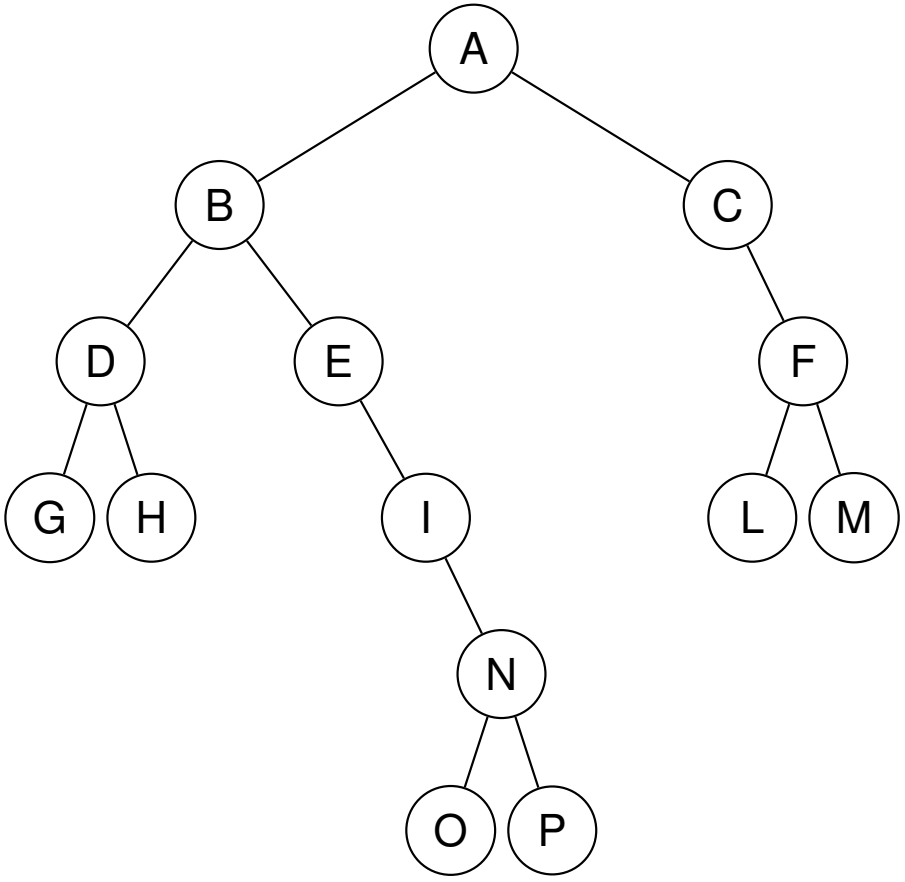
Esempio



- $\boxed{\epsilon}$
- $\boxed{\text{A}} \rightarrow \text{A}$
- $\boxed{\text{B,C}} \rightarrow \text{B}$

Visita in ampiezza

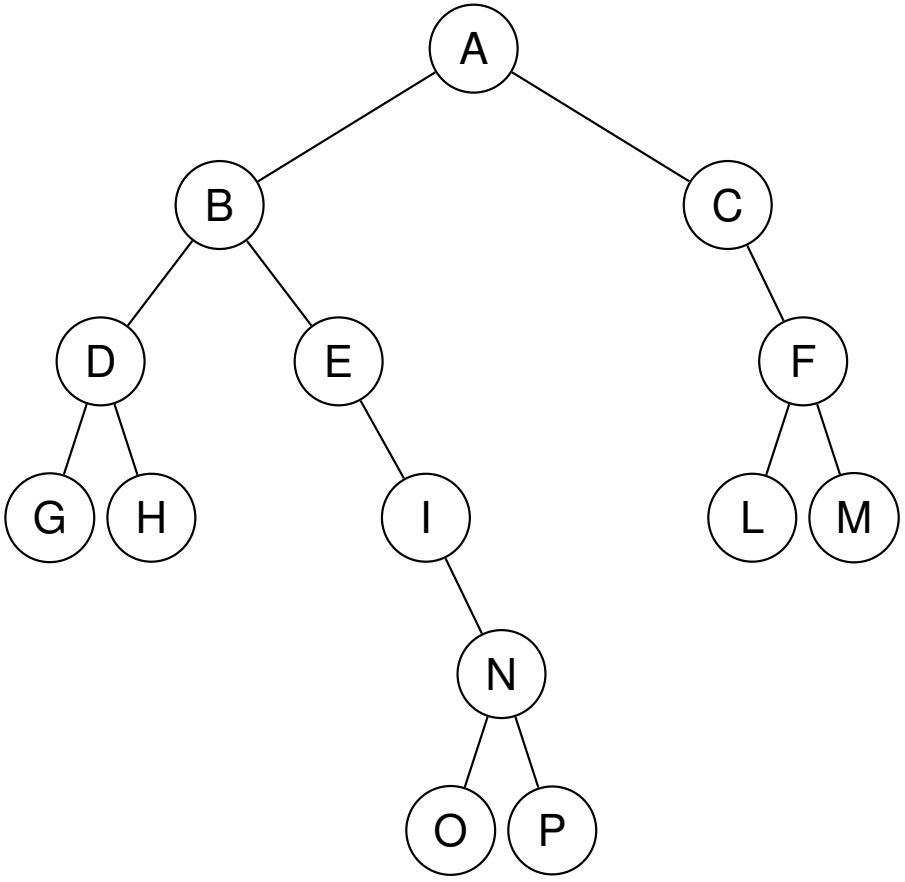
Esempio



- $\boxed{\epsilon}$
- $\boxed{\text{A}} \rightarrow \text{A}$
- $\boxed{\text{B,C}} \rightarrow \text{B}$
- $\boxed{\text{C,D,E}} \rightarrow \text{C}$

Visita in ampiezza

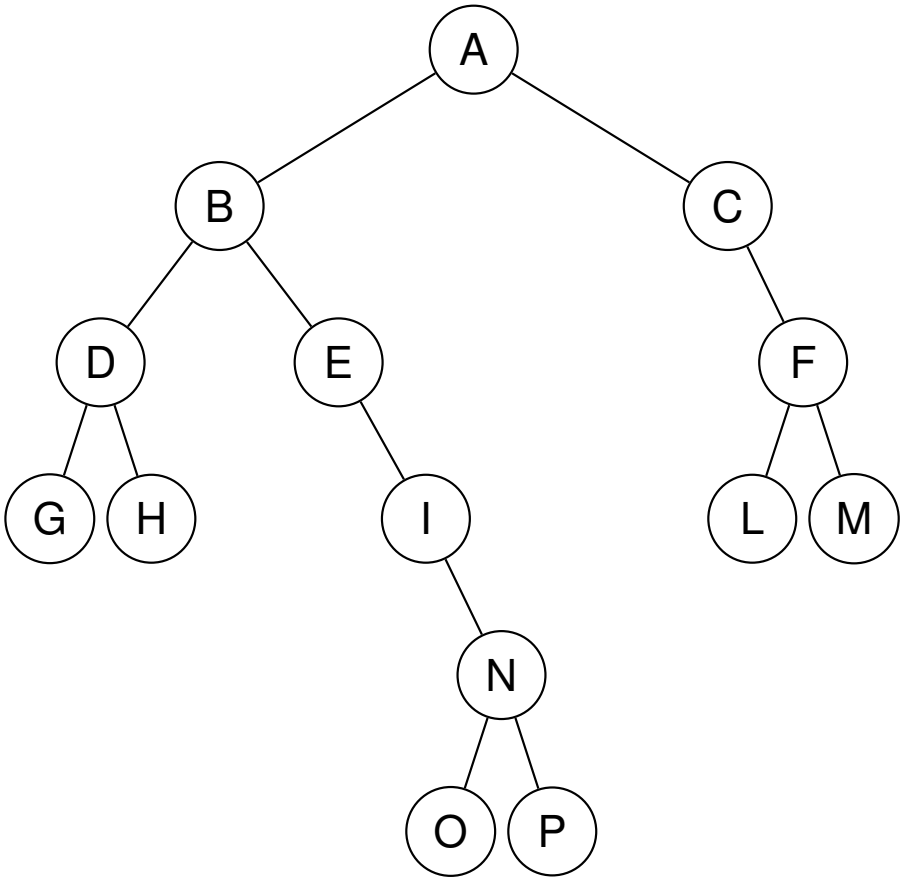
Esempio



- $\boxed{\epsilon}$
- $\boxed{\text{A}} \rightarrow \text{A}$
- $\boxed{\text{B,C}} \rightarrow \text{B}$
- $\boxed{\text{C,D,E}} \rightarrow \text{C}$
- ...

Visita in ampiezza

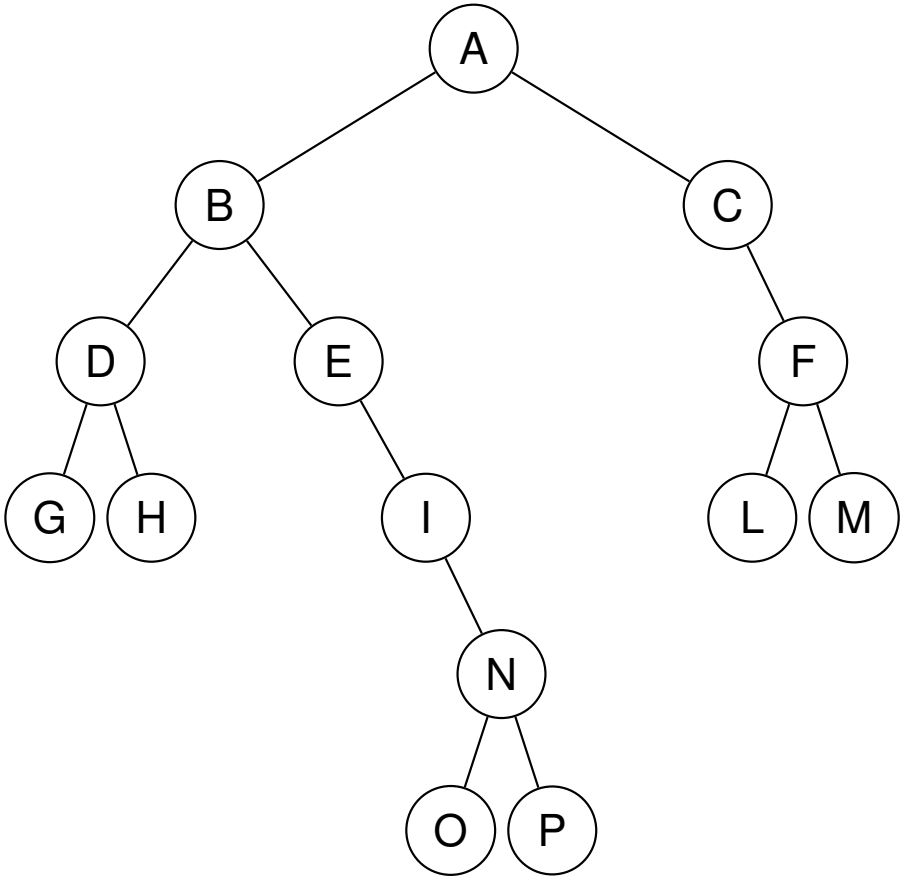
Esempio



- $\boxed{\epsilon}$
- $\boxed{\text{A}} \rightarrow \text{A}$
- $\boxed{\text{B,C}} \rightarrow \text{B}$
- $\boxed{\text{C,D,E}} \rightarrow \text{C}$
- ...
- $\boxed{\text{P}} \rightarrow \text{P}$

Visita in ampiezza

Esempio



- $\boxed{\epsilon}$
- $\boxed{\text{A}} \rightarrow \text{A}$
- $\boxed{\text{B,C}} \rightarrow \text{B}$
- $\boxed{\text{C,D,E}} \rightarrow \text{C}$
- ...
- $\boxed{\text{P}} \rightarrow \text{P}$
- $\boxed{\epsilon}$ condizione di terminazione

Visita in ampiezza

Discussione

Visita in profondità

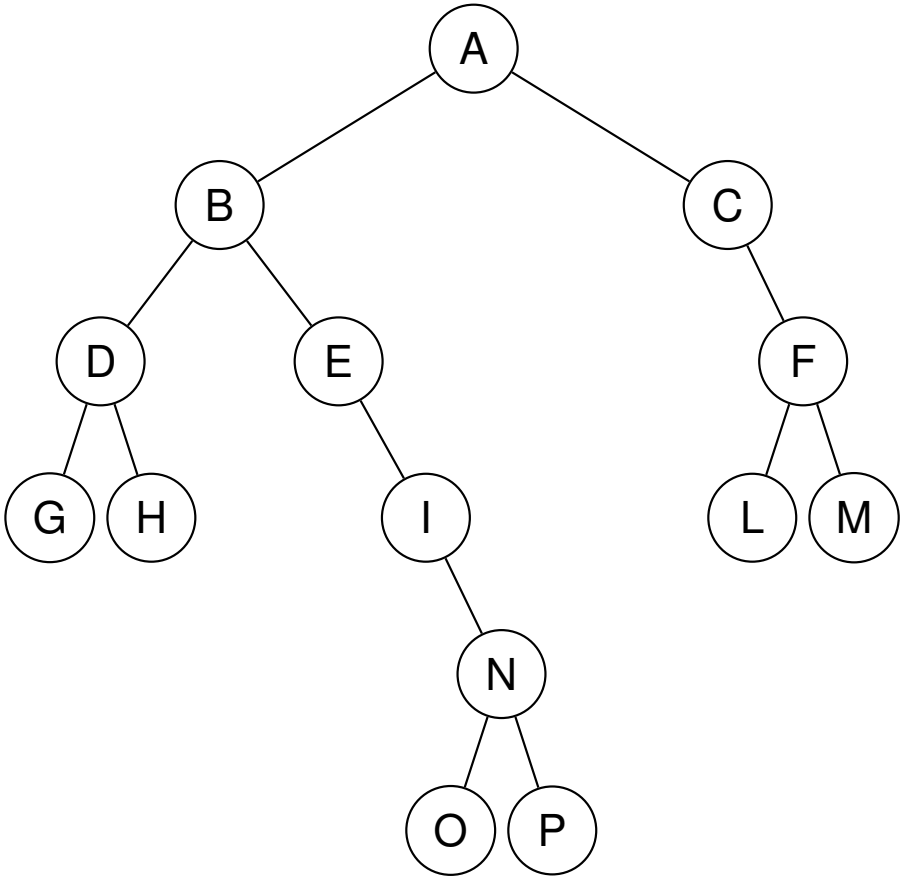
- Tre strategie:
  - Preorder
  - Inorder
  - Postorder
- La versione ricorsiva non ci aiuta molto: meglio pensare all'iterativa.

Visita preorder

- Serve uno **stack** di supporto.
  1. lettura del dato: metodo della classe;
  2. controllo di terminazione: stack vuoto;
  3. successore: pop del nodo visitato e push dei due figli del nodo corrente.

