

Programmazione e Algoritmi 1

A.A. 2023/24 — Compito del 12 febbraio 2024 – Versione Java

prof. Gianluca Amato

Gli esercizi di programmazione saranno valutati sulla base della correttezza, efficienza e comprensibilità della soluzione proposta. In generale, **non è possibile usare classi che non fanno parte del programma del corso**.

Esercizio 1 (5 punti)

Scrivere un metodo statico `removeDuplicate` che prende come parametro in input un array di interi e restituisce un nuovo array in cui tutti i duplicati sono stati rimossi. Ad esempio, `removeDuplicate(new int[] { 1, 10, 9, 2, 10, 7, 2 })` restituisce l'array `{ 1, 10, 9, 2, 7 }`.

Esercizio 2 (5 punti)

Scrivere alcuni test nel framework `JUnit` per verificare il corretto funzionamento di `removeDuplicate`. In particolare, si deve controllare che il valore di ritorno sia corretto per:

- l'esempio dell'Esercizio 1;
- l'array vuoto;
- un array in cui tutti gli elementi sono uguali: sia l'elemento da inserire nell'array sia la sua lunghezza devono essere generati casualmente.

Esercizio 3 (5 punti)

Scrivere un metodo statico `positivePairs` che prende due parametri: un array di interi `a` e un numero intero positivo `k`. La funzione restituisce un nuovo array con le prime `k` coppie di numeri interi positivi presenti in `a`. Ad esempio `positivePairs(new int[] {-5, 6, -11, -3, -9, 15, -1, 90, -21, 3}, 4)` restituisce l'array `{ {6, 15}, {6, 90}, {6, 3}, {15, 90} }`. Se il numero di coppie positive presenti in `a` è minore di `k`, la funzione restituisce tutte le coppie positive presenti in `a`. Si noti che:

- si considerano solo le coppie (x, y) dove x viene prima di y nell'array `a`;
- ogni elemento dell'array che `positivePairs` produce come risultato è a sua volta un array di due elementi, usato per rappresentare una coppia di valori.

Esercizio 4 (5 punti)

Scrivere un programma che legge dalla riga di comando una sequenza di numeri interi, che rappresentano in parametri con i quali chiamare il metodo statico `positivePairs` dell'esercizio precedente. Il primo numero sarà il valore di `k` e i restanti numeri sono gli elementi dell'array `a`. Il programma, dopo aver chiamato il metodo `positivePairs`, deve stampare sullo schermo il risultato ottenuto, ogni coppia su una riga divisa, e gli elementi della coppia separati da spazi. Ad esempio, se il programma viene implementato nella classe `PositivePairs` e viene invocato con:

```
java PositivePairs.java 4 -5 6 -11 -3 -9 15 -1 90 -21 3
```

l'output generato sarà

```
6 15
6 90
6 3
15 90
```

Esercizio 5 (5 punti)

Illustrare il funzionamento degli algoritmi di ricerca lineare e ricerca binaria, prima in termini generali e poi applicandoli alla ricerca del numero 10 nella lista che contiene, in ordine, tutti i numeri da 1 a 100. Discutere vantaggi e svantaggi dei due algoritmi.

Esercizio 6 (8 punti)

Si consideri il seguente codice Java (che, per la cronaca, implementa la *funzione di Ackermann*):

```
1  public class Ackermann {
2      public static int ackermann(int m, int n) {
3          if (m == 0) {
4              return n + 1;
5          } else if (n == 0) {
6              return ackermann(m - 1, 1);
7          } else {
8              int x = ackermann(m, n - 1);
9              return ackermann(m - 1, x);
10         }
11     }
12
13     public static void main(String[] args) {
14         System.out.println(ackermann(1, 2));
15     }
16 }
```

Eseguire passo passo il programma (senza l'ausilio di un computer) e fornirne la traccia di esecuzione.