Alberi binari

1. Introduzione e requisiti del problema
2. Specifica
3. Progetto della soluzione
4. Codifica

Requisiti del problema

Si scriva una funzione che data la radice di un albero binario restituisca i valori del cammino esterno e del cammino interno dell'albero stesso.

In questa esercitazione verrà mostrato come navigare in una struttura dati tipo albero binario.

L'esercizio richiede di calcolare
il cammino esterno e il cammino interno
di un albero.

esterno

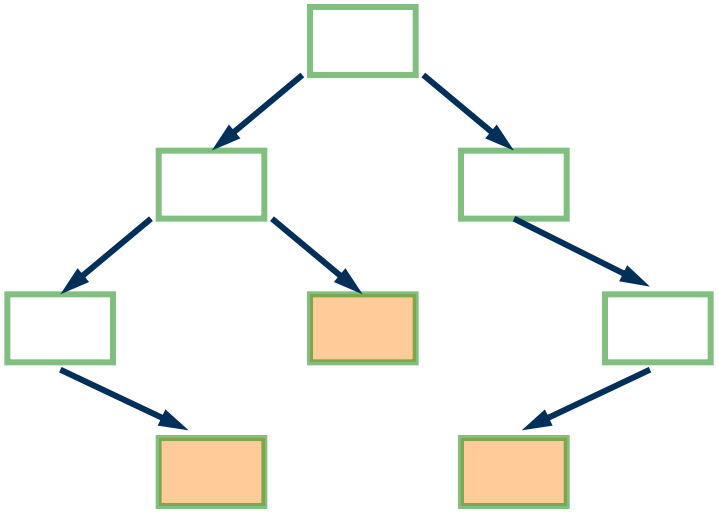
Per cammino esterno di un albero
si intende il valore intero pari alla somma
di tutti i cammini che partono dalla radice
dell'albero e che arrivano alle foglie.

interno

Il cammino interno di un albero
è definito come un valore intero pari alla somma
delle lunghezze di tutti i cammini, che partono
dalla radice dell'albero e che arrivano ai nodi
non-foglia.

Casi di test

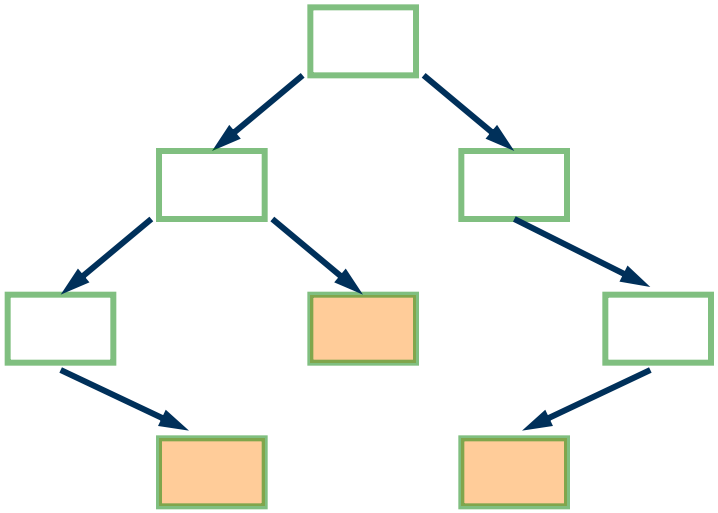
Input:



Output:

```
il cammino esterno ha il valore = 8
il cammino interno ha il valore = 6
```

Alberi binari



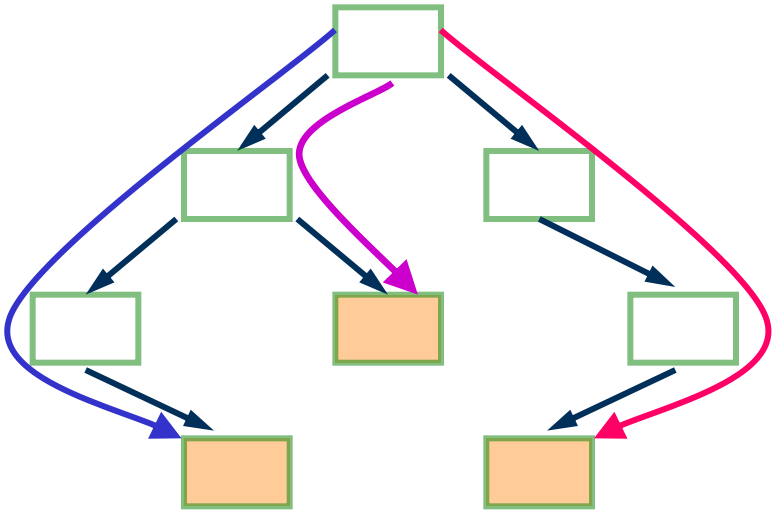
0 (lunghezza del cammino della radice per il nodo radice)

1 (lunghezza del cammino della radice per i nodi di questo livello)

2 (lunghezza del cammino della radice per i nodi di questo livello)