Visita preorder

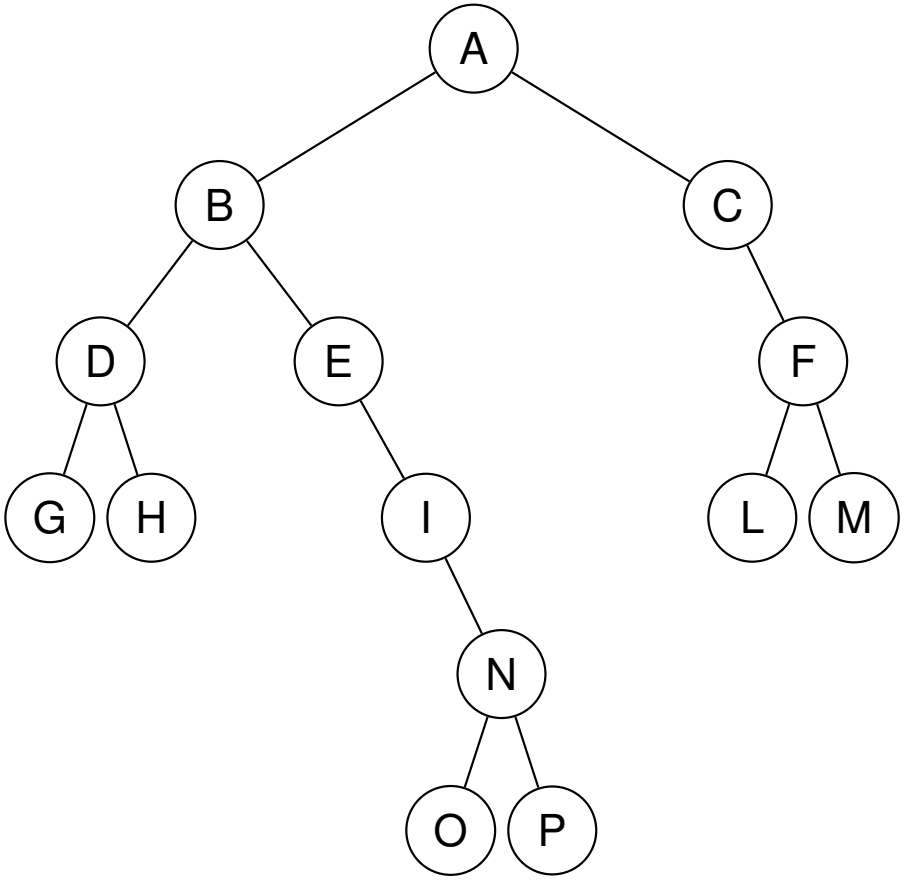
Esempio



- $\boxed{\epsilon}$

Visita preorder

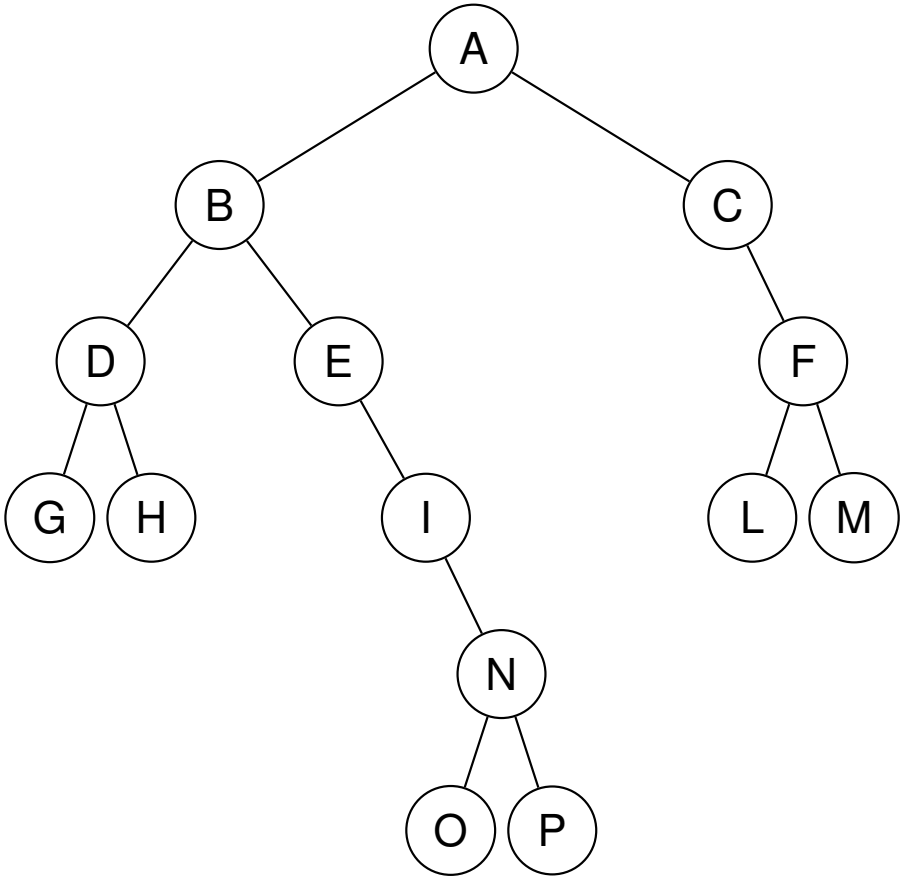
Esempio



- $\boxed{\epsilon}$
- $\boxed{A}$: A

Visita preorder

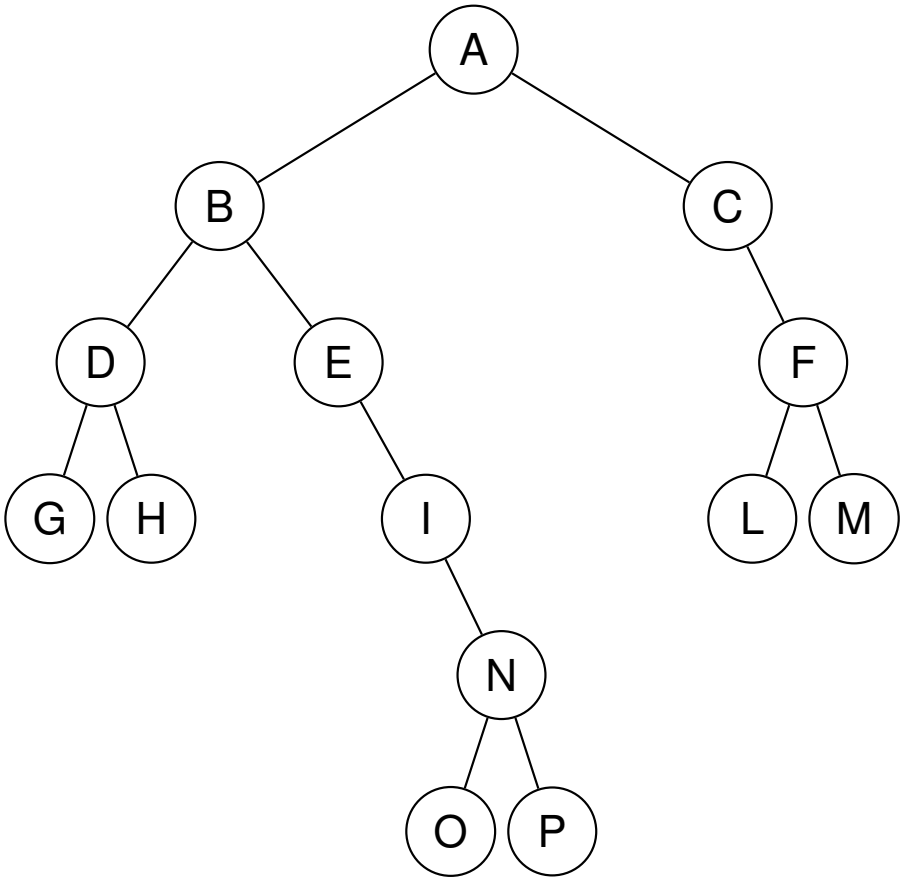
Esempio



- ϵ
- A : A
- B,C : B

Visita preorder

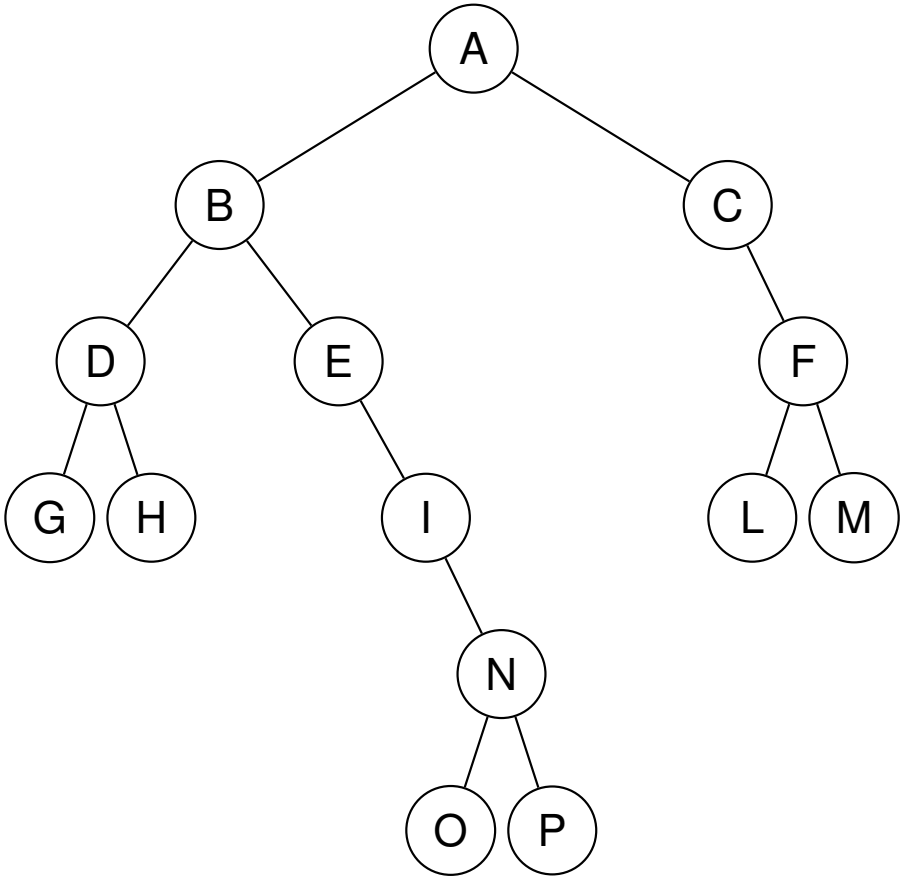
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D

Visita preorder

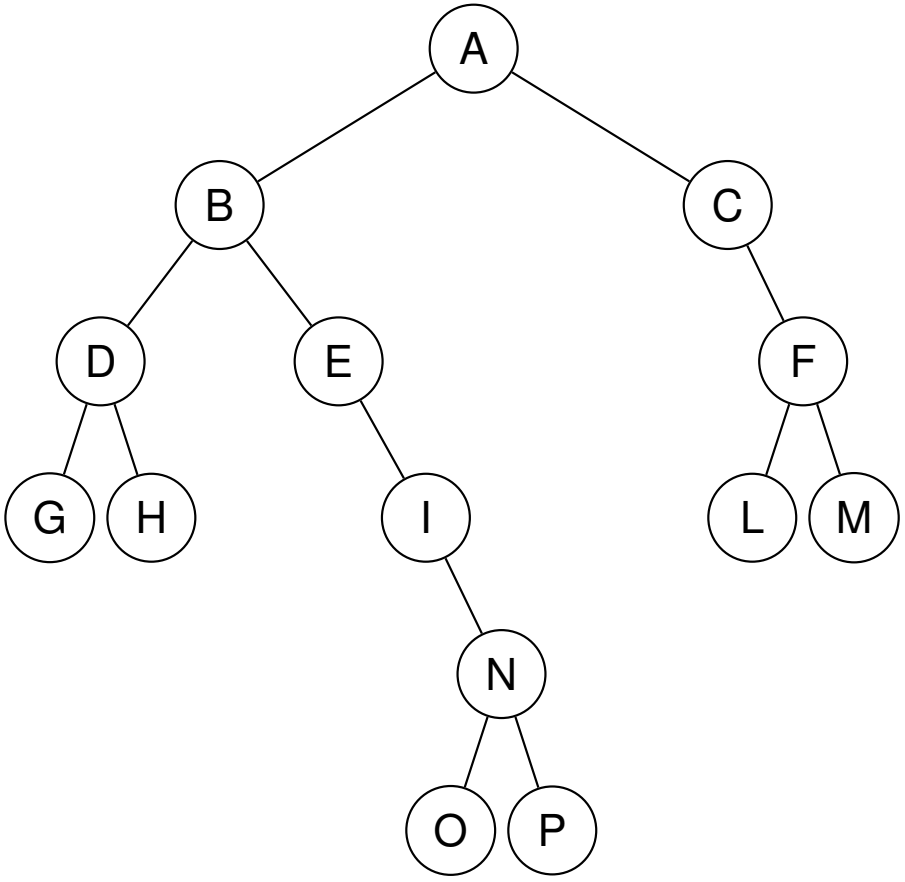
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G

Visita preorder

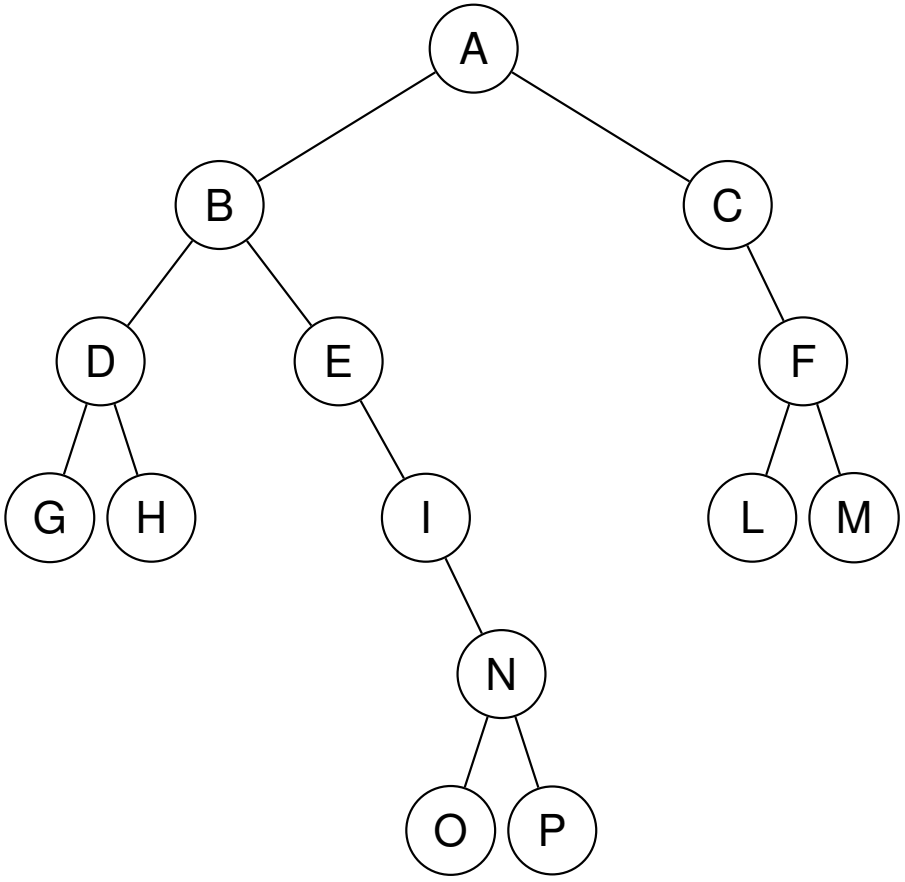
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H

Visita preorder

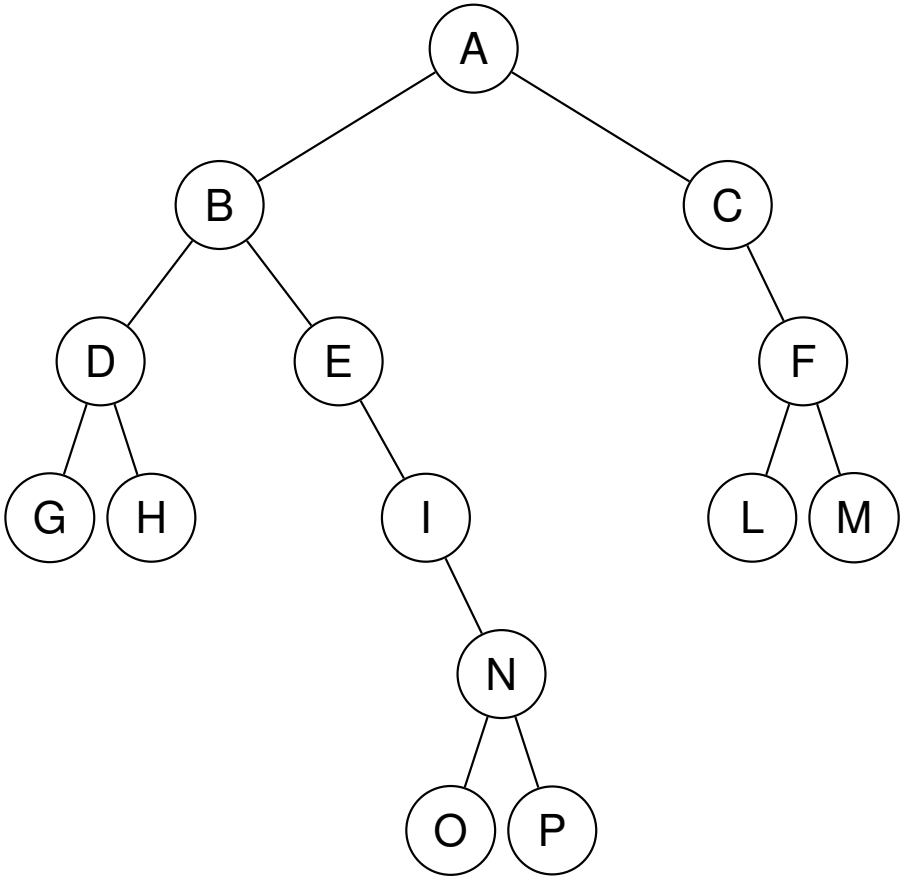
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E

