

Problema 1

Si considerino i seguenti metodi:

```
static void mistero1(int[] v) {
    mistero1(v, 0);
}

static void mistero1(int[] v, int i) {
    if(i == v.length) return;
    int a = v[i];
    i++;
    mistero1(v, i);
    mistero2(a, v, i);
}

static void mistero2(int a, int[] v, int j) {
    if((j == v.length) || (a <= v[j])) v[j-1] = a;
    else {
        v[j-1] = v[j];
        mistero2(a, v, j+1);
    }
}
```

- (a) Calcolare, motivandolo adeguatamente, il costo computazionale del metodo `mistero1(int[])` in funzione della dimensione dell'input.
- (b) Valutare, motivandolo adeguatamente, il costo computazionale di *caso migliore* del metodo `mistero1(int[])` in funzione della dimensione dell'input.
- (c) Come viene trasformato l'array `{3, -2, 0, -2, 4, -1}` dal metodo `mistero1(int[])`?

Problema 2

Definire una opportuna struttura ad albero per la rappresentazione semplificata di un file system, che può contenere cartelle, file e link (simbolici, cioè espressi come percorsi assoluti). Per ciascuna cartella si desidera rappresentare il nome e il numero di elementi contenuti; per ciascun file, il nome e la dimensione in byte; per ciascun link, il nome e il percorso assoluto di un elemento del file system a cui il link fa riferimento (non deve esserne garantita l'esistenza). All'interno di una stessa cartella non possono coesistere due elementi con lo stesso nome.

- (a) Definire la classe Java `FileSystem` e le eventuali altre classi a supporto per la rappresentazione semplificata di file system e la soluzione dei punti successivi (per ciascuna classe: campi, costruttori e firme dei metodi).
- (b) Definire e dettagliare il metodo `search(String name)` della classe `FileSystem` che, data una stringa `name`, restituisce una opportuna lista di cui elementi contengono i percorsi assoluti (stringhe) di file o cartelle (sono esclusi i link) di nome `name`. Qualora non esistano occorrenze di `name`, restituire `null`.
- (c) Definire e dettagliare il metodo `long getSize(.)` della classe `FileSystem` che, dato un riferimento a un elemento del file system (file o cartella), restituisce la dimensione dell'elemento. In particolare, nel caso di cartella, si dovrà restituire la somma delle dimensioni in byte dei file contenuti nella porzione di file system la cui radice è la cartella stessa (senza seguire i link simbolici).

Facoltativo: scrivere una variante del metodo `getSize` che, nel calcolo della somma delle dimensioni, segua anche i link simbolici.

Problema 3

- (a) Definire il concetto di minimum spanning tree (MST) di un grafo pesato sugli archi e disegnare un esempio di grafo che ammette due MST differenti, evidenziandoli.
- (b) Descrivere l'algoritmo di Kruskal per il calcolo degli MST e mostrare la sua esecuzione sul grafo di esempio di cui al punto (a).
- (c) Descrivere una struttura dati, a supporto dell'algoritmo di Kruskal, per la gestione di insiemi disgiunti.