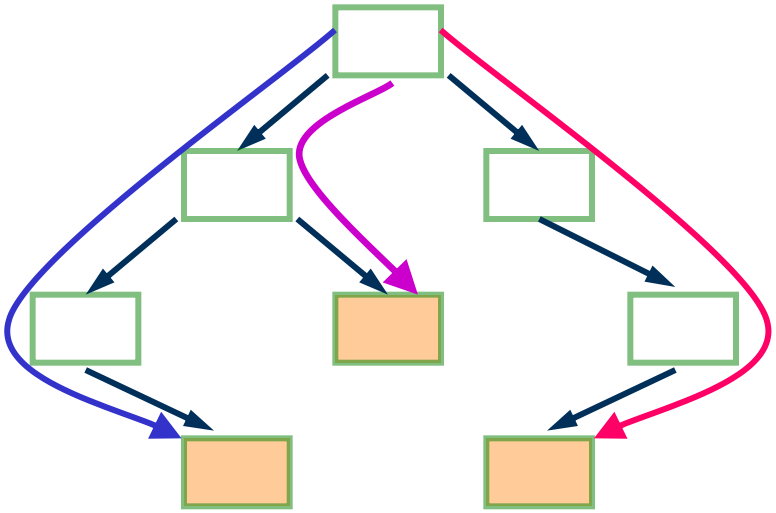
3 (lunghezza del cammino della radice per i nodi di questo livello)