Visita preorder

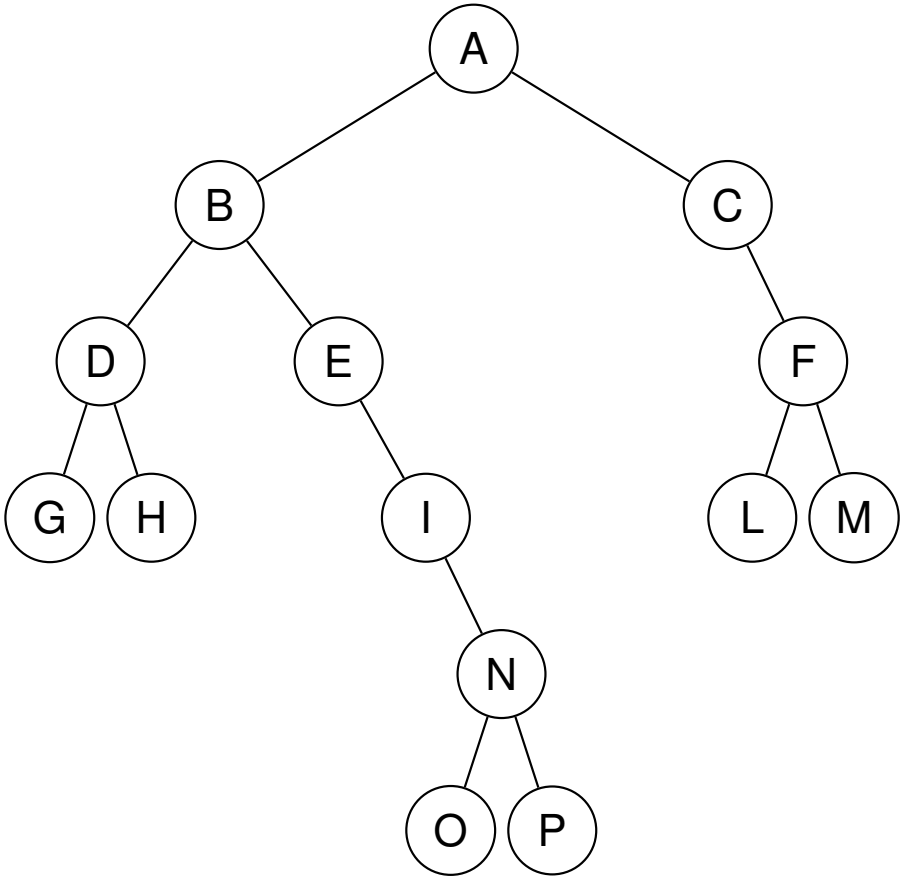
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I

Visita preorder

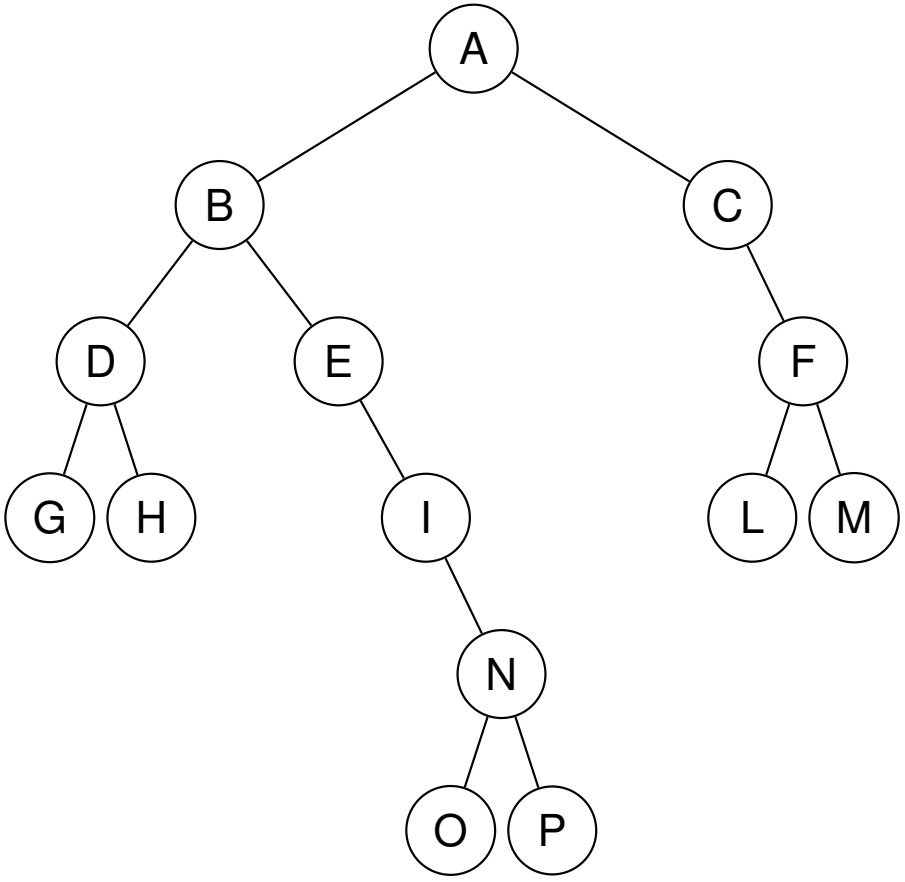
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N

Visita preorder

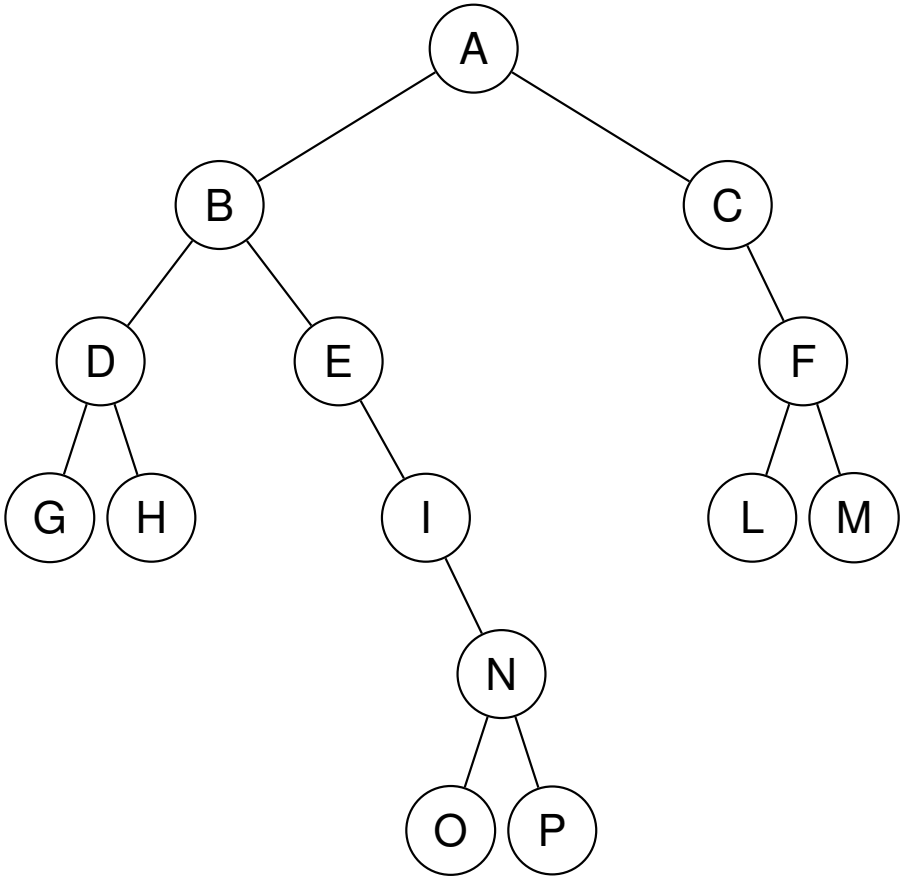
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N
- O,P,C : O

Visita preorder

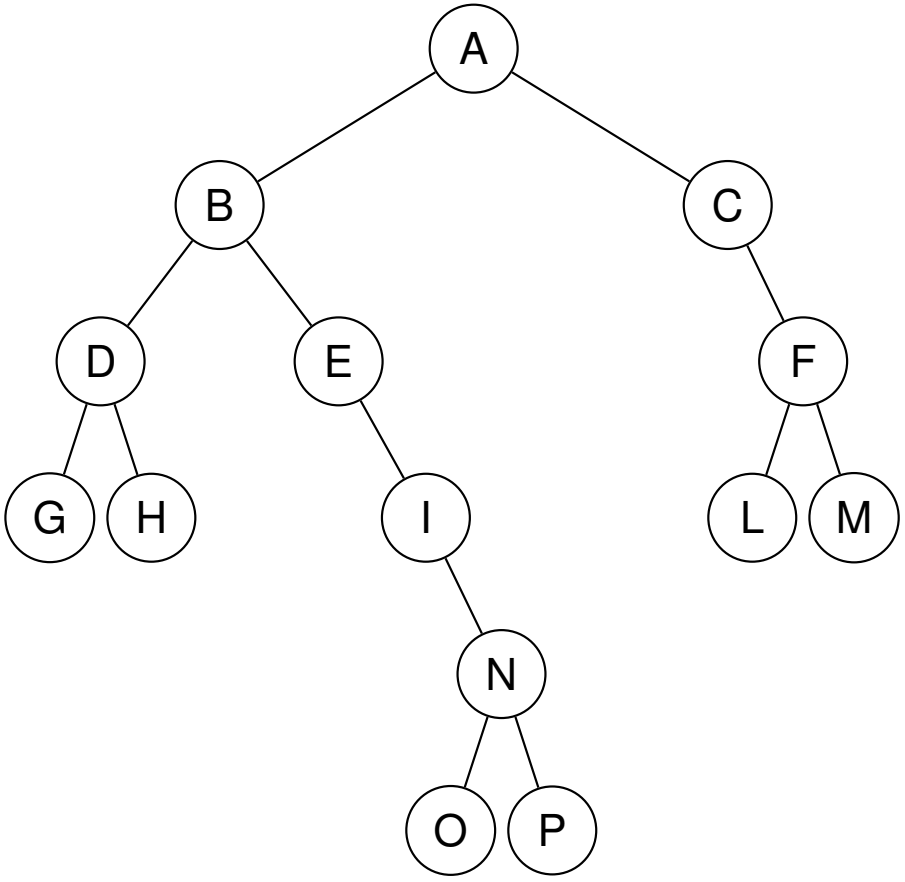
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N
- O,P,C : O
- P,C : P

Visita preorder

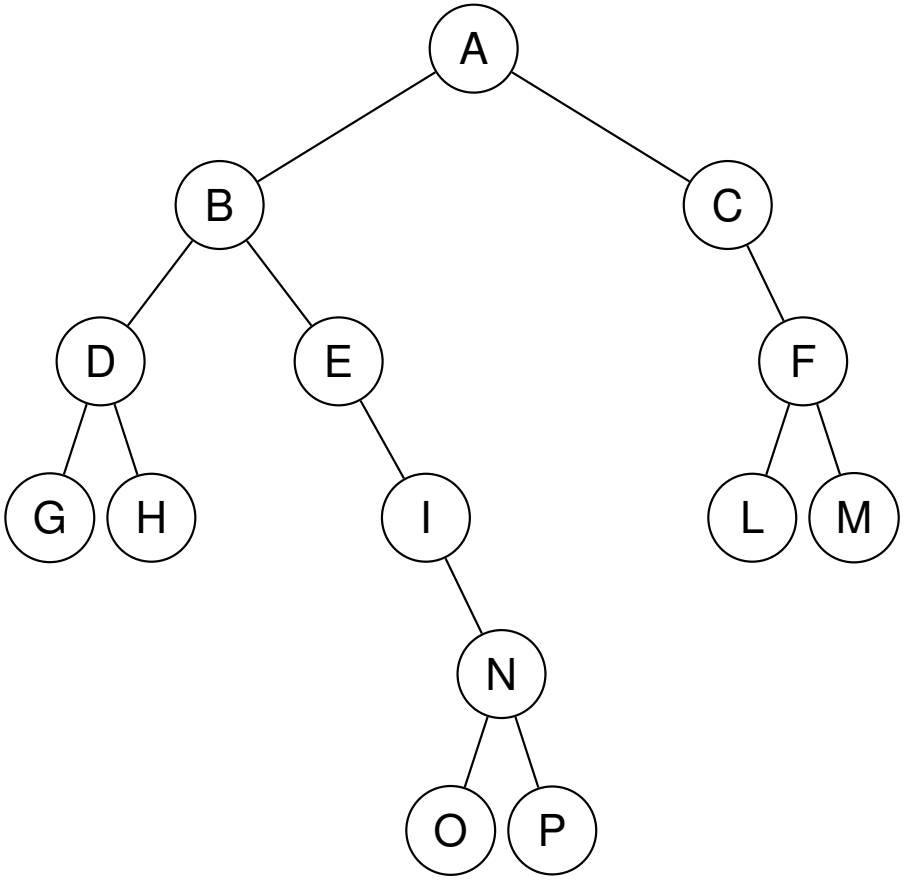
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N
- O,P,C : O
- P,C : P
- C : C

Visita preorder

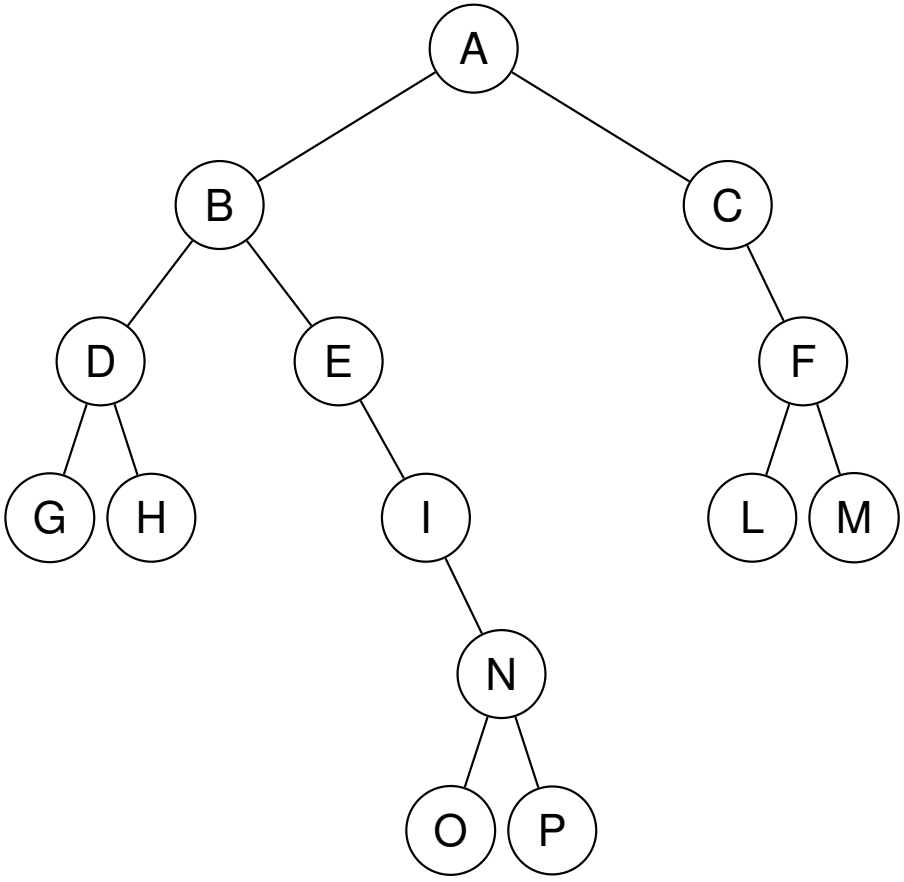
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N
- O,P,C : O
- P,C : P
- C : C
- F : F

