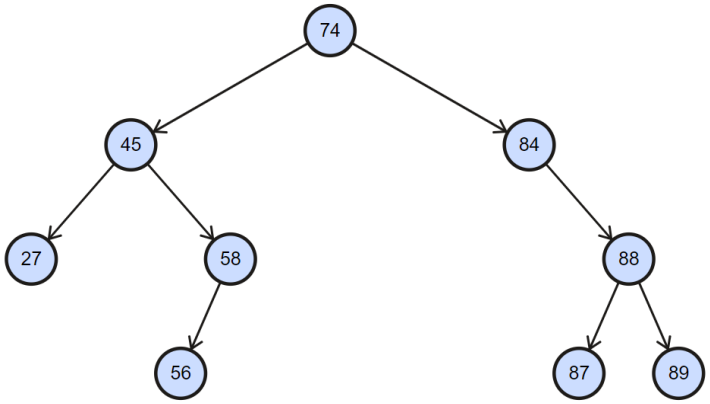
Esercizi Binary Tree

Esercizio 1: Scrivere un programma per creare due alberi binari identici, che corrispondono entrambi all'albero binario mostrato in figura, inserendo i nodi nei due alberi con una sequenza di inserimento dei nodi diversa tra i due alberi. Verificare la corretta creazione dell'albero eseguendo un attraversamento level-order.



Esercizio 2: Scrivere una funzione membro della classe `BinaryTree` che ritorni il minimo tra tutti i nodi dell'albero.

Esercizio 3: Scrivere una funzione membro della classe `BinaryTree` per verificare se TUTTI i nodi di un albero binario o hanno due figli oppure non hanno nessun figlio.

Esercizio 4: Scrivere una funzione membro ricorsiva per calcolare la "lunghezza dei percorsi interni" di un albero binario (IPL). IPL è definita come la somma delle profondità di tutti i nodi che hanno due figli. La profondità di un nodo è definita come la lunghezza (ovvero il numero di archi) del cammino dal nodo radice al nodo stesso.

Esercizio 5: Scrivere una funzione membro della classe BinaryTree per inserire elementi in un albero binario in modo che ogni nuovo elemento venga inserito da sinistra verso destra nell'albero un livello per volta (level-order).

Esercizio 6: Scrivere una funzione membro della classe BinaryTree per calcolare la larghezza di un albero binario. La larghezza è il numero massimo di nodi che stanno tutti al medesimo livello. Suggerimento: utilizzare una versione modificata dell'attraversamento level_order con due iterazioni annidate. L'iterazione più esterna si esegue per ogni livello dell'albero. In questa iterazione si valuta se la larghezza del livello corrente (dimensione della coda) è maggiore della larghezza massima attuale. Nell'iterazione

più interna si svuota la coda del numero di elementi presenti nel livello corrente e si inseriscono in coda gli elementi figli. In questo modo ad ogni iterazione più esterna la coda contiene sempre un numero di elementi pari al numero di nodi del livello attuale dell'albero.

Esercizio 7: Scrivere una funzione membro della classe BinaryTree per calcolare e stampare a video "l'altezza minimale" di ciascun nodo di un albero binario. L'altezza minimale di un nodo v è la minima distanza di v da una delle foglie del suo sottoalbero.