cammino esterno



Totale cammino esterno: 2 + 3 + 3 = 8

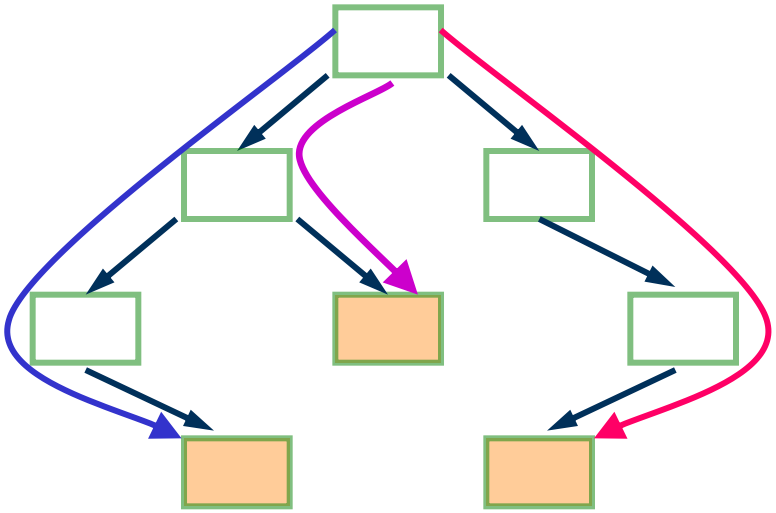
cammino esterno



Cammino=2

Totale cammino esterno: 2 + 3 + 3 = 8

cammino esterno

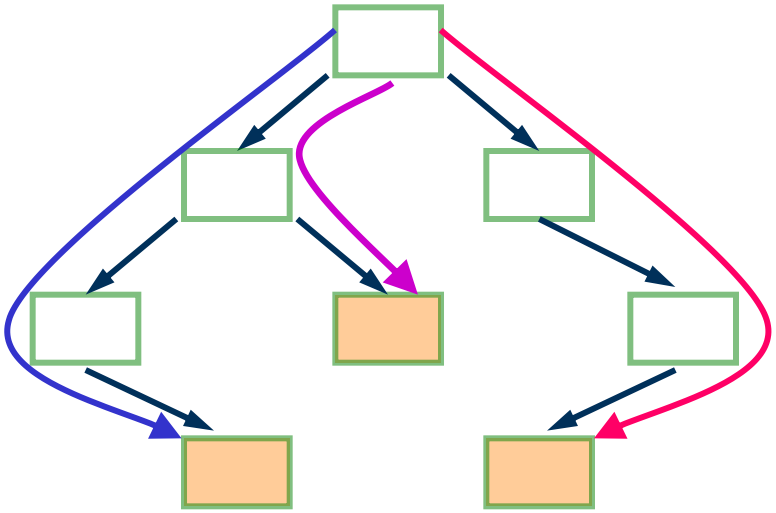


Cammino=2

Cammino=3

Totale cammino esterno: 2 + 3 + 3 = 8

cammino esterno



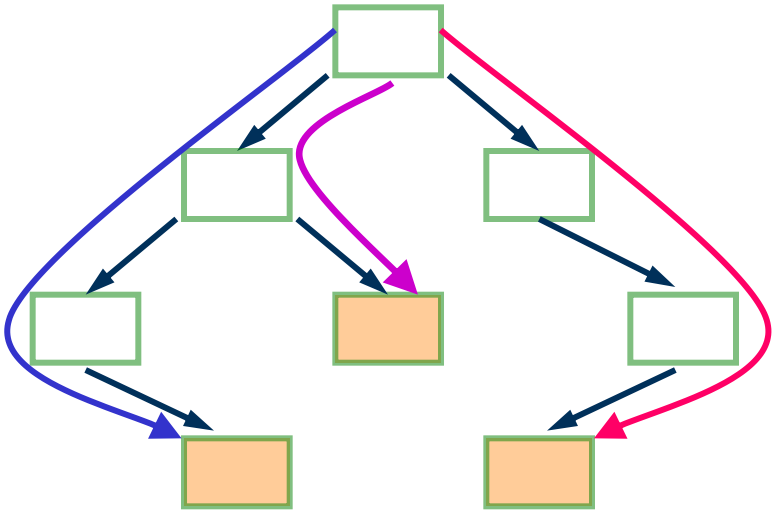
Cammino=2

Cammino=3

Cammino=3

Totale cammino esterno: 2 + 3 + 3 = 8

cammino esterno



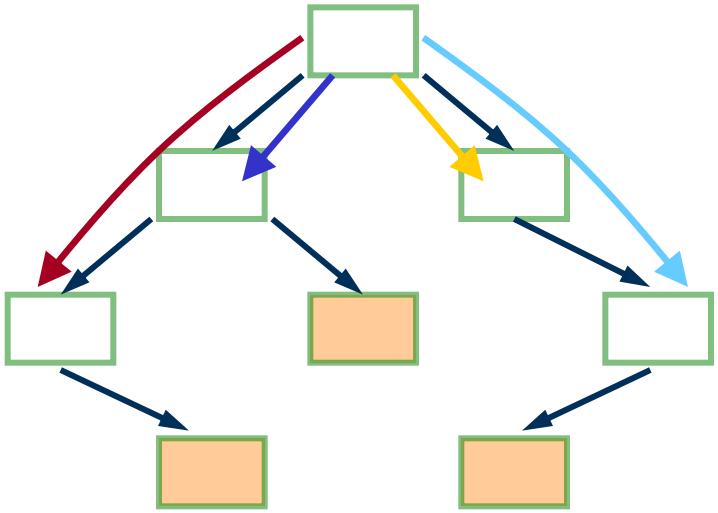
Cammino=2

Cammino=3

Cammino=3

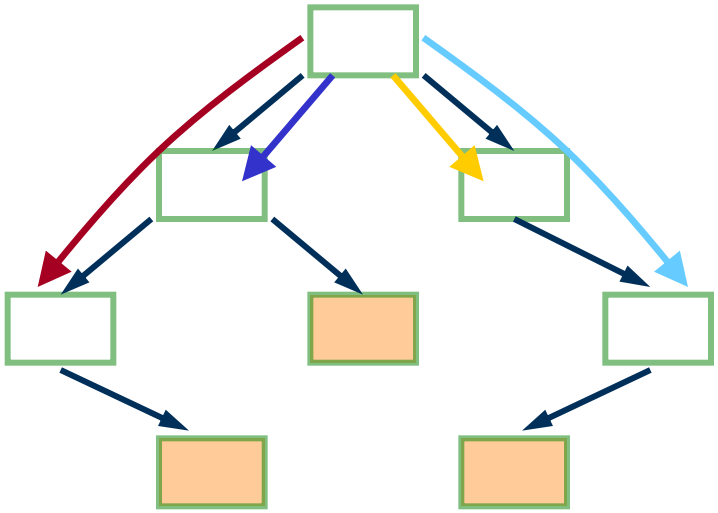
Totale cammino esterno: 2 + 3 + 3 = 8

cammino interno *interno*



Totale cammino interno: 1 + 2 + 1 + 2 = 6

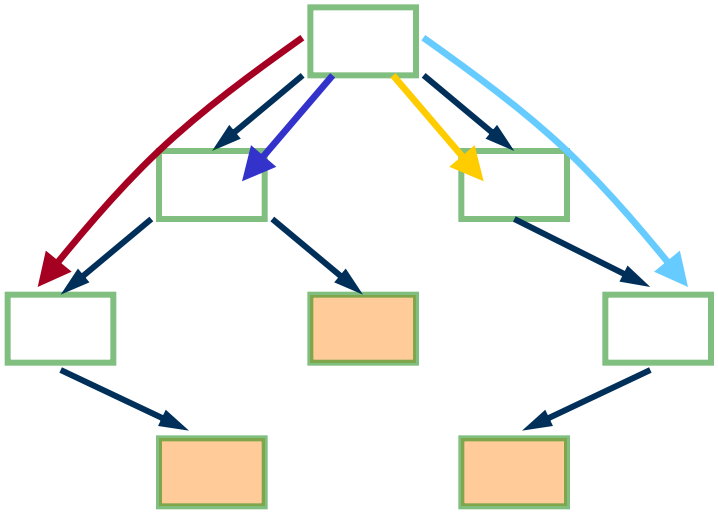
cammino interno



Cammino=1

Totale cammino interno: 1 + 2 + 1 + 2 = 6

