Esame di Programmazione II, 3 settembre 2012

Si considerino le seguenti interfacce:

```
public interface Set extends Iterable<Object> {
  boolean contains(Object element);
  boolean intersects(Set other);
  int size();
}

public interface ModifiableSet extends Set {
  boolean add(Object element);
  boolean remove(Object element);
  boolean addAll(Set set);
  boolean removeAll(Set set);
}
```

La prima specifica l'interfaccia di un insieme senza metodi di modifica, mentre la seconda quella di un insieme con metodi di modifica.

Esercizio 1 [11 punti]

Si scriva una classe `ArraySet` che implementa `Set`, utilizza un array per contenere i suoi elementi e implementa i seguenti costruttori e metodi:

```
public ArraySet(Object... elements)
protected Object[] getElements()
protected void setElements(Object[] elements)
public boolean contains(Object element)
public boolean intersects(Set other)
public int size()
public Iterator<Object> iterator()
public boolean equals(Object other)
public int hashCode()
```

Si noti che il costruttore costruisce un insieme non modificabile che contiene gli elementi forniti. I metodi `getElements` e `setElements` forniscono accesso in lettura e scrittura agli elementi contenuti nell'insieme. Il metodo `contains` determina se un elemento è contenuto nell'insieme. Il metodo `intersects` determina se `this` interseca `other`. Il metodo `size` restituisce il numero di elementi contenuti nell'insieme. Il metodo `iterator` restituisce un iteratore sugli elementi dell'insieme. Il metodo `equals` determina se `this` e `other` sono due `Set` che contengono gli stessi elementi, in qualsiasi ordine. Il metodo `hashCode` deve essere non banale e consistente con `equals`.

L'uguaglianza fra gli elementi degli insiemi deve essere determinata dal metodo `equals` di tali elementi, non da `==`.

Esercizio 2 [11 punti] Si scriva una classe `ModifiableArraySet` che implementa `ModifiableSet` ed estende `ArraySet`. Oltre ai metodi di `ArraySet`, deve implementare quindi anche i metodi e costruttori:

```
public ModifiableArraySet()
public ModifiableArraySet(Object... elements)
public ModifiableArraySet(Set father)
public boolean add(Object element)
public boolean remove(Object element)
public boolean addAll(Set set)
public boolean removeAll(Set set)
```

Si noti che il costruttore senza argomenti costruisce un insieme inizialmente vuoto. Il costruttore che riceve come argomento un altro insieme `father` costruisce un insieme modificabile che contiene inizialmente gli stessi elementi di `father`. Le funzioni che aggiungono o rimuovono uno o più elementi restituiscono `true` se e solo se viene effettivamente aggiunto o rimosso un elemento. Per esempio, se si aggiunge un elemento a un insieme che già lo contiene, `add` deve restituire `false`; se si rimuove un elemento da un insieme che non lo contiene, `remove` deve restituire `false`.

Se tutto è corretto, l'esecuzione del seguente programma:

```
public class Main {

  public static void main(String[] args) {
    Set s1 = new ArraySet("ciao", "amico", "come", "va?");
    Set s2 = new ModifiableArraySet("oggi", "va?", 12, 113);

    System.out.println("1: " + s1.equals(s2));
    System.out.println("2: " + s1.intersects(s2));

    ModifiableSet s3 = new ModifiableArraySet("amico", "va?", "ciao", "va?");
    s3.add("come");
    s3.add(new String("ciao"));

    System.out.println("3: " + s1.equals(s3));
    System.out.println("4: " + s1.intersects(s3));
  }
}
```

dovrà stampare:

```
1: false
2: true
3: true
4: true
```