Visita preorder

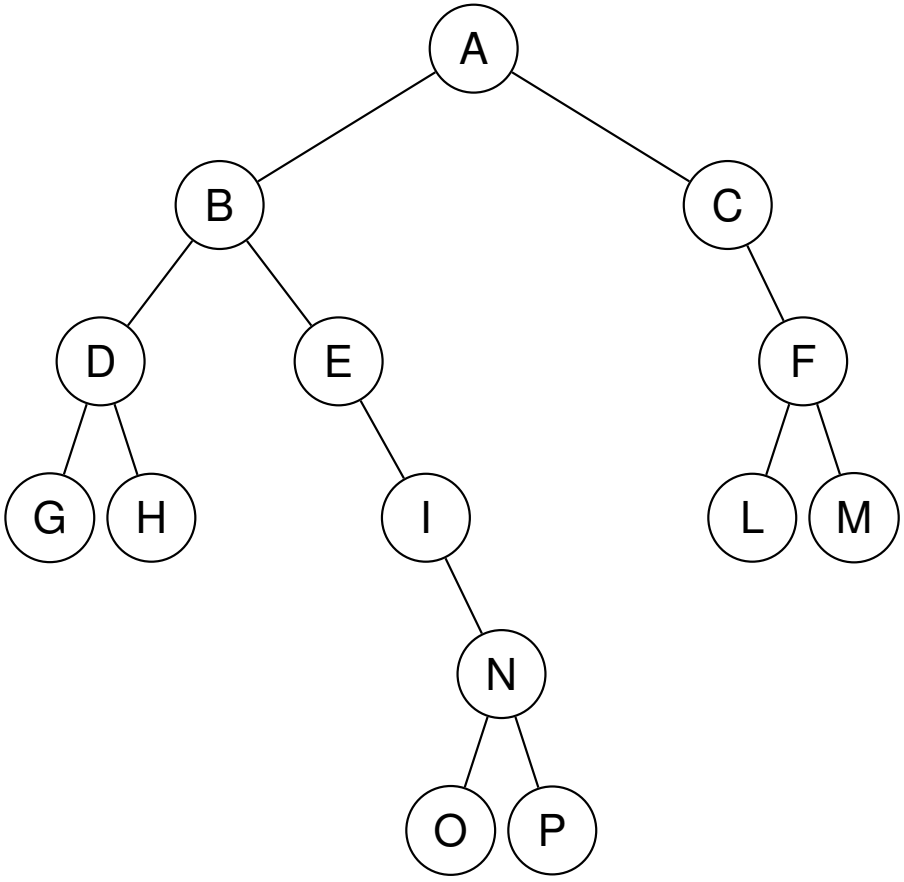
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N
- O,P,C : O
- P,C : P
- C : C
- F : F
- L,M : L

Visita preorder

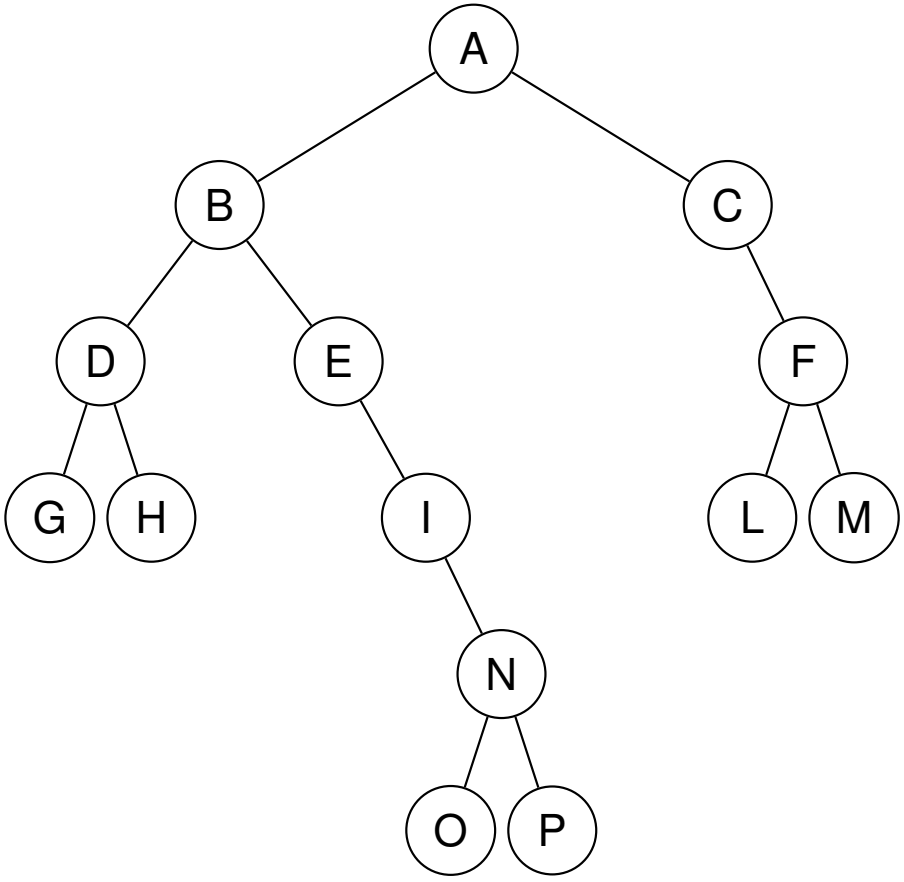
Esempio



- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N
- O,P,C : O
- P,C : P
- C : C
- F : F
- L,M : L
- M : M

Visita preorder

Esempio



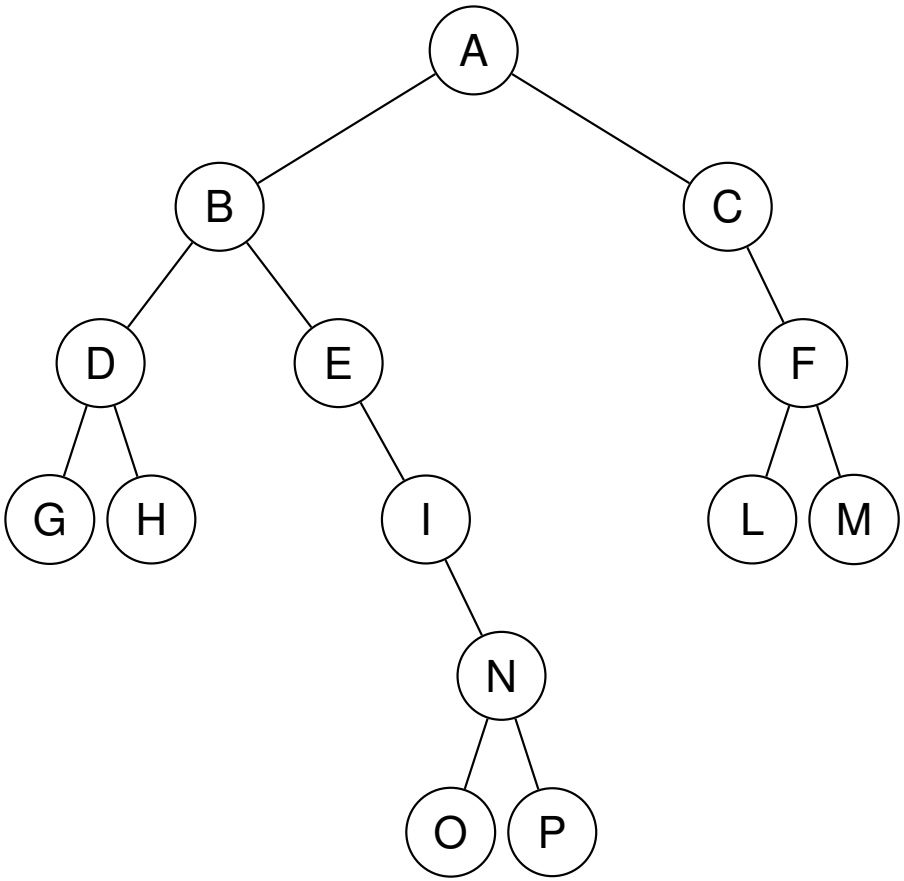
- ϵ
- A : A
- B,C : B
- D,E,C : D
- G,H,E,C : G
- H,E,C : H
- E,C : E
- I,C : I
- N,C : N
- O,P,C : O
- P,C : P
- C : C
- F : F
- L,M : L
- M : M
- ϵ condizione di terminazione

Visita inorder

- In questo caso il nodo corrente viene **scoperto**, ma non ancora **visitato**: prima bisogna visitare il sottoalbero sinistro.
- Solito **stack** di supporto.
- `searchLeftMostNode` Ci serve anche una funzione che continua a scendere a sinistra fino a quando possibile inserendo man mano i nodi che incontra nello stack: si ferma al primo nodo il cui figlio sinistro è vuoto.
- `scopri` Quando scopriamo un nodo, ne facciamo il push nello stack e poi scendiamo a sinistra con `searchLeftMostNode`
  1. lettura del dato: metodo della classe;
  2. controllo di terminazione: stack vuoto;
  3. successore: Pop e restituisco il nodo; visita il suo nodo destro, se c'è.
- Naturalmente parto dalla radice dell'albero.

Visita inorder

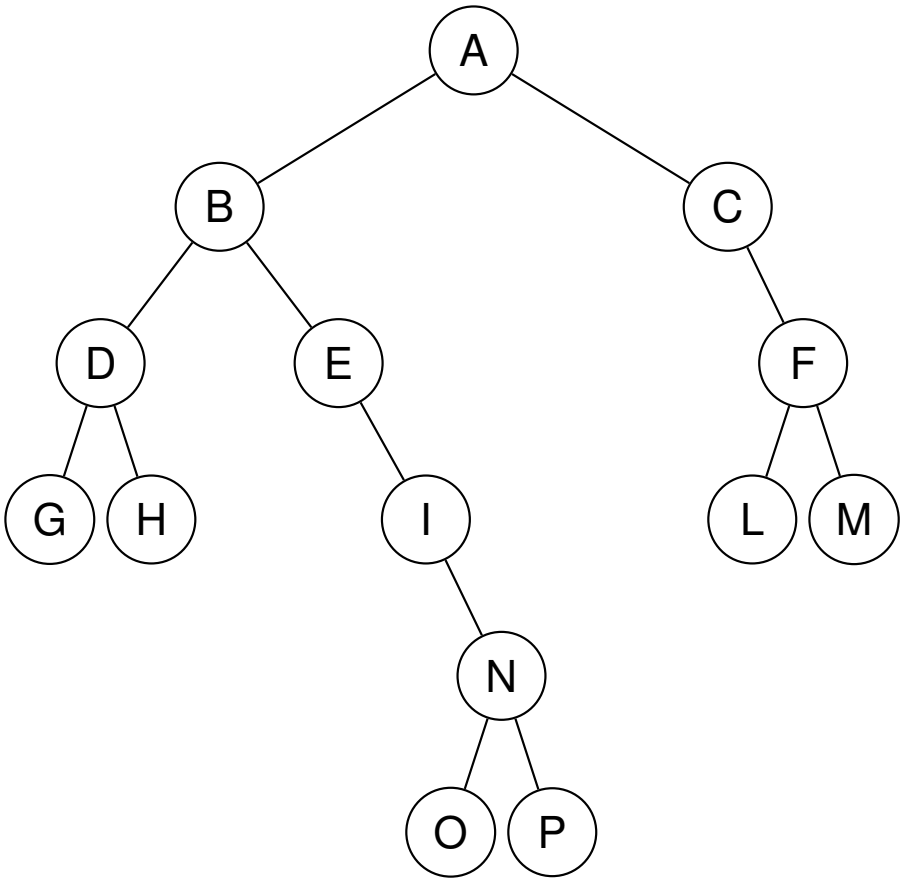
Esempio



Visita inorder

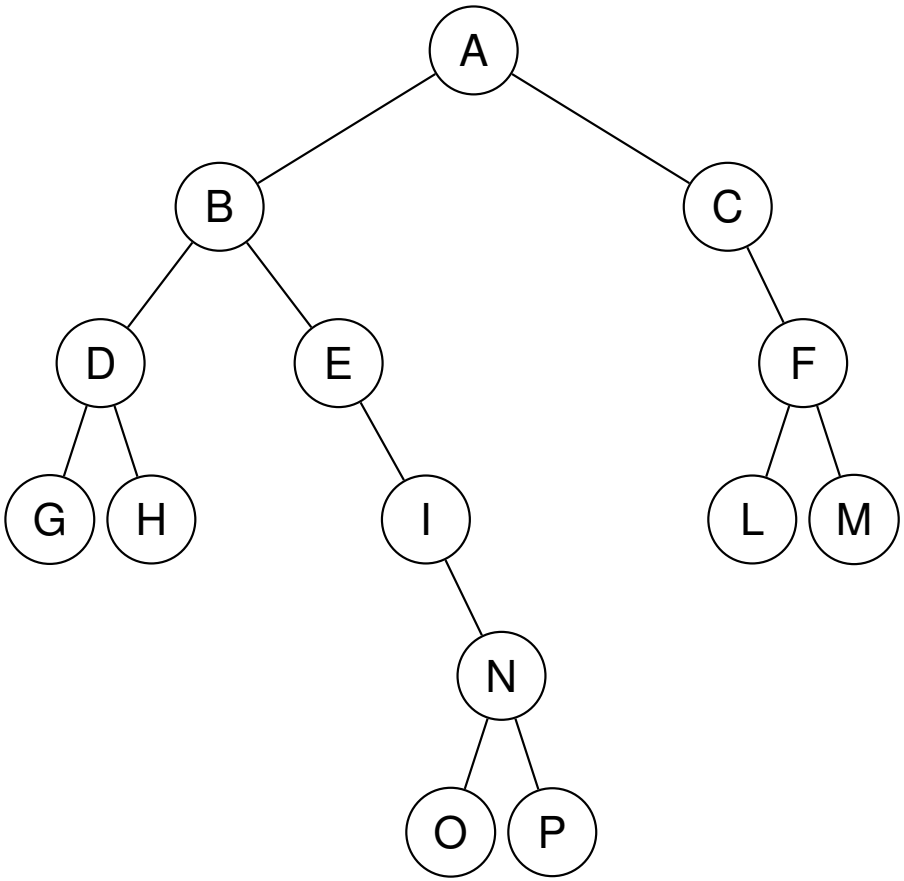
Esempio

- G, D, B, A : G



Visita inorder

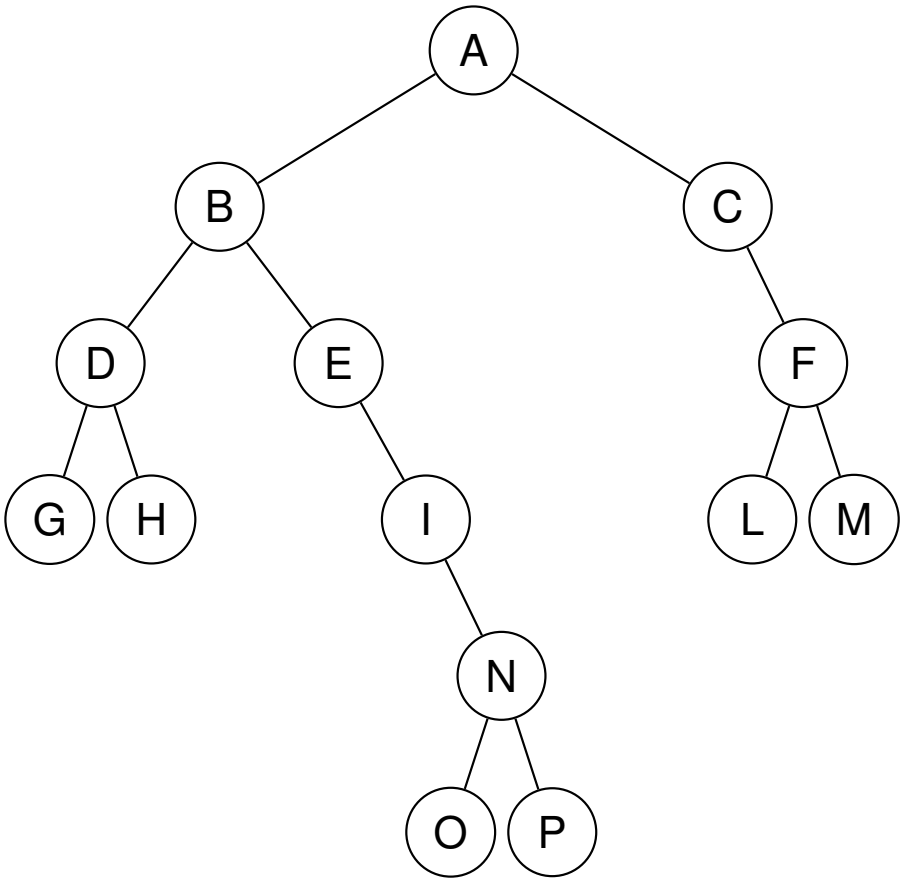
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H