cammino interno

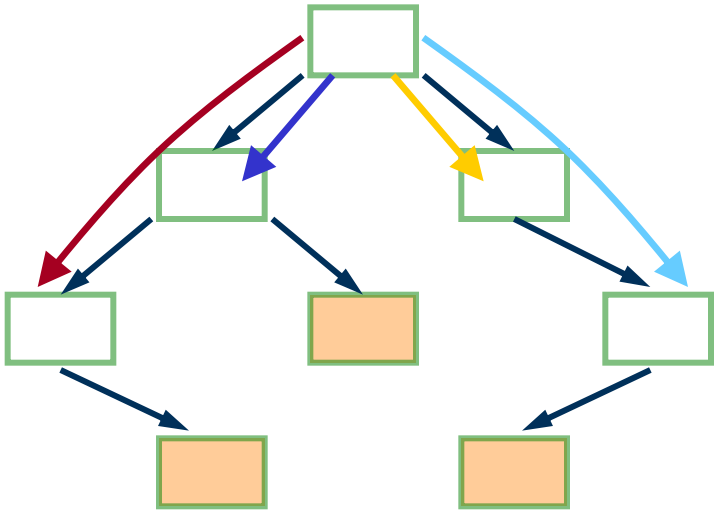


Cammino=1

Cammino=2

Totale cammino interno: 1 + 2 + 1 + 2 = 6

cammino interno



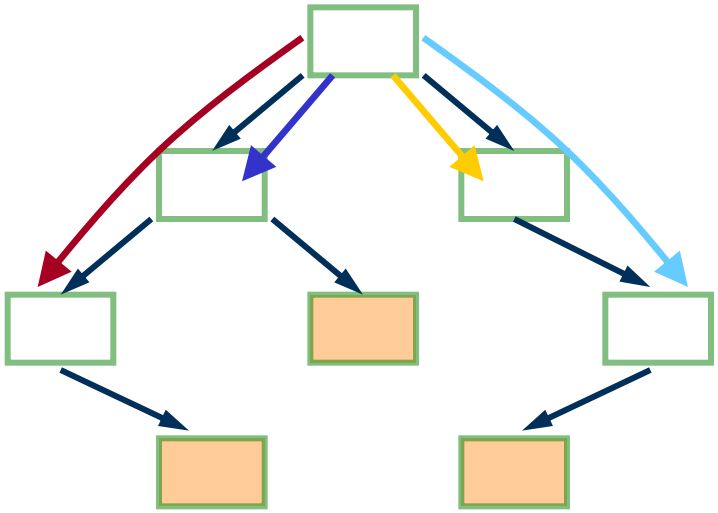
Cammino=1

Cammino=2

Cammino=1

Totale cammino interno: 1 + 2 + 1 + 2 = 6

cammino interno



Cammino=1

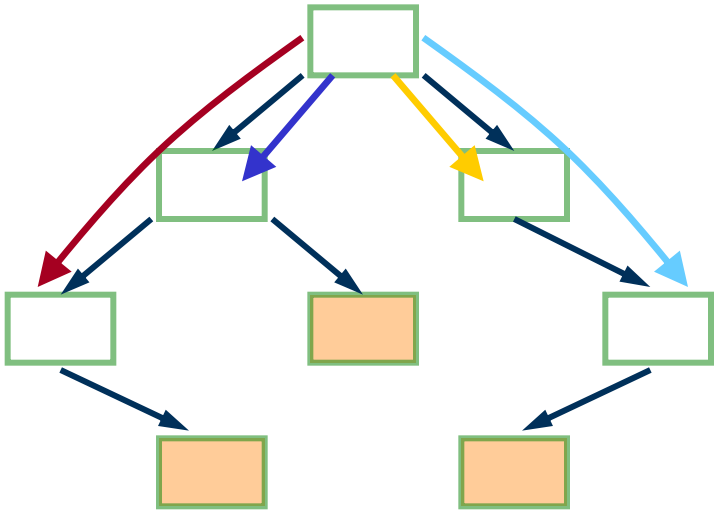
Cammino=2

Cammino=1

Cammino=2

Totale cammino interno: 1 + 2 + 1 + 2 = 6

cammino interno



Cammino=1

Cammino=2

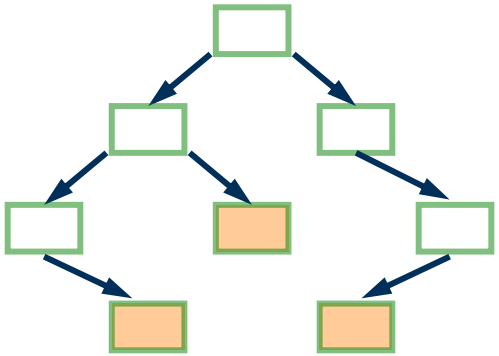
Cammino=1

Cammino=2

Totale cammino interno: 1 + 2 + 1 + 2 = 6

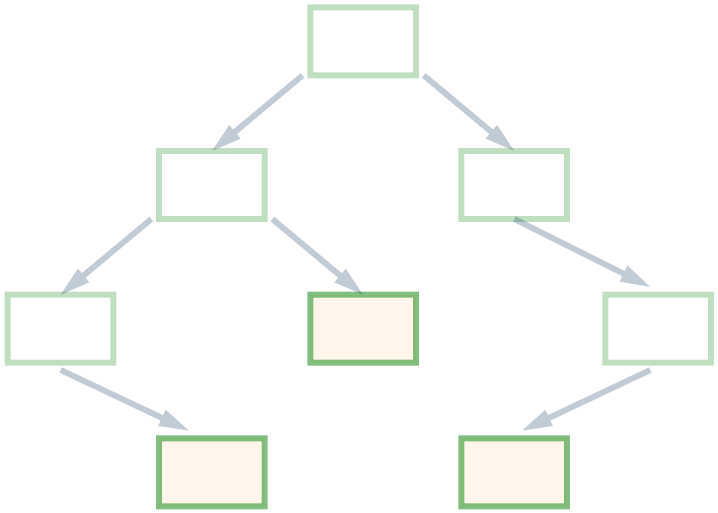
Esempio di definizione di un albero binario

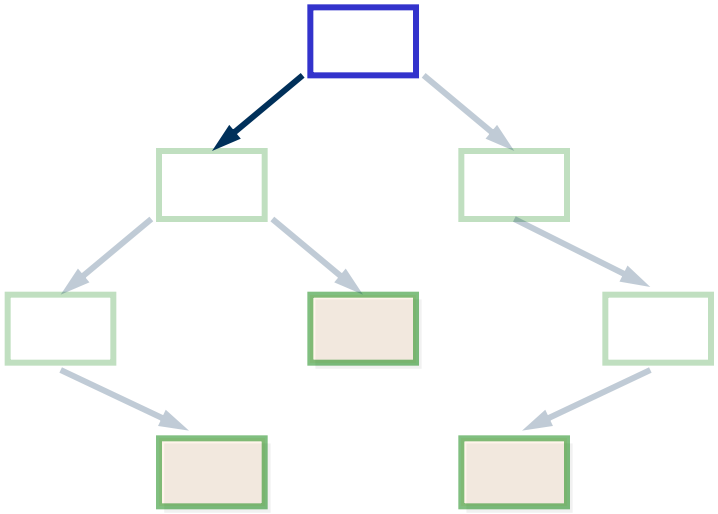
```
struct Tree {
  int info;
  Tree *leftptr;
  Tree *rightptr;
};
```

