Esercizio 1
Dimostrare che O(n log n) è il limite inferiore alla complessità in tempo al caso pessimo per gli algoritmi di ordinamento di n elementi basati su confronti.

Soluzione: vedi libro di testo.

Esercizio 2
Dato un array ordinato di n interi, progettare un algoritmo efficiente che costruisca un albero binario di ricerca di altezza O(log n) che contenga gli interi dell'array come chiavi, e analizzarne la complessità.

Soluzione:

```
CreaAlbero(A, i, j)
if (i > j) return null;
m = (i+j) / 2;
u = NuovoNodo();
u.dato = A[m];
u.sx = CreaAlbero(A, i, m-1);
u.dx = CreaAlbero(A, m+1, j);
return u;
```

Complessità in tempo:
$T(n) = 2\,T(n/2) + O(1) = O(n)$ (Teorema Principale, primo caso).

Esercizio 3
Dato un grafo orientato, memorizzato con le seguenti liste di adiacenza:

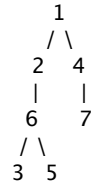
1 --> 2, 4, 7
2 --> 6
3 --> 1
4 --> 7
5 --> 2, 6
6 --> 3, 5
7 --> 5, 6

eseguire una visita DFS a partire dal vertice 1, indicando:

1. la sequenza dei vertici incontrati
2. l'albero DFS
3. la classificazione degli archi indotta dalla visita.

Soluzione:

1. Sequenza dei vertici: 1, 2, 6, 3, 5, 4, 7
2. Albero DFS:



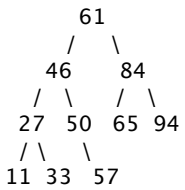
3. Classificazione degli archi:
Archi in avanti: (1,7)
Archi all'indietro: (3,1), (5,2), (5,6)
Archi di attraversamento (o trasversali): (7,5), (7,6)
Archi dell'albero: tutti gli altri

Esercizio 4
Inserire la sequenza di chiavi:

50, 27, 46, 65, 61, 94, 11, 33, 57, 84

in un albero AVL inizialmente vuoto, indicando i nodi critici e le rotazioni eseguite per ribilanciarel'albero.

L'albero finale è il seguente:



Le rotazioni eseguite sono, nell'ordine: SD(50), DS(50), DD(46), DS(65)