

Laboratorio 8

Esercizio 1

In una classe **MetodiSuArray** scrivere i seguenti metodi iterativi:

- `int[] initArrayInt()` che legge da input un numero intero $N \geq 0$ come dimensione di un array, e successivamente legge altri N valori interi. Ritorna l'array di dimensione N contenente gli elementi letti. Usare `Scanner.nextInt()` di `Scanner` per leggere l'input.
- `void stampaArrayInt(int[] a)` che stampa a video gli elementi di `a`.

Scrivere una classe **TestInitArray** che inizializza un array con il metodo `initArrayInt()` e ne stampa il contenuto. Provare ad eseguire il programma Java in modo interattivo.

Successivamente, creare un file `dati.txt` contenente:

```
3
10
15
12
```

ed eseguire il programma dal terminale comandi come:

```
java TestInitArray < dati.txt
```

NOTA: il file `dati.txt` **DEVE** finire con un ritorno a capo dopo il 12, altrimenti non viene completato l'input.

Esercizio 2

Aggiungere alla classe **MetodiSuArray** i seguenti metodi iterativi:

- `void copiaElementi(int[] from, int[] to)` che copia tutti gli elementi dall'array `from` all'array `to`. Il metodo assume che `from` e `to` siano inizializzati ed abbiano la stessa dimensione.
- `int[] clonaArray(int[] a)` che ritorna una copia di `a` opportunamente allocata nello heap. Utilizzare `copiaElementi` per effettuare le operazioni di copia. Se `a` è null, allora ritorna null.

Modificare la classe **TestInitArray** per clonare l'array inserito. Verificare, tramite opportune operazioni di stampa, che l'array clonato contenga gli stessi elementi dell'array iniziale.

Esercizio 3

Aggiungere alla classe **MetodiSuArray** i seguenti metodi iterativi:

- `int[] filtroMinoriDi(int[] a, int limiteSuperiore)` che restituisce un nuovo array di interi contenente tutti gli elementi di `a` che sono strettamente minori del valore `limiteSuperiore`;
- `int[] filtroIntervalloDisp(int[] a, int min, int max)` che restituisce l'array degli interi copiati da `a` che sono dispari e compresi tra `min` e `max` (estremi inclusi);
- `boolean[] trasduttore(int[] a, int limiteSuperiore)` che restituisce un array di booleani, in cui ogni elemento sia true se l'elemento di posizione corrispondente in `a` è inferiore o uguale a `limiteSuperiore` e false altrimenti;
- `void stampaArrayBoolean(boolean[] a)` che stampa a video gli elementi di `a` (perché non si può usare `stampaArrayInt?`).

Verificare il corretto funzionamento di questi metodi creando una classe **TestFiltriArray**, e sperimentando vari input.

Esercizio 4

Aggiungere alla classe **MetodiSuArray** i seguenti metodi in forma **iterativa** e **ricorsiva**:

- un metodo *eqArray* che prende come parametri due array di interi e restituisce true se questi contengono gli stessi valori, false altrimenti.
- un metodo *tuttiPariMaggioriDi* che prende in input un array a e un intero num, e ritorna true se tutti gli elementi di a sono pari e maggiori di num, false altrimenti

NOTA: cosa devono ritornare se a è vuoto? E se a è null?

Scrivere una classe **TestEqArray** con un main che dichiara i seguenti array:

```
a1 = {0, 2, 3, 5, -4, 9, 10}
a2 = {6, 4, 4, 8, 12, 4, 22}
a3 = {10, 7, 5, 99, 31, 20}
a4 = {5, 7, 8, 9, 12}
a5 = MetodiSuArray.clonaArray(a2)
a6 = null
```

e stampa a video i risultato di queste operazioni:

- il risultato di *eqArray(a1, a2)*, *eqArray(a4, a2)*, *eqArray(a5, a2)*, *eqArray(a3, a3)*, *eqArray(a2, a6)*, *eqArray(a6, a6)*
- il risultato di *tuttiPariMaggioriDi(a1, 0)*, *tuttiPariMaggioriDi(a2, 2)*, *tuttiPariMaggioriDi(a2, 8)*, *tuttiPariMaggioriDi(a4, -5)*, *tuttiPariMaggioriDi(a6, 3)*

Esercizio 5

Sviluppare nella classe **MetodiSuArray** un metodo *indiceSubSeq*(int[] a, int[] b) che restituisce l'indice al primo elemento in a della prima occorrenza della sottosequenza b in a.

Ad esempio, se a = {1, **2, 3**, 1, 2, 3} e b = {**2, 3**}, allora *indiceSubSeq*(a, b) restituisce 1.

Se la sottosequenza b non compare in a, il metodo deve restituire -1.

Scrivere una classe di test **TestSubSeq** che chiede all'utente due sequenze in input, e stampa a video il risultato della ricerca.

Provare con il seguente input (scritto in modo interattivo oppure come file):

```
8
6 8 2 5 0 9 1 3
4
5 0 9 1
```

e verificare che la sotto sequenza viene trovata all'indice 3.