Esame di Programmazione II, 24 febbraio 2012

Esercizio 1 [2 punti] Un `Draw` è un rettangolo di caratteri. Per esempio, il seguente è un `Draw` di altezza (*height*) 4 e larghezza (*width*) 6:

```
DrAw
123ABC
  $@@@
abcdef
```

Si noti che le righe di un `Draw` sono tutte lunghe tanto quanto la larghezza del `Draw` e non hanno nessun new line alla fine. Per esempio, la prima riga del `Draw` visto sopra è la stringa `DrAw__` (con due spazi alla fine). La terza riga è `__$@@@` (con due spazi all'inizio).

Si completi la definizione di un `Draw` data dalle seguente classe astratta:

```
public abstract class Draw {
  public abstract int getWidth();
  public abstract int getHeight();
  protected abstract String getRow(int num) throws IllegalArgumentException;

  @Override
  public final String toString() {
    // questo dovete scriverlo voi
  }
}
```

Lo scopo del metodo `getRow` (implementato nelle sottoclassi!) sarà di restituire la riga numero `num` del `Draw` (dove `num` è 0 per la riga più in alto ed è `getHeight() - 1` per quella più in basso). Tale metodo genera un'eccezione non controllata di classe `java.lang.IllegalArgumentException` (classe già esistente in Java) se `num` non sta dentro tali limiti. Tutto quello che dovete fare in questo esercizio è di implementare il metodo `toString()` che restituisce la concatenazione delle righe del `Draw`, andando a capo alla fine di ciascuna riga.

Esercizio 2 [3 punti] Si definisca una sottoclasse astratta `Letter` di `Draw` i cui oggetti hanno larghezza 7 e altezza 8. Queste dimensioni non devono essere modificabili dalle sottoclassi di `Letter`.

Esercizio 3 [5 punti] Si definiscano delle sottoclassi finali di `Letter`, chiamate `H`, `E`, `L` e `O`, che implementano i seguenti `Draw` (si ricordi che `_` è solo un modo per evidenziare gli spazi):

```
_______      _______      _______      _______
_*___*_      _*****_      _*_____      _*****_
_*___*_      _*_____      _*_____      _*___*_
_*___*_      _*_____      _*_____      _*___*_
_*****_      _*****_      _*_____      _*___*_
_*___*_      _*_____      _*_____      _*___*_
_*___*_      _*_____      _*_____      _*___*_
_*___*_      _*****_      _*****_      _*****_
```

Si definisca quindi una sottoclasse finale `Star` di `Draw` che implementa il seguente `Draw`:

```
X___X
_X_X_
__X__
_X_X_
X___X
```

Esercizio 4 [6 punti] Un `HorizontalDraw` è un `Draw` formato concatenando orizzontalmente uno o più `Draw` (i suoi *figli*). I figli posso avere altezze e larghezze diverse e vengono tutti allineati in alto. Per esempio, l'`HorizontalDraw` con due figli, uno `Star` e una `H`, è il seguente:

```
X___X_______
_X_X__*___*_
__X___*___*_
_X_X__*___*_
X___X_*****_
______*___*_
______*___*_
______*___*_
```

Si noti che `Star` ed `H` hanno altezze diverse e le righe mancanti allo `Star` diventano spazi aggiuntivi.

Si completi la seguente implementazione di un `HorizontalDraw`:

```
public final class HorizontalDraw extends Draw {
  private final Draw[] children;

  public HorizontalDraw(Draw... children) throws IllegalArgumentException {
    if (children.length == 0)
      throw new IllegalArgumentException("HorizontalDraw: one Draw is needed at least");
    this.children = children;
  }
  // qui dovete continuare voi
}
```

Esercizio 5 [6 punti] Un `FrameDraw` è un `Draw` che inserisce un altro `Draw` (il *figlio*) dentro una cornice di `@`. Per esempio, il `FrameDraw` che ha come figlio l'`HorizontalDraw` dell'esercizio precedente è:

```
@@@@@@@@@@@@@@
@X___X_______@
@_X_X__*___*_@
@__X___*___*_@
@_X_X__*___*_@
@X___X_*****_@
@______*___*_@
@______*___*_@
@______*___*_@
@@@@@@@@@@@@@@
```

Si completi la seguente implementazione di un `FrameDraw`:

```
public final class FrameDraw extends Draw {
  private final Draw child;

  public FrameDraw(Draw child) {
    this.child = child;
  }
  // qui dovete continuare voi
}
```

Se tutto è corretto, il seguente `main`:

```
public class Main {
  public static void main(String[] args) {
    Draw h = new H(), e = new E(), l = new L(), o = new O(), star = new Star();
    Draw hello = new HorizontalDraw(star, h, e, l, l, o, star);
    Draw frame = new FrameDraw(hello);
    System.out.println(new HorizontalDraw(hello, frame));
  }
}
```

stamperà:

```
X   X                                   X   X@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
 X X  *   *  *****  *      *      *****  X X @X   X                                   X   X@
  X   *   *  *      *      *      *   *   X  @ X X  *   *  *****  *      *      *****  X X @
 X X  *   *  *      *      *      *   *  X X @  X   *   *  *      *      *      *   *   X  @
X   X *****  *****  *      *      *   * X   X@ X X  *   *  *      *      *      *   *  X X @
      *   *  *      *      *      *   *      @X   X *****  *****  *      *      *   * X   X@
      *   *  *      *      *      *   *      @      *   *  *      *      *      *   *      @
      *   *  *****  *****  *****  *****      @      *   *  *      *      *      *   *      @
                                             @      *   *  *****  *****  *****  *****      @
                                             @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```