```
/**
  raccolta di metodi statici per Operazioni su insiemi
*/

public class OpInsiemi
{
    public static int[] readConsole()
    { int n = Console.readInt("Di che lunghezza vuoi l'array?");
      int[] elementi = new int[n];

      for(int i=0; i<n; i++) {
        elementi[i] = Console.readInt("immetti elemento n. " + i);
      }
      return elementi;
    }

  /**
    scrive sulla consolle gli elementi dell'array
    in sequenza
  */
    public static void println(int[] a) {
      System.out.println("l'array e': ");
      for(int i=0; i < a.length; i++)
      { System.out.print(a[i]);
      }
      System.out.println("");
  }

  public static boolean èOrdinato(int[] a) {
    int n = a.length;
    int i = 1;
    while(i < n && a[i-1] <= a[i]) i++;
    return i==n;
  }


/** presi come argomenti due array ordinati privi di elementi ripetuti,
    crea e restituisce un nuovo array ordinato (privo di elementi ripetuti)
    che rappresenta l'INTERSEZIONE insiemistica dei due argomenti,
    senza modificare tali array-argomenti
 */
  public static int[] intersezione(int[] a, int[] b) {
    int m = a.length;
    int n = b.length;
    int p;
    if(m < n) p = m; else p = n;
    int[] c = new int[p];

    int i = 0, j = 0, k = 0;
    while(i < m && j < n) {
      if(a[i] < b[j]) i++;
      else if(a[i] > b[j]) j++;
      else {
        c[k] = a[i];
        i++; j++; k++;
      }
    }

    int[] d = new int[k];
    for(i = 0; i < k; i++) d[i] = c[i];
```

```
    return d;
  }

/** presi come argomenti due array ordinati privi di elementi ripetuti,
    crea e restituisce un nuovo array ordinato (privo di elementi ripetuti)
    che rappresenta l'unione insiemistica dei due argomenti,
    senza modificare tali array-argomenti
 */
  public static int[] unione(int[] a, int[] b) {
    int m = a.length;
    int n = b.length;
    int p;
    int[] c = new int[m+n];

    int i = 0, j = 0, k = 0;
    while(i < m && j < n) {
      if(a[i] < b[j]) {
        c[k] = a[i];
        i++; k++;
      }
      else if(a[i] > b[j]) {
        c[k] = b[j];
        j++; k++;
      }
      else {
        c[k] = a[i];
        i++; j++; k++;
      }
    }
    while(i < m) {
      c[k] = a[i];
      i++; k++;
    }
    while(j < n) {
      c[k] = b[j];
      j++; k++;
    }

    int[] d = new int[k];
    for(i = 0; i < k; i++) d[i] = c[i];
    return d;
  }

  static final int N = 10000; // lunghezza dell'array da ordinare
  static final int K = 3*N; // N/3; // numero di valori diversi possibili

  public static void main(String args[]) {
    int[] ar1 = {3,5,21, 30,30, 65, 88, 90, 99};
    int[] ar2 = {-5, -2, 4, 11, 13, 21, 25, 30, 89, 90, 94, 97};
    scriviSuOutputBox(fondi(ar1, ar2));

    int[] ar3 = {3,5,21, 30, 65, 88, 90, 99};
    int[] ar4 = {-5, -2, 4, 11, 13, 21, 25, 30, 89, 90, 94, 97};
    scriviSuOutputBox(fondi(ar3, ar4));
    scriviSuOutputBox(unione(ar3, ar4));
    scriviSuOutputBox(intersezione(ar3, ar4));
    scriviSuOutputBox(IntArrayUtil.intersezione(ar3, ar4));

    int[] fuso = fondi(ar3,ar4);
    System.out.println("ricerca");
```

```
    System.out.println(fuso[ricercaIndiceWh(89, fuso)]);
    System.out.println(fuso[ricercaIndiceWhOtt(89, fuso)]);
    System.out.println(fuso[binRicercaIndiceDi(89, fuso)]);

    int[] arr1 = leggi();
    scriviSuOutputBox(arr1);

    Random generatore = new Random();

    int[] bigArray1 = new int[N];
    for(int i = 0; i < N; i++) {
      inserisci(generatore.nextInt(K), bigArray1, i);
    }
    System.out.println(èOrdinato(bigArray1));

    int[] bigArray2 = new int[N];
    for(int i = 0; i < N; i++) {
      inserisci(generatore.nextInt(K), bigArray2, i);
    }
    System.out.println(èOrdinato(bigArray2));

    int[] veryBigArray = fondi(bigArray1, bigArray2);
    System.out.println(èOrdinato(veryBigArray));

    int num = generatore.nextInt(K);
    int j = ricercaIndiceWh(num, veryBigArray);
    System.out.println(num + " "+ (j != -1 ? veryBigArray[j]+" " :
    "non trovato"));

    j = ricercaIndiceWhOtt(num, veryBigArray);
    System.out.println(num + " "+ (j != -1 ? veryBigArray[j]+" " :
    "non trovato"));

    j = ricercaIndiceDi(num, veryBigArray);
    System.out.println(num + " "+ (j != -1 ? veryBigArray[j]+" " :
    "non trovato"));

  }

}
```