Visita inorder

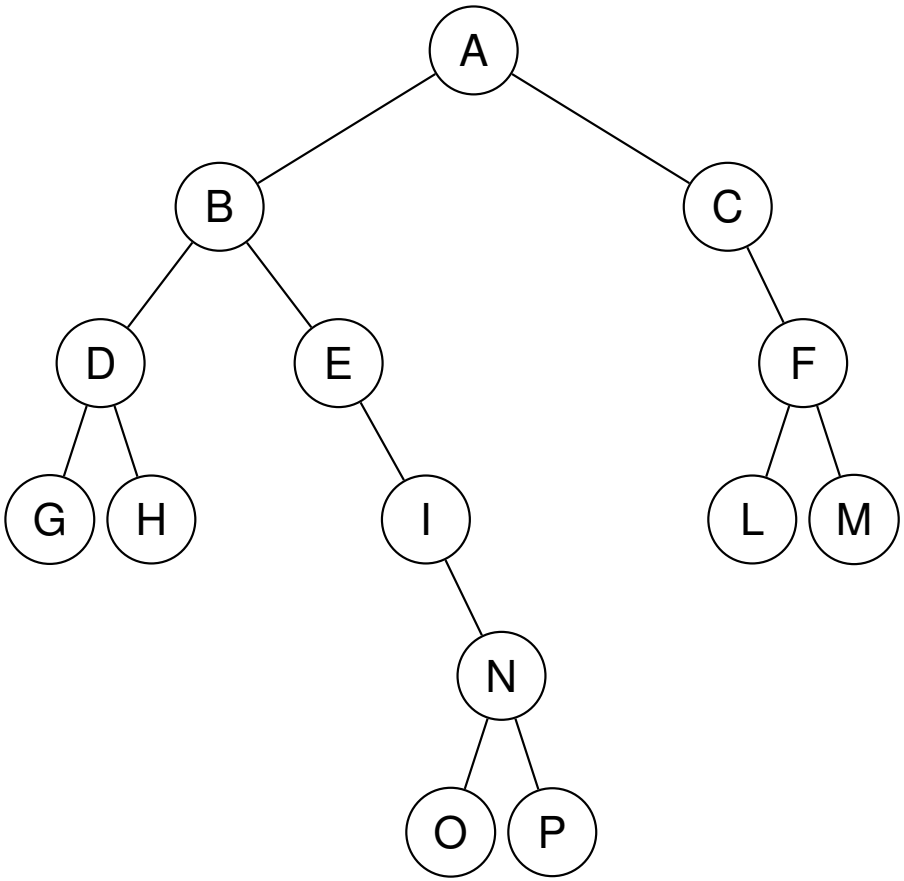
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B

Visita inorder

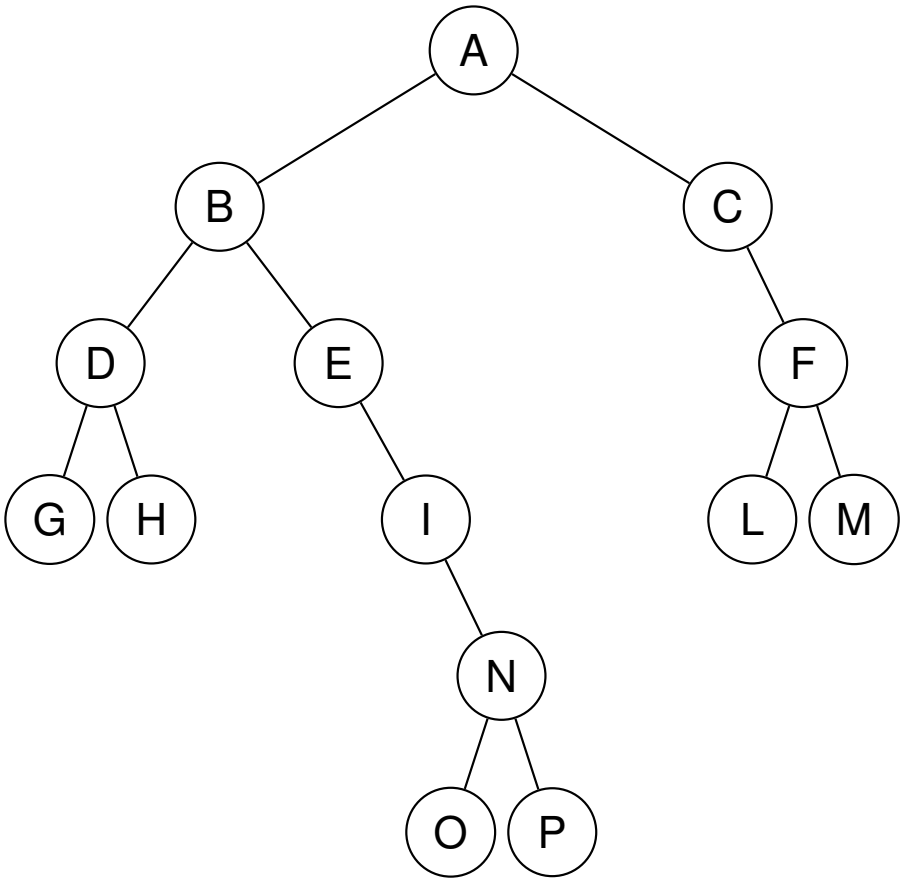
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E

Visita inorder

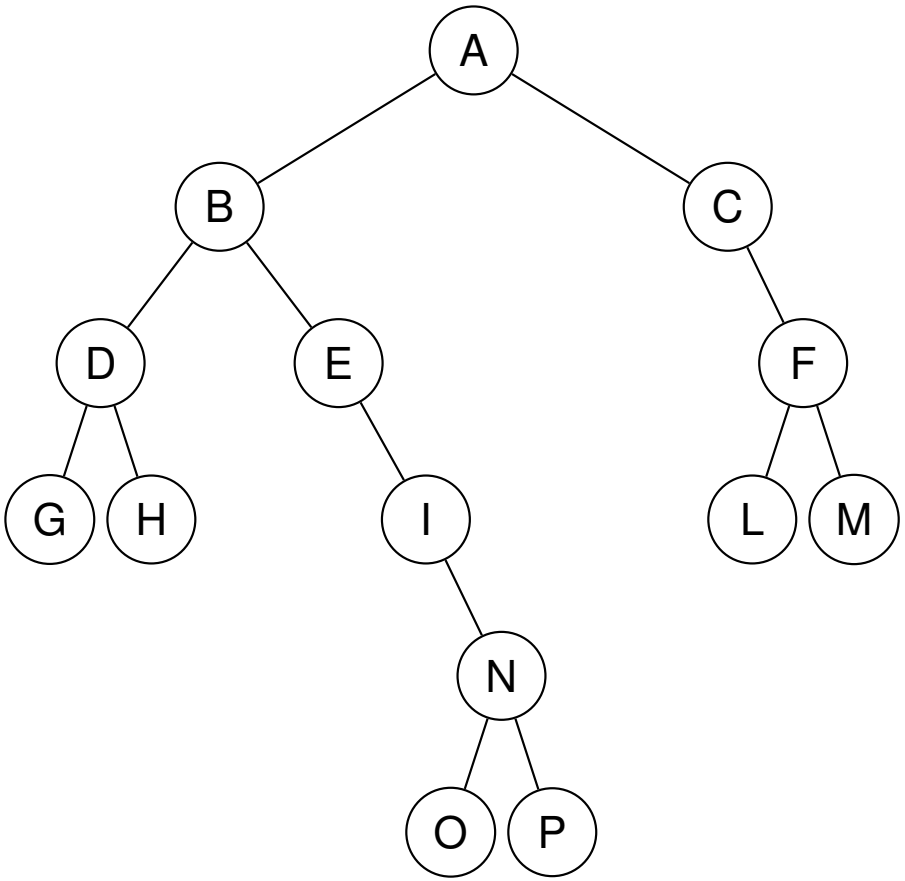
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I

Visita inorder

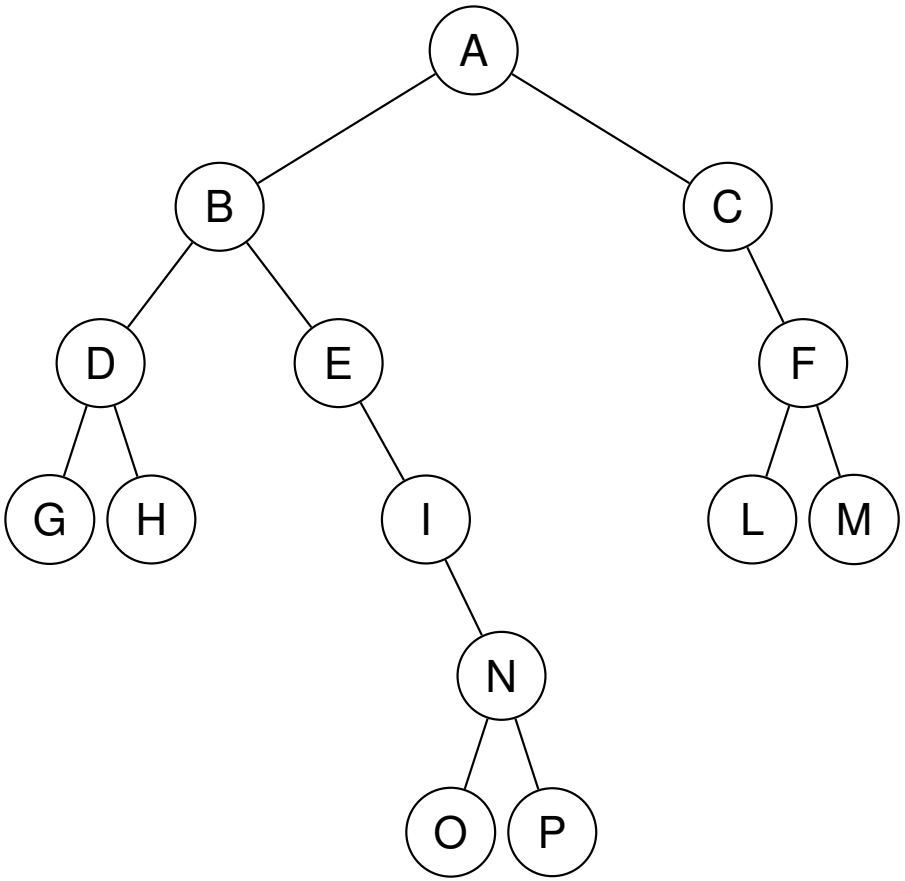
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O

Visita inorder

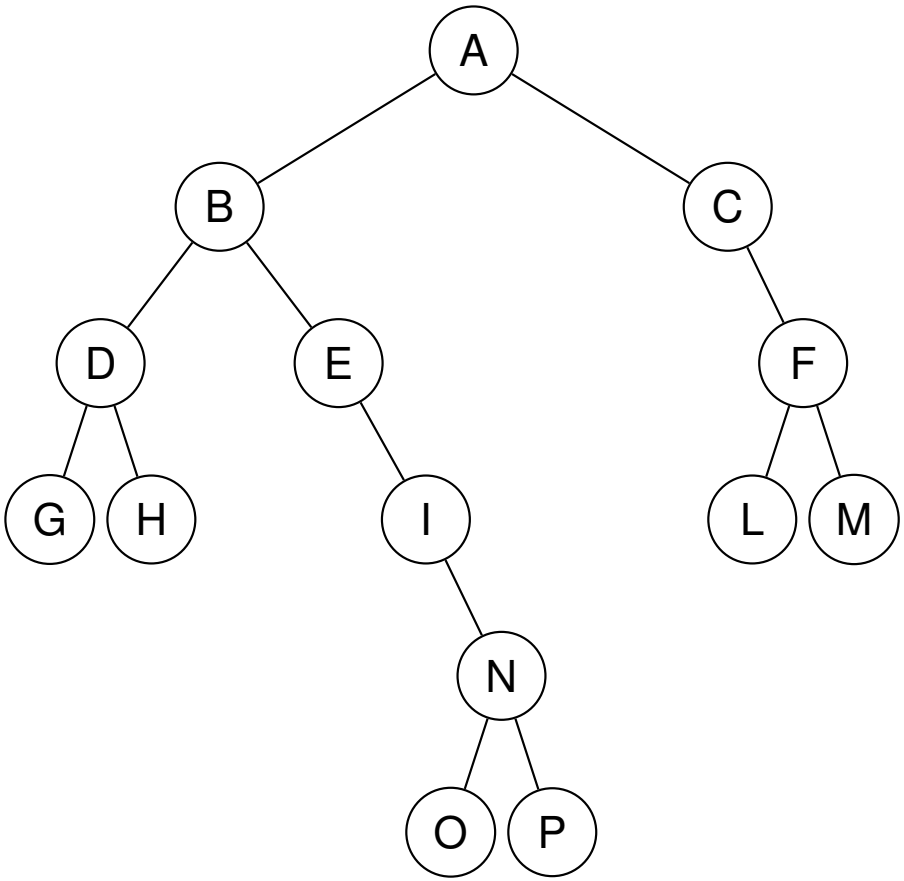
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O
- N,A : N

Visita inorder

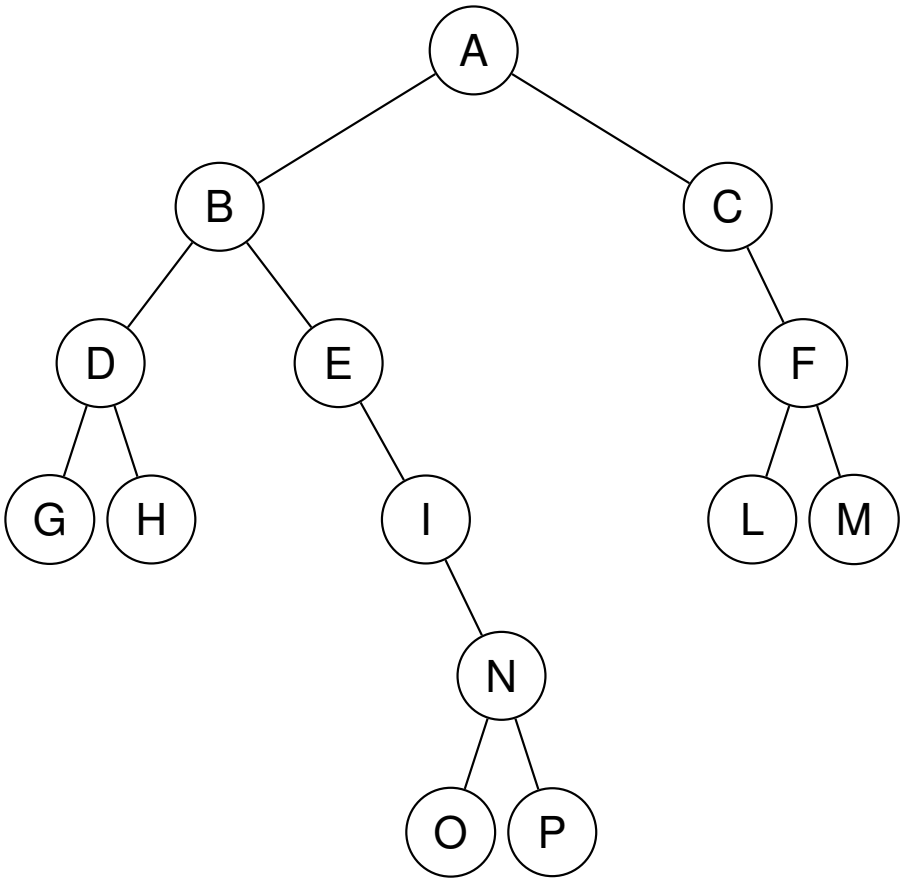
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O
- N,A : N
- P,A : P

