Esercizio 1
Scrivi un programma Python per creare un albero di ricerca binario bilanciato, partendo da un array i cui elementi sono ordinati (in ordine crescente).

Esercizio 2
Scrivi un programma Python per verificare se un albero binario è anche un albero di ricerca binario.

Esercizio 3
Scrivi un programma Python per trovare il k-esimo minimo elemento in un albero di ricerca binario.

Esercizio 4

Un **albero ricoprente** (o **spanning tree**) di un grafo, connesso e con archi non orientati, è un albero che contiene

- tutti i vertici del grafo e
- soltanto un sottoinsieme degli archi,

cioè solo quelli necessari per connettere tra loro tutti i vertici con uno e un solo cammino. Se un vertice manca, allora non è un albero ricoprente.

L'albero ricoprente **minimo** (o **minimum spanning tree**) è un albero ricoprente nel quale sommando i pesi degli archi si ottiene un valore minimo.

Algoritmo di Prim
Steps:

1. Inizializzare il minimum spanning tree con un vertice scelto randomicamente.
2. Identificare gli archi che connettono il tree a nuovi vertici, trovare quello con peso minimo e aggiungerlo al tree.
3. Ripetere il punto 2 fino ad ottenere un minimum spanning tree.

Pseudocodice

```
T = ∅;  # contiene la lista di archi del minimum spanning tree
U = { 1 };  # contiene la lista di vertici già visitati
while (U ≠ V)  # quando U==V abbiamo visitato tutti i vertici
    sia (u, v) l'arco con costo minore, dato u ∈ U and v ∈ V - U;
    T = T ∪ {(u, v)}  # aggiungiamo l'arco con costo minore a T
    U = U ∪ {v}  # aggiungiamo u alla lista di vertici visitati
```