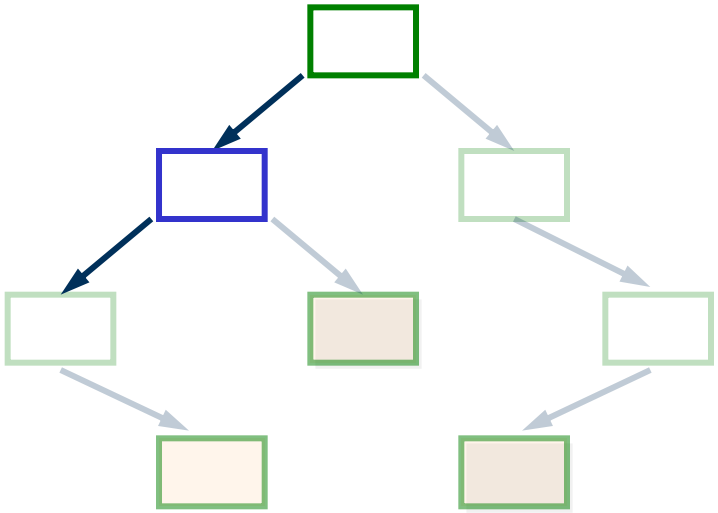


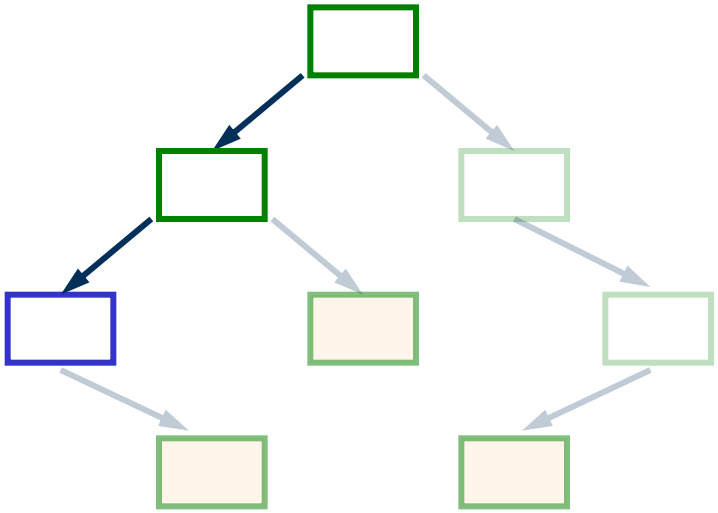
Per calcolare i cammini è necessario definire una funzione che realizzi la navigazione tramite una funzione ricorsiva.

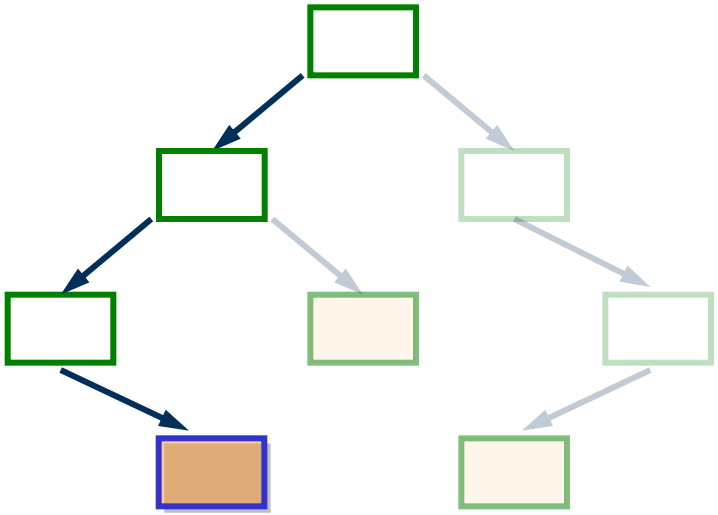
Tale funzione riceve come ingresso i sotto alberi puntati dai puntatori del nodo corrente.