Visita inorder

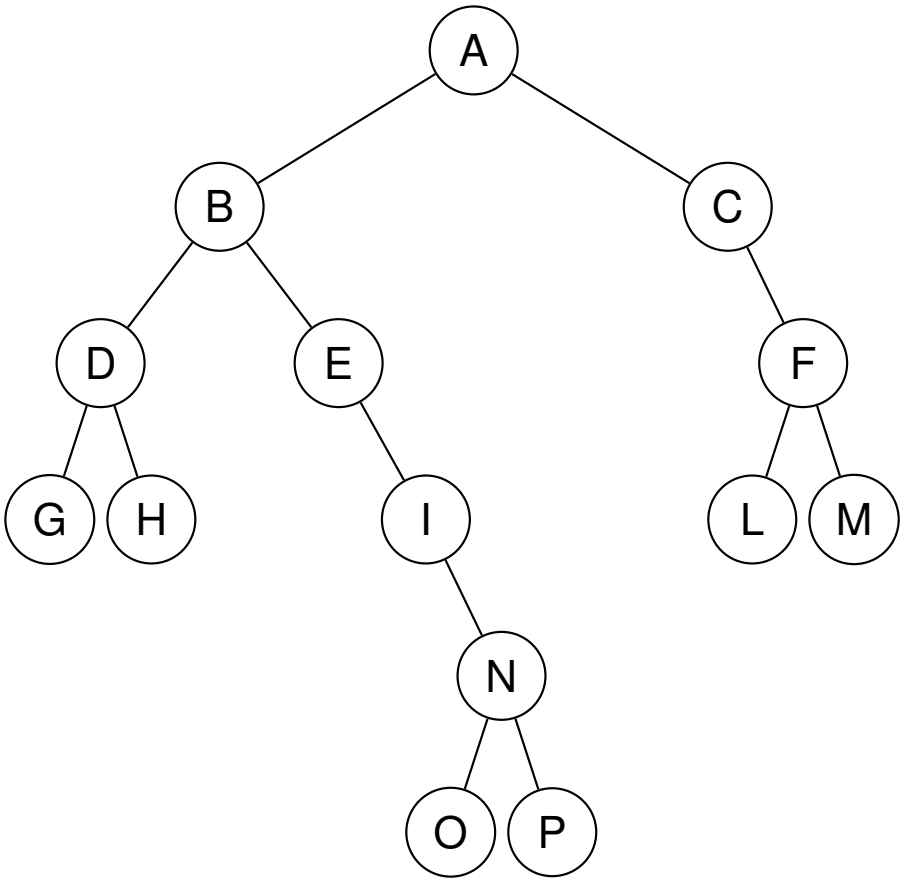
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O
- N,A : N
- P,A : P
- A : A

Visita inorder

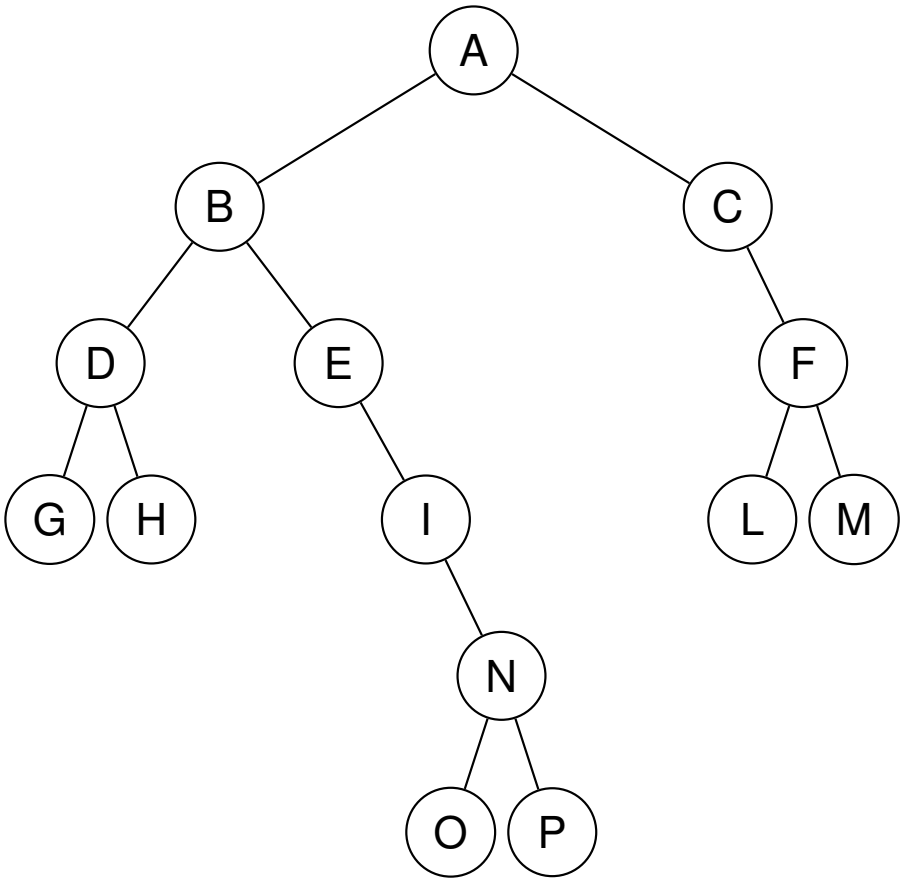
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O
- N,A : N
- P,A : P
- A : A
- C : C

Visita inorder

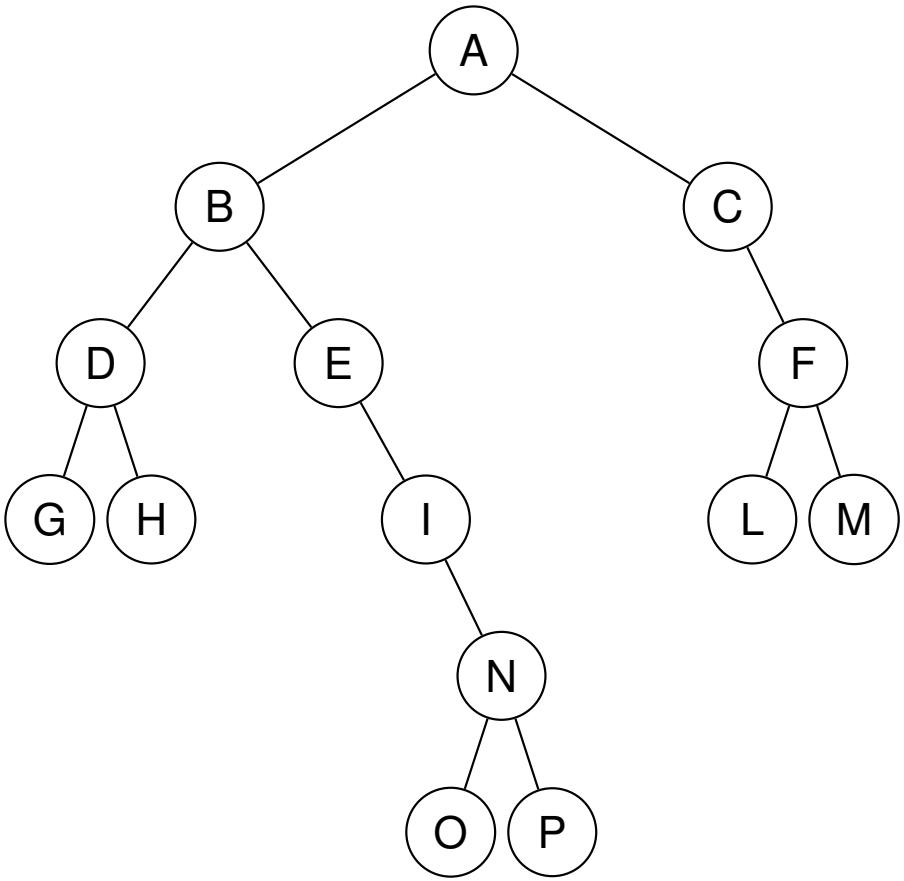
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O
- N,A : N
- P,A : P
- A : A
- C : C
- L,F : L

Visita inorder

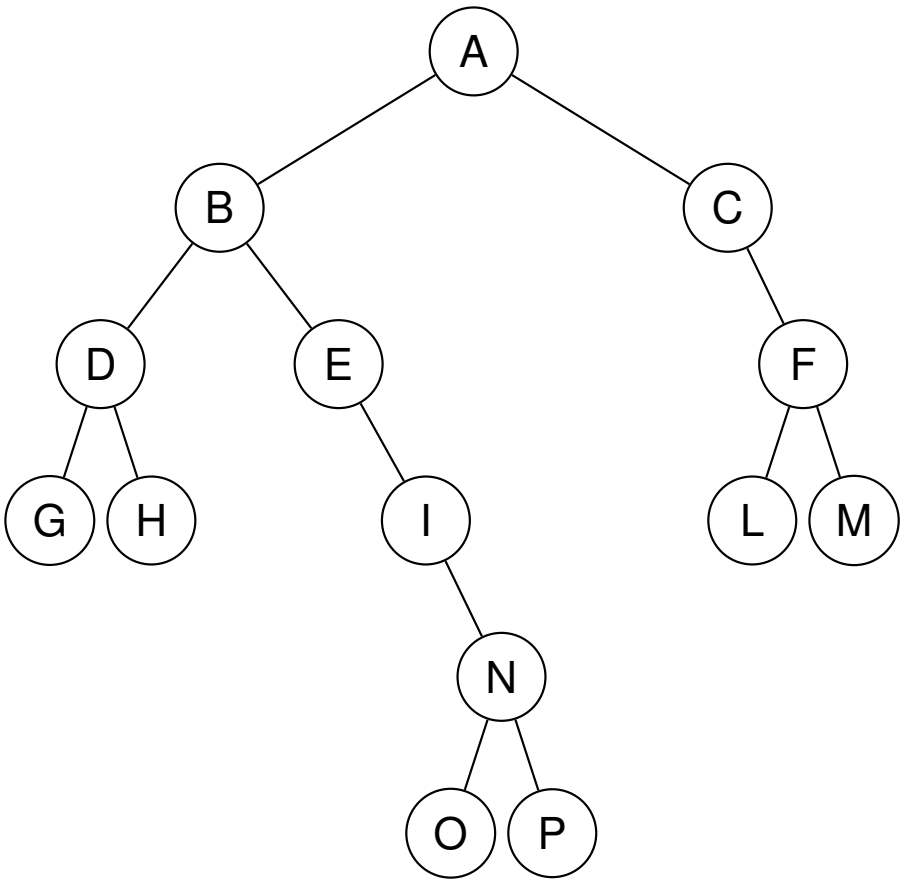
Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O
- N,A : N
- P,A : P
- A : A
- C : C
- L,F : L
- F : F

Visita inorder

Esempio



- G, D, B, A : G
- D, B, A : D
- H, B, A : H
- B,A : B
- E,A : E
- I,A : I
- O,N,A : O
- N,A : N
- P,A : P
- A : A
- C : C
- L,F : L
- F : F
- M : M e termino

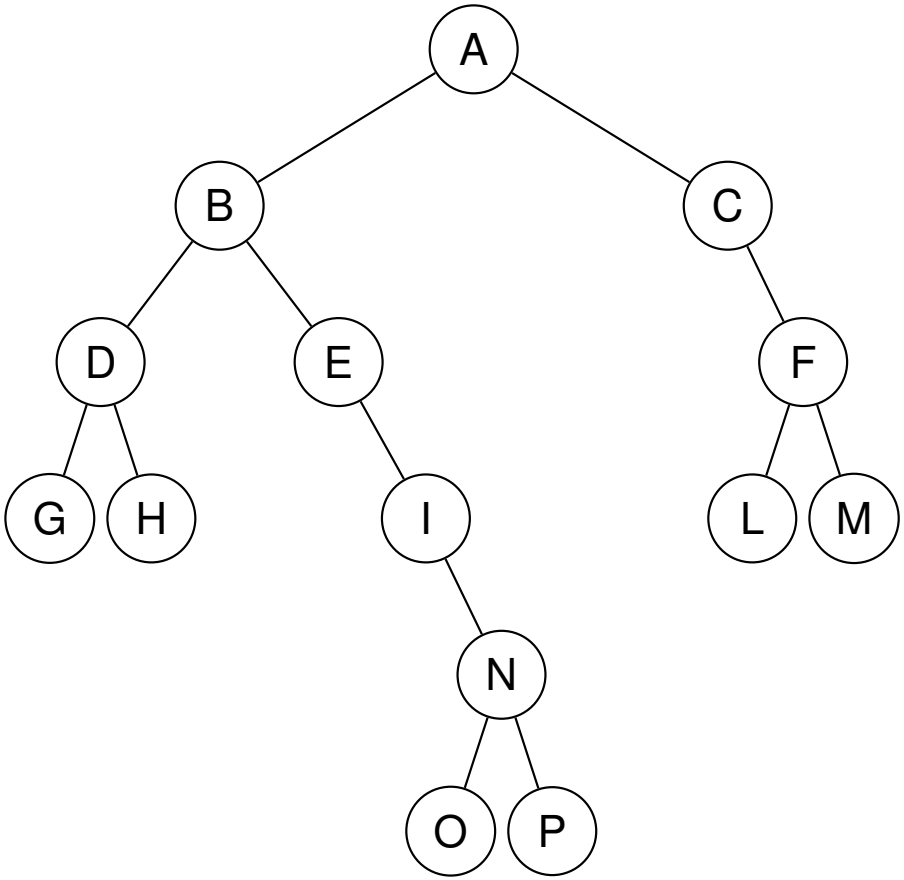
Visita postorder

- È un poi' più complicato, perché la strategia cambia a seconda che stia risalendo l'albero da destra o da sinistra: per capirlo devo tenere l'ultimo nodo restituito, che chiameremo `current`.
- La visita di un nodo implica fare il pop del nodo e poi scendere a sinistra con `searchLeftMostList`: quando trova un nodo col figlio sinistro vuoto, salta al destro e rincomincia a sinistra, fino a quando non trova una foglia.
- 1. lettura del dato: metodo della classe;
  2. controllo di terminazione: stack vuoto;
  3. successore: ripeti fino a quando non arrivi ad un pop e restituisci:
     - se `current` == figlio sinistro del `top` dello stack, allora scopri il figlio destro;
     - se `current` == figlio destro del `top` dello stack, pop e restituisci
     - altrimenti (foglia), pop e restituisci
- Naturalmente parto dalla radice dell'albero.

