

Semplificazione dell'esame di Programmazione II del 2 settembre 2013 (2 ore)

Il gioco del 15 consiste in una matrice quadrata di 16 tessere di cui una sola è vuota e le altre 15 sono riempite con i numeri tra 1 e 15. Il gioco è risolto se le tessere sono ordinate in senso crescente secondo una lettura per righe, cioè se la matrice è nella configurazione:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Si intende adesso generalizzare questo gioco a una matrice di dimensioni generiche $width \times height$. Le tessere non sono più solo numeriche, ma possono più generalmente essere degli oggetti con una relazione di ordinamento fra di loro. Ad esempio, un gioco 4×3 con tessere alfabetiche di lunghezza fra 1 e 5 è il seguente:

swhv	kt	g	wohp
sqvtc		nzwuo	evs
hkf	qb	lt	me

Si noti che tale gioco non è risolto poiché le stringhe non sono in ordine alfabetico secondo una lettura per righe, né la casella vuota è in basso a destra.

Una tessera deve estendere la classe:

```
public abstract class Tessera implements Comparable<Tessera> {
    public abstract boolean equals(Object other);
    public abstract String toString();
}
```

e quindi avrà anche il metodo `abstract int compareTo(Tessera other)` ereditato da `java.lang.Comparable<Tessera>`.

Esercizio 1 [3 punti] Si completi la sottoclasse di `Tessera` che implementa una tessera numerica etichettata con `num`:

```
public final class TesseraNumerica extends Tessera {
    private final int num;
    TesseraNumerica(int num) { this.num = num; }
    ...
}
```

Una tessera numerica è `equals` solo a una tessera numerica che contiene lo stesso numero. L'ordinamento fra tessere numeriche avviene secondo l'ordine crescente del numero contenuto. L'ordinamento fra una tessera numerica e un altro tipo di tessera avviene secondo l'ordine alfabetico crescente del nome delle loro classi.

Esercizio 2 [3 punti] Si completi la sottoclasse di `Tessera` che implementa una tessera alfabetica etichettata con `s`:

```
public final class TesseraAlfabetica extends Tessera {
    private final String s;
    TesseraAlfabetica(String s) { this.s = s; }
    ...
}
```

Una tessera alfabetica è `equals` solo a una tessera alfabetica che contiene la stessa stringa (rispetto all'`equals` fra stringhe). L'ordinamento fra tessere alfabetiche avviene secondo l'ordine alfabetico della stringa contenuta. L'ordinamento fra una tessera alfabetica e un altro tipo di tessera avviene secondo l'ordine alfabetico crescente del nome delle loro classi.

Esercizio 3 [2 punti] Una fattoria di tessere è un oggetto con un metodo che restituisce una tessera a caso (non necessariamente diversa) ogni volta che viene chiamato:

```
public interface FattoriaDiTessere {
    public Tessera mkRandom();
}
```

Si completi una sua implementazione che genera tessere numeriche a caso, numerate con un numero a caso fra 1 e `max` inclusi:

```

public class FattoriaDiTessereNumeriche implements FattoriaDiTessere {
    public FattoriaDiTessereNumeriche(int max) { ... }
    ...
}

```

Esercizio 4 [4 punti] Si completi un'implementazione di una fattoria che ritorna tessere alfabetiche casuali, etichettate con stringhe alfabetiche a caso di lunghezza a caso fra 1 e 5 inclusi:

```

public class FattoriaDiTessereAlfabetiche implements FattoriaDiTessere {
    public FattoriaDiTessereAlfabetiche() {}
    ...
}

```

Esercizio 5 [10 punti] Un gioco è una matrice di tessere distinte (non equals), di cui una sola è vuota. Un gioco è risolto quando la tessera vuota è in basso a destra e le tessere non vuote sono in ordine crescente secondo una lettura per righe. Si completi l'implementazione di un gioco:

```

public class Gioco {
    ...
    public Gioco(FattoriaDiTessere fattoria, int width, int height) {
        // costruisce un gioco a caso della dimensione indicata, posizionando la casella vuota a caso
        // e creando le altre caselle a caso usando la fattoria indicata
        ...
    }

    @Override
    public String toString() { // restituisce una stringa come da esempi in basso
        ...
    }

    public boolean risolto() { // determina se questo gioco e' risolto
        ...
    }
}

```

Se tutto è corretto, il seguente programma:

```

public class Main {
    public static void main(String[] args) {
        Gioco gioco = new Gioco(new FattoriaDiTessereAlfabetiche(), 4, 4);
        System.out.println(gioco);

        FattoriaDiTessere f = new FattoriaDiTessereNumeriche(8);
        do {
            gioco = new Gioco(f, 3, 2);
            System.out.println(gioco);
        }
        while (!gioco.risolto());
    }
}

```

stamperà qualcosa del tipo:

```

swhv   kt    g   wohp
sqvtc      nzwuo  evs
  hkf   qb    lt   me
aotgf ehnlf dyaxc  dnc

```

```

3      8
6    1  2

```

```

5      1
2    7  8

```

.....
[molti tentativi non risolti]
.....

2	3	4
6	7	

terminando quindi con un gioco risolto.