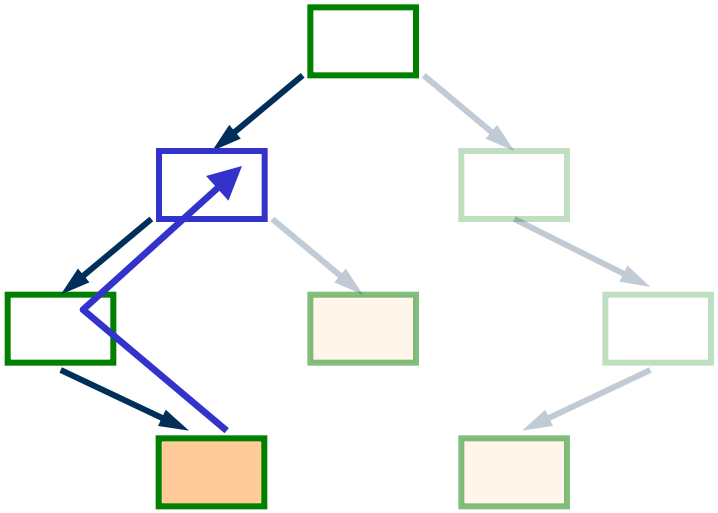


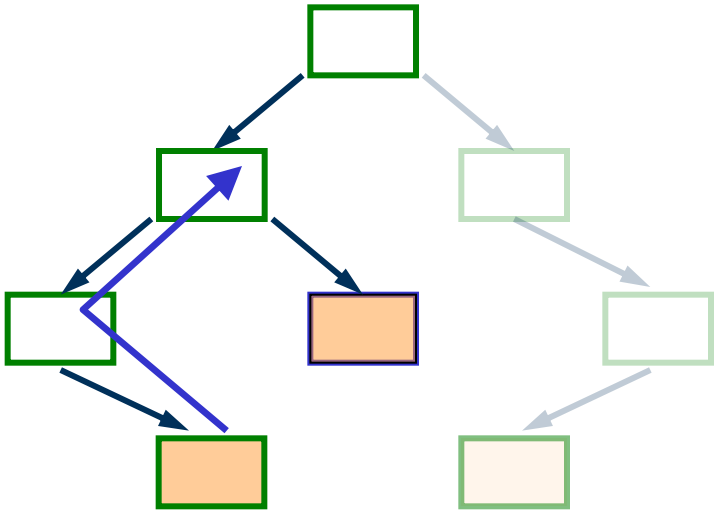


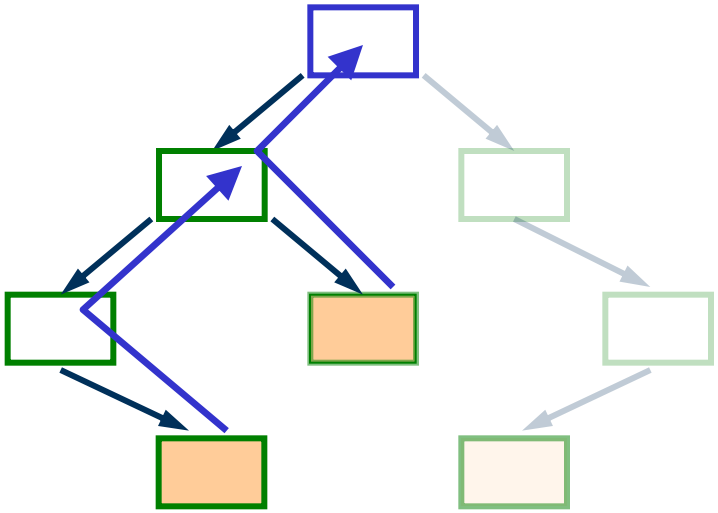


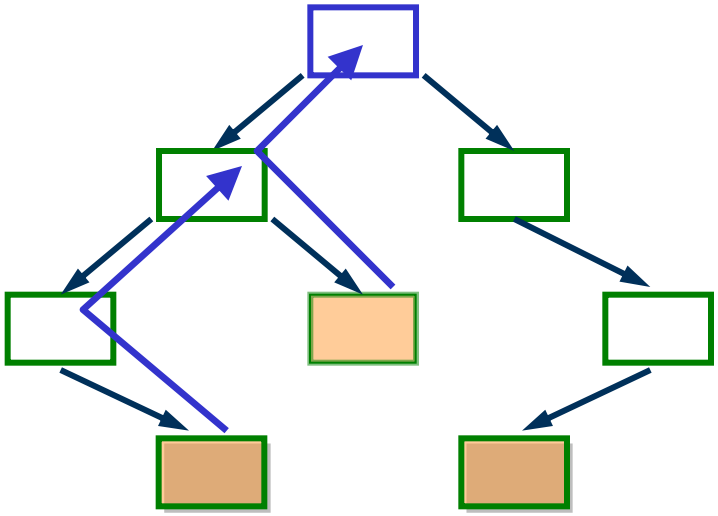












- L'esercizio richiede di contare i cammini dell'albero

  ↓

  problema di contare tali cammini all'interno di una funzione ricorsiva

- Passaggio di 3 parametri:
  - albero
  - cammino interno
  - cammino esterno
- Problema del calcolo della lunghezza dei cammini
- Diversi algoritmi di calcolo della lunghezza

Prima soluzione

Modificare la struct aggiungendo una variabile

```
int livello;
```

Vantaggi

- Semplicità dell'algoritmo richiesto dall'esercizio.

Svantaggi

- Si sposta il problema in fase di costruzione dell'albero.
- Problema del calcolo del livello nel caso di modifiche alla struttura dell'albero.

Seconda soluzione

Utilizzo di una variabile contatore che si incrementa passando da una funzione alla sua istanza ricorsiva

- **Intestazione della funzione**

```
void Cammini (Tree*&radice  int&Camminoest, int&Camminoint, int curr)
```

- **Intestazione della funzione**

```
void Cammini (Tree*&radice, int&Camminoest, int&Camminoint, int curr)
```

- **Intestazione della funzione**

```
void Cammini (Tree*&radice, int&Camminoest, int&Camminoint, int curr)
```

- **Intestazione della funzione**

```
void Cammini (Tree*&radice, int&Camminoest, int&Camminoint, int curr)
```

- **Algoritmo di risoluzione della funzione Cammini**

- **Intestazione della funzione**

```
void Cammini (Tree*&radice, int&Camminoest, int&Camminoint, int curr)
```

- **Algoritmo di risoluzione della funzione Cammini**

Valuto il nodo corrente

è un nodo *foglia*?

è un nodo *non foglia*?

- **Intestazione della funzione**

```
void Cammini (Tree*&radice, int&Camminoest, int&Camminoint, int curr)
```

- **Algoritmo di risoluzione della funzione Cammini**

  Valuto il nodo corrente

  è un nodo *foglia*?