Visita postorder

Esempio

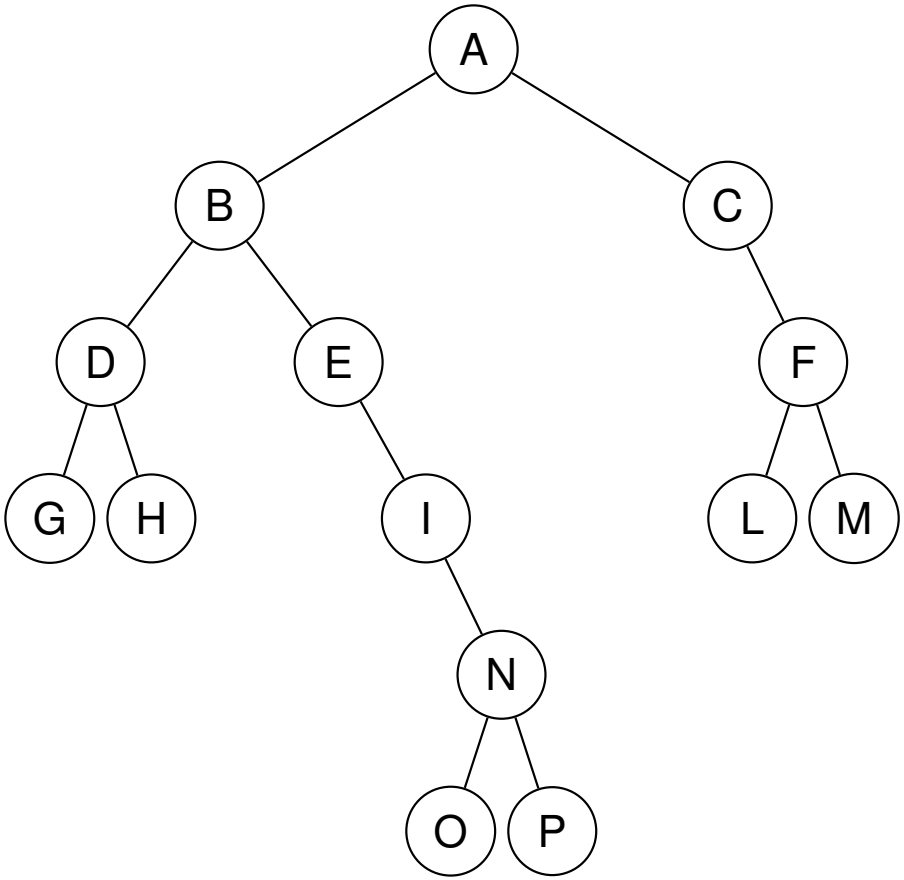
- curr: ϵ; G, D, B, A : G



Visita postorder

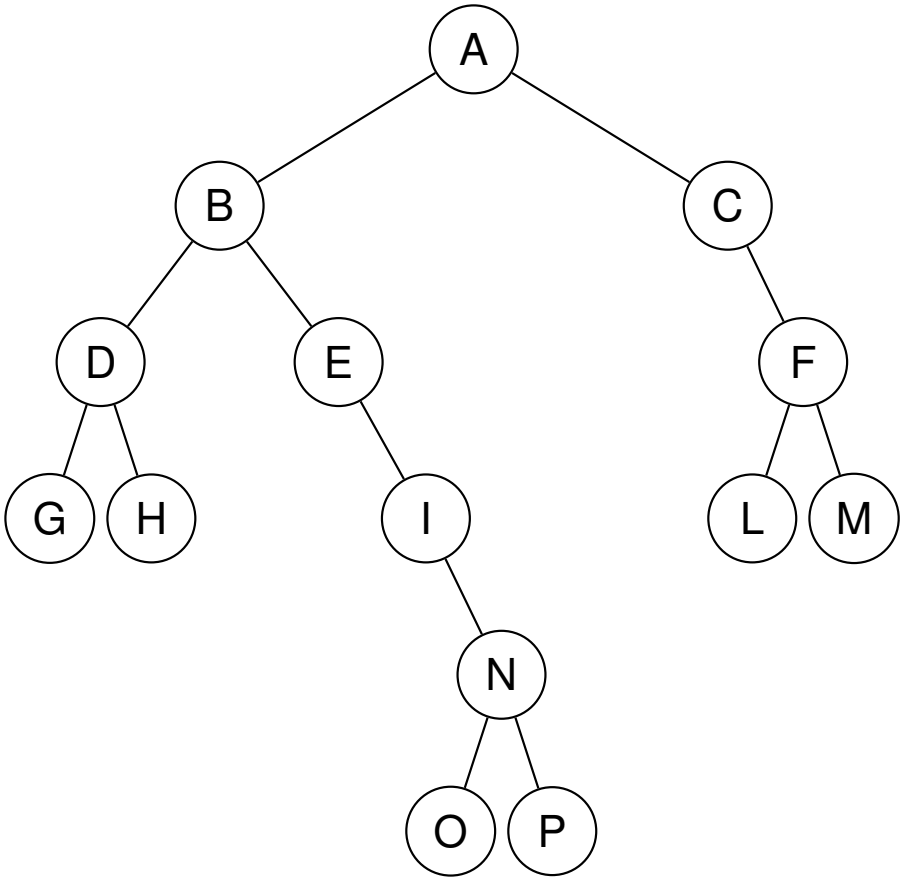
Esempio

- curr: ϵ; | G, D, B, A | : G
- curr: G; | H, D, B, A | : H



Visita postorder

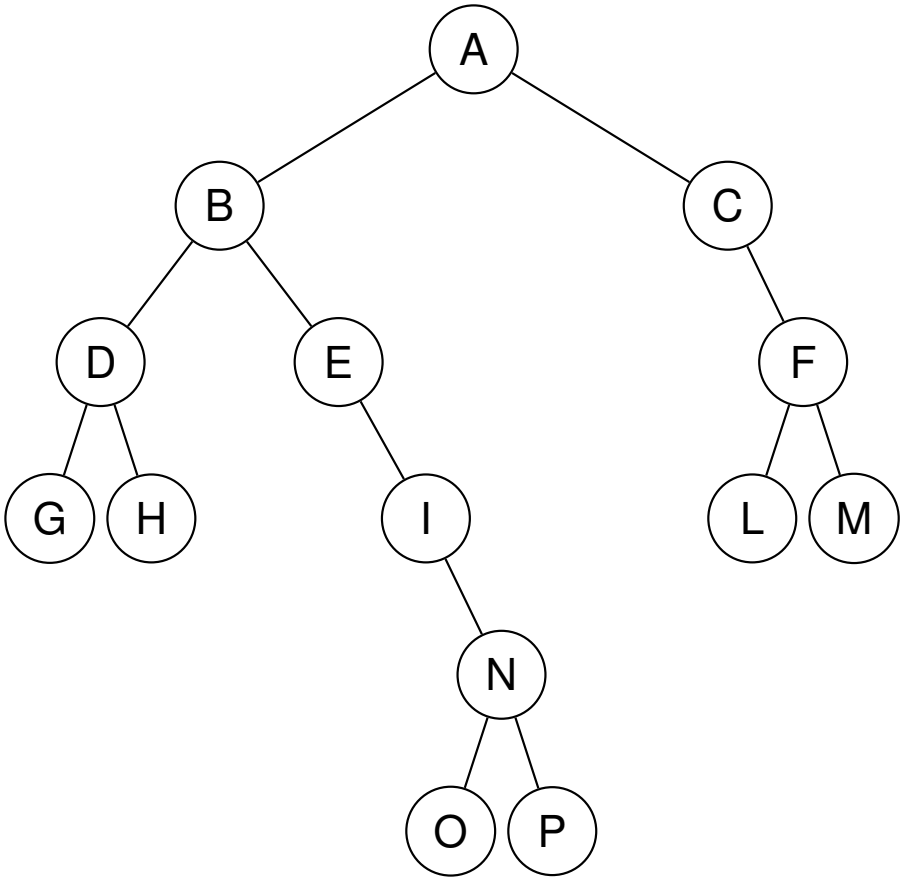
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D

Visita postorder

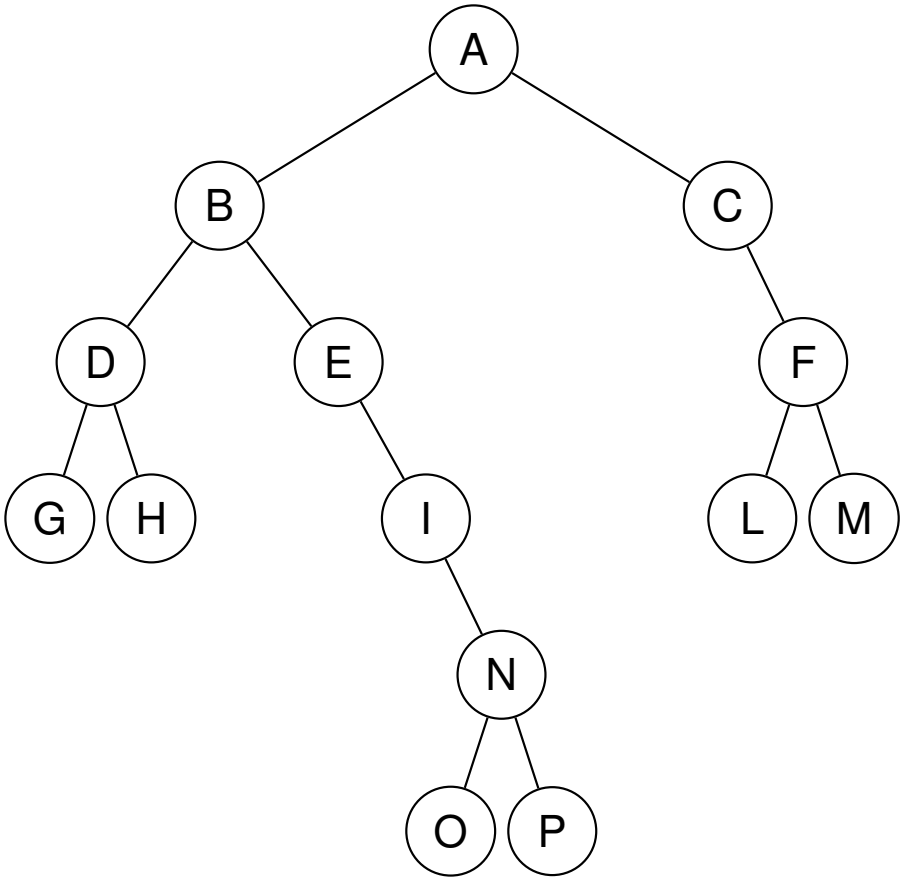
Esempio



- curr: ϵ; | G, D, B, A | : G
- curr: G; | H, D, B, A | : H
- curr: H; | D, B, A | : D
- curr: D; | O, N, I, E, B,A | : O

Visita postorder

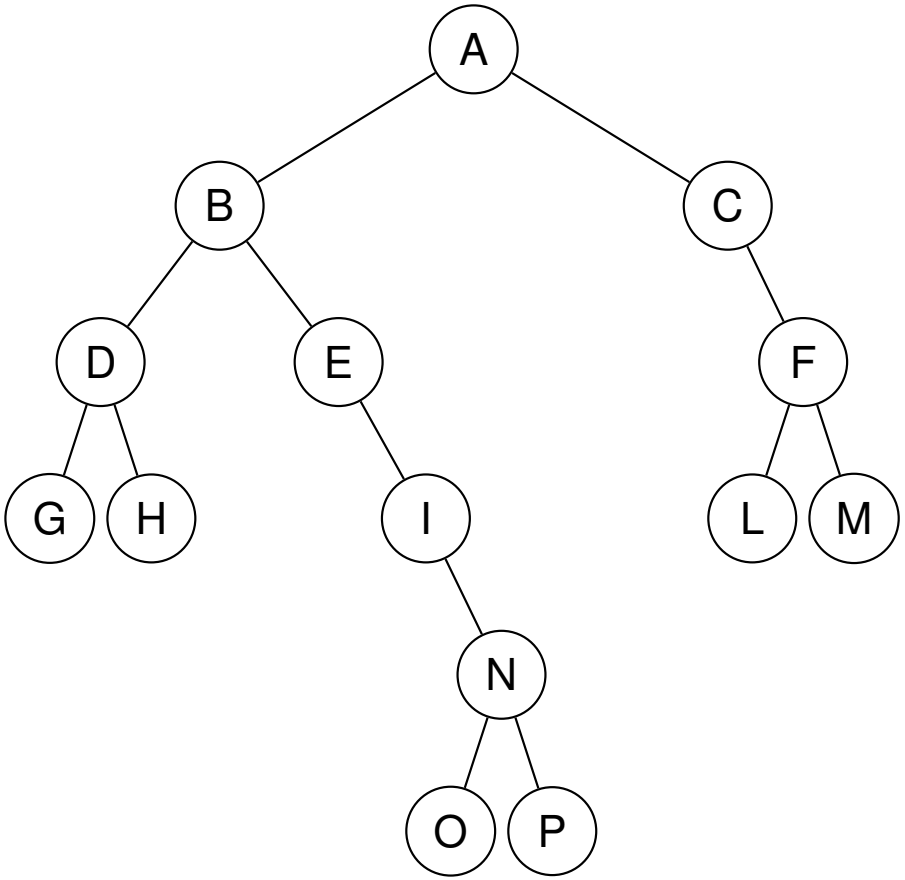
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P

Visita postorder

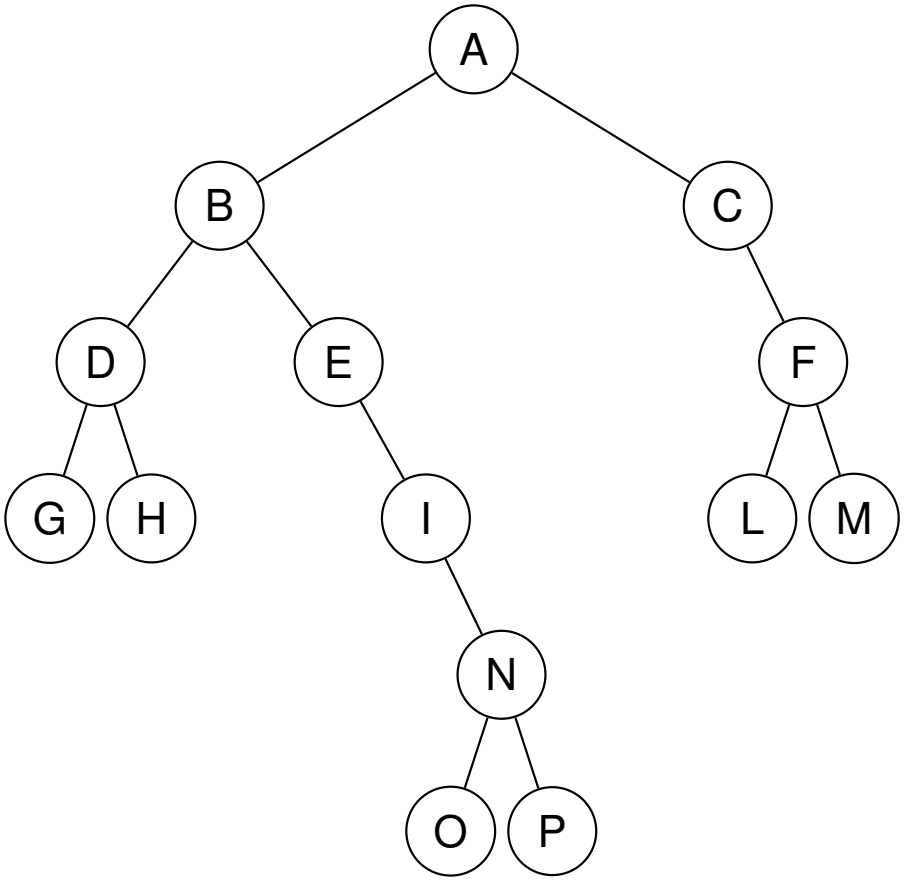
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N

Visita postorder

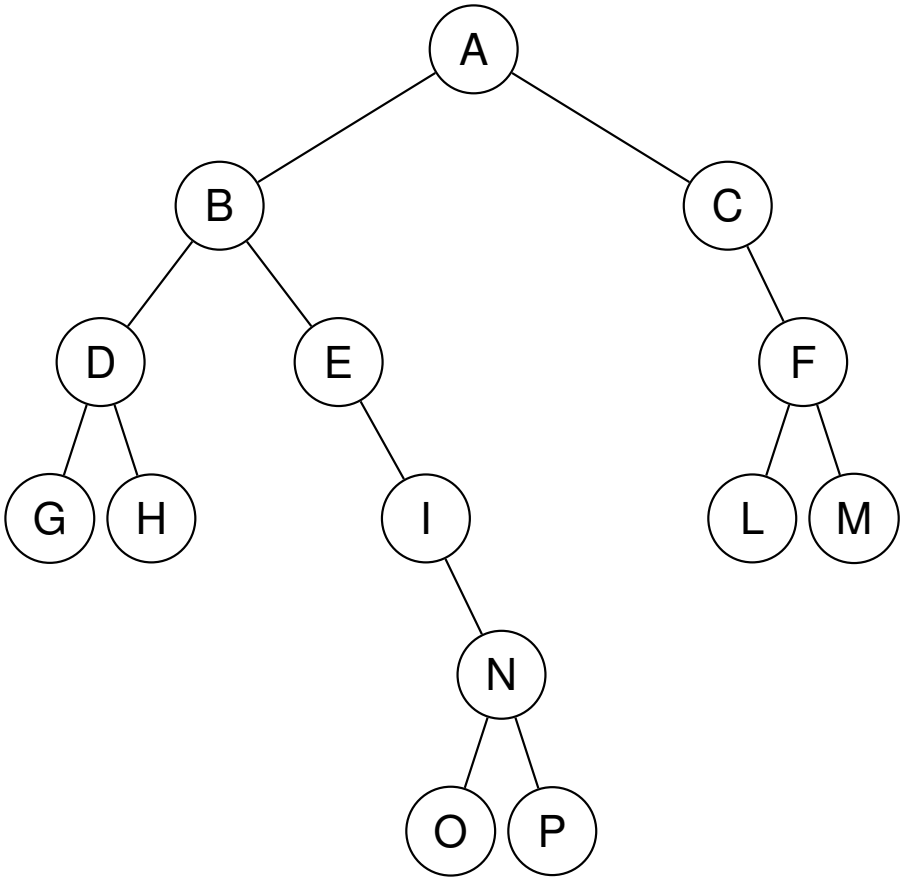
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I

Visita postorder

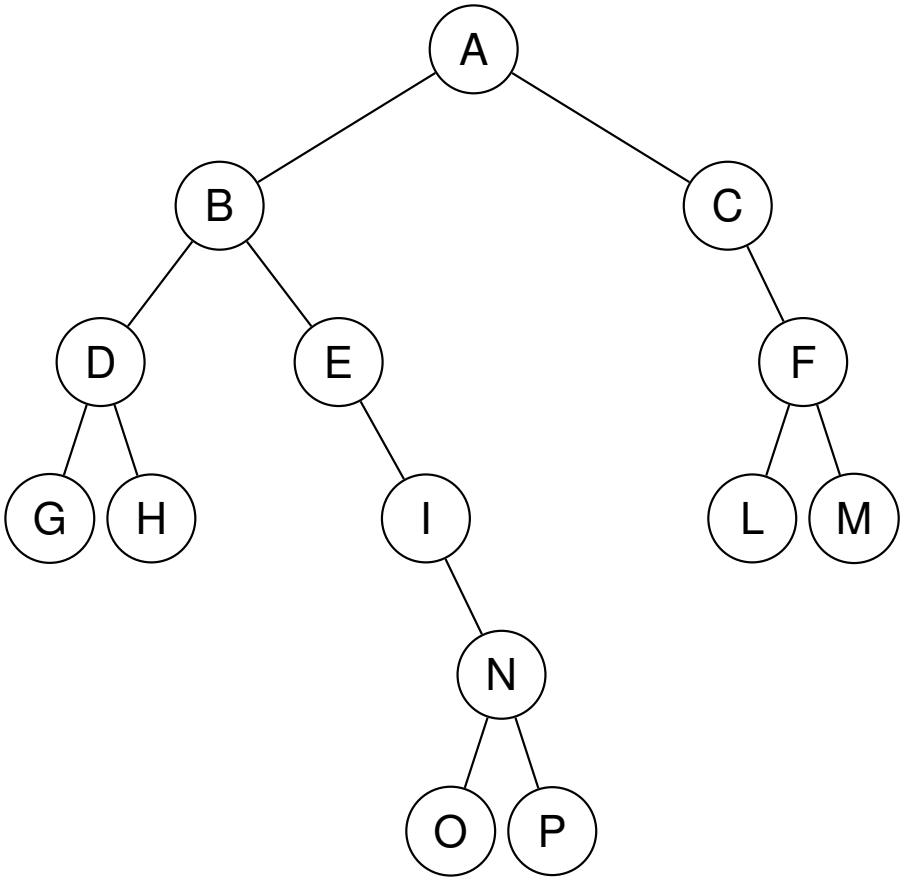
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I
- curr: I E,B,A : E

Visita postorder

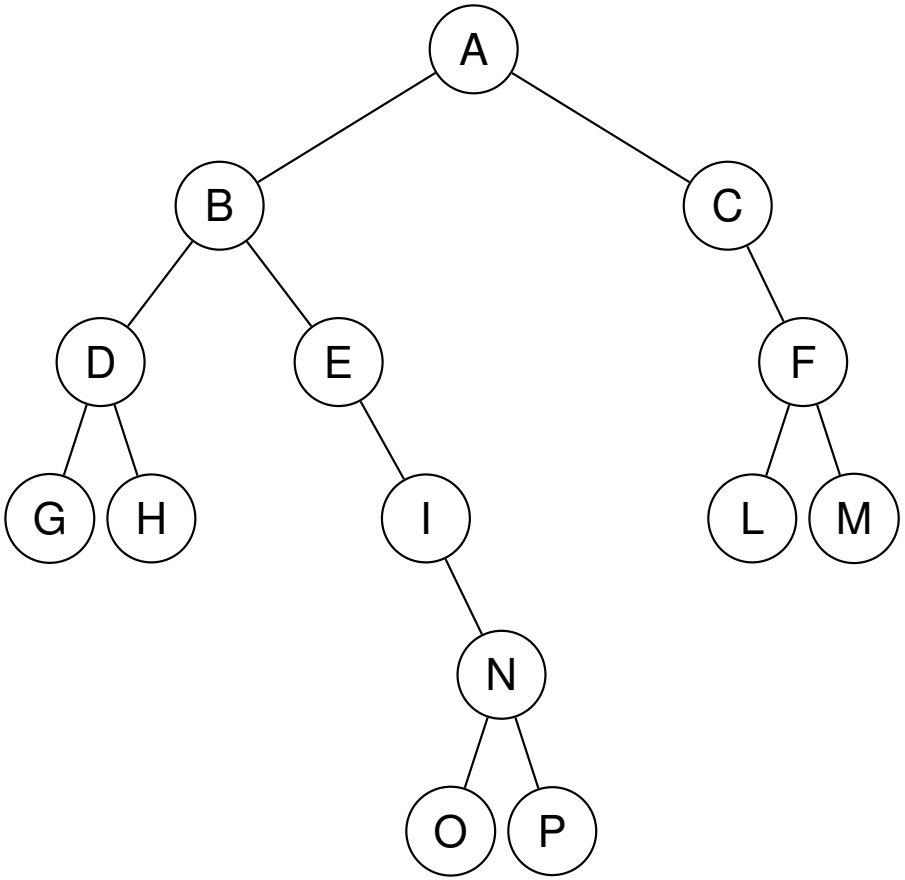
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I
- curr: I E,B,A : E
- curr: E B,A : B

Visita postorder

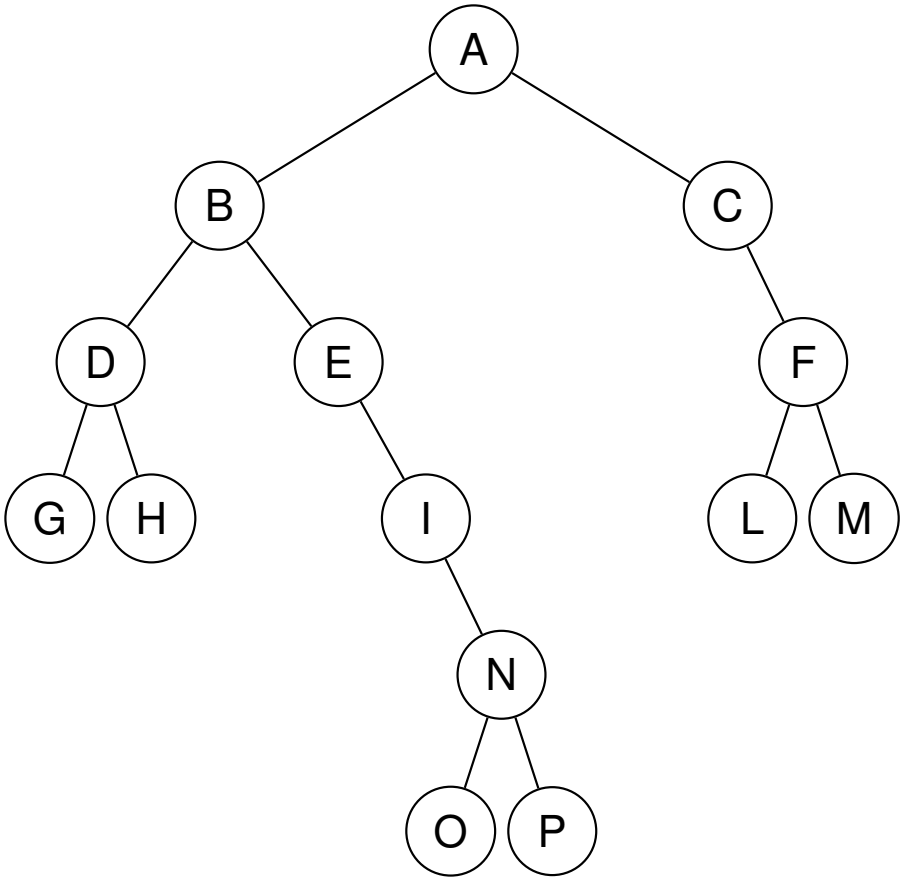
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I
- curr: I E,B,A : E
- curr: E B,A : B
- curr:B L,F,C,A : L

Visita postorder

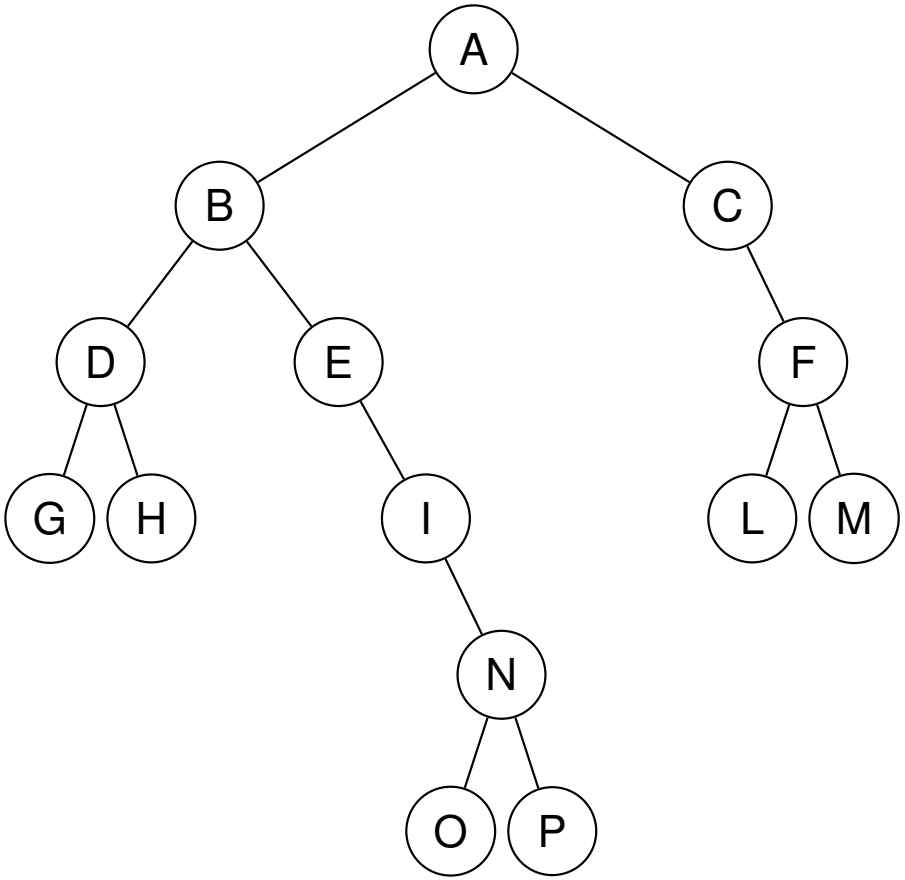
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I
- curr: I E,B,A : E
- curr: E B,A : B
- curr:B L,F,C,A : L
- curr:L M,F,C,A : M

Visita postorder

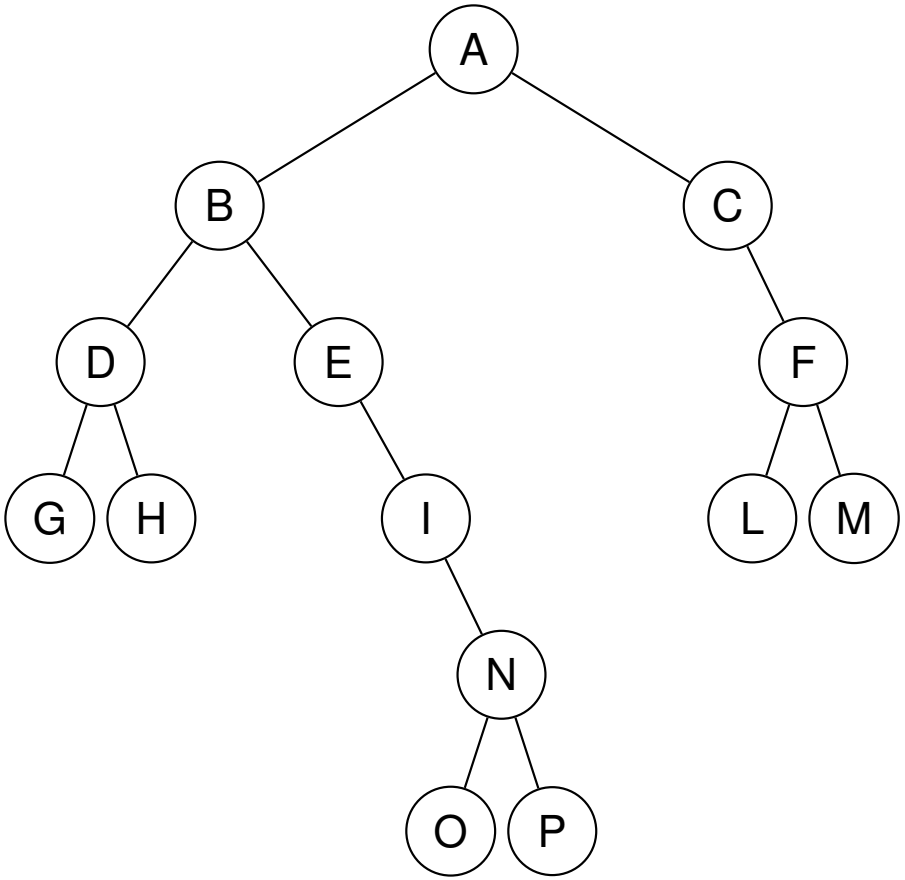
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I
- curr: I E,B,A : E
- curr: E B,A : B
- curr:B L,F,C,A : L
- curr:L M,F,C,A : M
- curr:M F,C,A : F

Visita postorder

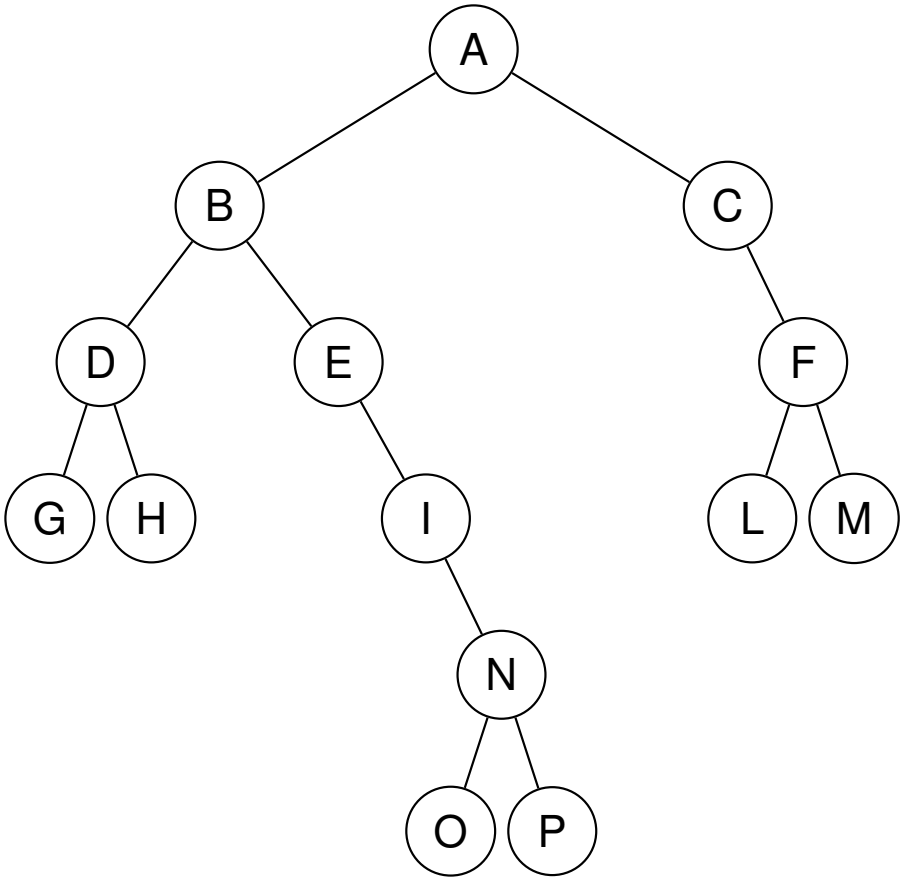
Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I
- curr: I E,B,A : E
- curr: E B,A : B
- curr:B L,F,C,A : L
- curr:L M,F,C,A : M
- curr:M F,C,A : F
- curr:F C,A : C

Visita postorder

Esempio



- curr: ϵ; G, D, B, A : G
- curr: G; H, D, B, A : H
- curr: H; D, B, A : D
- curr: D; O, N, I, E, B,A : O
- curr: O P,N,I,E,B,A : P
- curr: P N,I,E,B,A : N
- curr: N I,E,B,A : I
- curr: I E,B,A : E
- curr: E B,A : B
- curr:B L,F,C,A : L
- curr:L M,F,C,A : M
- curr:M F,C,A : F
- curr:F C,A : C
- curr:C A : A e termino