  SI

```
CamminoEsterno=CamminoEsterno+livello nodo
```

  è un nodo *non foglia*?

- **Intestazione della funzione**

```
void Cammini (Tree*&radice, int&Camminoest, int&Camminoint, int curr)
```

- **Algoritmo di risoluzione della funzione Cammini**

Valuto il nodo corrente

è un nodo *foglia*?

SI

```
CamminoEsterno=CamminoEsterno+livello nodo
```

è un nodo *non foglia*?

SI

```
CamminoInterno=CamminoInterno+livello nodo
```

- **Intestazione della funzione**

```
void Cammini (Tree*&radice, int&Camminoest, int&Camminoint, int curr)
```

- **Algoritmo di risoluzione della funzione Cammini**

Valuto il nodo corrente

è un nodo *foglia*?

SI

```
CamminoEsterno=CamminoEsterno+livello nodo
```

è un nodo *non foglia*?

SI

```
CamminoInterno=CamminoInterno+livello nodo
```

```
Cammini(sottoalbero di destra)
Cammini(sottoalbero di sinistra)
```

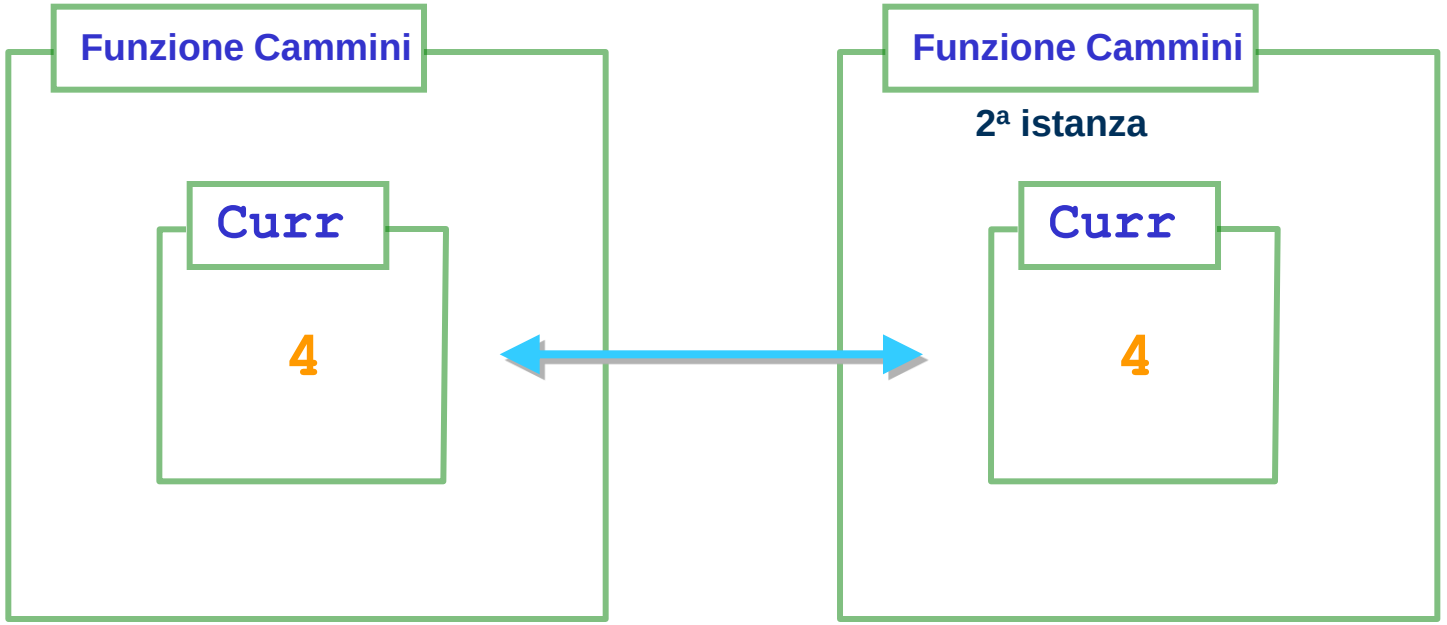
Per riconoscere se un nodo è foglia basterà scrivere la seguente condizione:

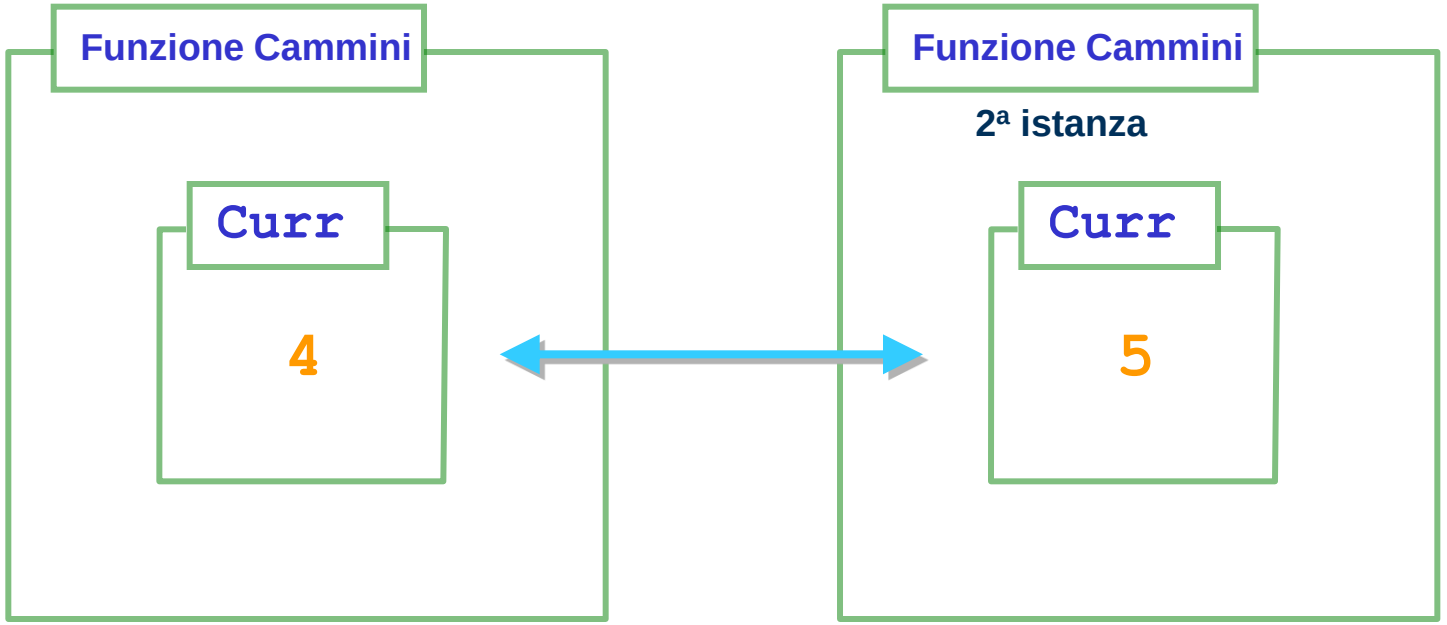
```
if ((nodo->leftptr==0) && (nodo->rightptr==0))
{
  //nodo è foglia
}
```

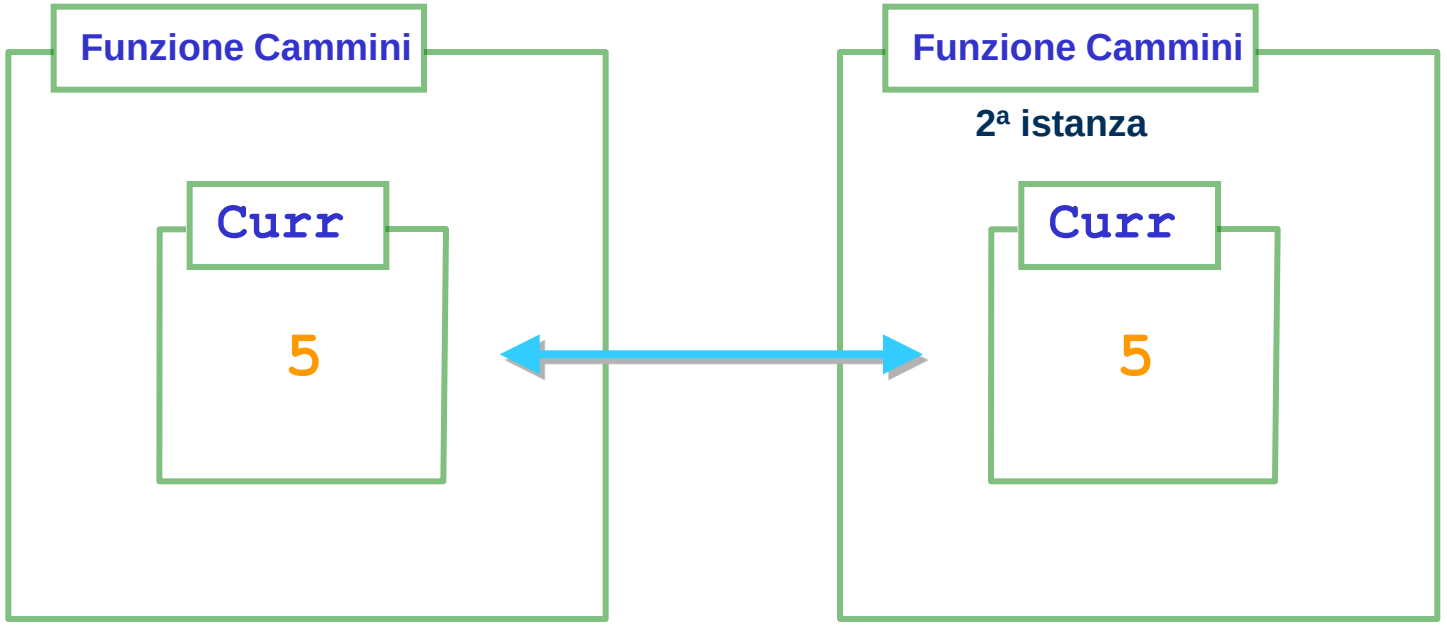
```
if ((radice->leftptr)!=0) || ((radice->righptr)!=0))
```

```
Cammini(radice->leftptr,Camminoest,Camminoint,curr);
Cammini(radice->rightptr,Camminoest,Camminoint,curr);
```

Le variabili definite nelle funzioni ricorsive hanno uno spazio di memoria autonomo e dunque è necessario passare esplicitamente tale valore.


Funzione Cammini
Curr
4
Funzione Cammini
2ª istanza
Curr
4


Funzione Cammini
Curr
4
Funzione Cammini
2ª istanza
Curr
5


Funzione Cammini
Curr
5
Funzione Cammini
2ª istanza
Curr
5