Laboratorio di Programmazione
Edizione 1 - Turni A, B, C

ESAME del 10 Luglio 2015

Avvertenze

- Nello svolgimento dell'elaborato è possibile usare qualunque classe delle librerie standard di Java.
- Non è invece ammesso l'uso delle classi del package `prog` allegato al libro di testo del Prof. Pighizzini e impiegato nella prima parte del corso.
- Si consiglia CALDAMENTE l'utilizzo dello script "checker.sh" per compilare ed effettuare una prima valutazione del proprio elaborato. Si consiglia anche di leggere il sorgente dei `Test_*.java` per capire cosa devono offrire le classi da sviluppare.
- Ricordarsi, quando si programma: *Repetita NON iuvant* o DRY (*Don't Repeat Yourself*).

Tema d'esame

Lo scopo dell'esercizio è realizzare un'applicazione che gestisca la sottomissione di articoli ad una conferenza.
Una conferenza ha un titolo e alcuni vincoli per gli articoli che vengono proposti, tipicamente un numero di pagine (dati il numero di righe per pagina e il numero di caratteri per riga).
Il sistema della conferenza deve accettare gli articoli solo se rispettano i vincoli, poi deve poter assegnare un revisore agli articoli che ne sono ancora privi.
Le **classi** da realizzare sono le seguenti (dettagli nelle sezioni successive):

- `Conferenza`: la classe principale, conterrà gli articoli e fornirà alcuni servizi di query
- `Articolo`: rappresenta un testo con un autore e un titolo
- `Persona`: classe ASTRATTA, rappresenta una persona generica con nome, cognome e mail
- `Autore`: sottoclasse di Persona, oltre agli attributi di Persona ha anche un ente di appartenenza
- `Revisore`: sottoclasse di Persona, oltre agli attributi di Persona ha anche un campo "competenza"

Specifica delle classi

Le classi (**pubbliche**!) dovranno esporre almeno i metodi e costruttori **pubblici** specificati, più eventuali altri metodi e costruttori se ritenuti opportuni. Gli attributi (campi) delle classi devono essere *privati*. per leggere e modificarne i valori, creare opportunamente, e solo dove necessario, i metodi di accesso (`set` e `get`). Se si usano tipi generici, si suggerisce di utilizzarne le versioni opportunamente istanziate (es. `ArrayList<String>` invece di `ArrayList`). Ogni classe deve avere il metodo `toString` che rappresenti lo stato delle istanze.

public class Conferenza

Ha come attributi: titolo della conferenza, numero di pagine massimo per gli articoli, numero di righe per pagina, numero di caratteri per riga. Inoltre deve poter contenere gli articoli, usare un "container" di propria scelta. Oltre agli opportuni costruttori deve disporre dei seguenti metodi pubblici:

- `public Articolo[] articoliDiUnAutore(Autore aut)` restituisce gli articoli di un singolo autore
- `public String[] autori()` restituisce l'elenco degli autori
- `public int nrArticoli()` restituisce il numero di articoli
- `public int quantiArticoliNonHannoRevisore()` restituisce quanti articoli non hanno ancora un revisore assegnato
- `public int submit(Articolo art)` verifica che l'articolo rispetti i vincoli; se sí, accetta e restituisce l'ID (si usi come ID la posizione nel 'container'), altrimenti '-1'
- `public String[] titoli()` restituisce l'elenco dei titoli degli articoli

public class Articolo

Ha come attributi: titolo dell'articolo, autore (reference all'Autore), testo (verrà caricato da un file), revisore (reference al Revisore assegnato). Deve avere almeno il seguente costruttore:

- `public Articolo(String filename)` legge il file ed estrae le info (prima riga del file = titolo, seconda riga = autore, resto del file = testo)

Inoltre deve disporre dei seguenti metodi pubblici:

- `public Autore getAutore()` restituisce l'Autore
- `public Revisore getRevisore()` restituisce il Revisore assegnato, null se non è stato assegnato
- `public String getTitolo()` restituisce il titolo dell'articolo
- `int pagine(int righePerPagina, int caratteriPerRiga)` restituisce il numero di pagine calcolato in base al testo e al numero di righe per pagina e al numero di caratteri per riga
- `public int righe(int caratteriPerRiga)` restituisce il numero di righe dato un numero di caratteri per riga
- `public void setRevisore(Revisore rev)` assegna un Revisore

public abstract class Persona

Classe astratta che rappresenta una persona generica, ha come attributi: nome, cognome e mail. I campi non devono essere vuoti ("") né nulli (`null`), in particolare il campo mail deve contenere almeno il carattere "@", se non sono soddisfatti questi vincoli va lanciata una eccezione.
Definire costruttori e set&get opportuni.

public class Autore

Sottoclasse di Persona, aggiunge solo il campo 'ente' che non deve essere né vuoto né nullo, lanciare eccezione se non sono verificati questi vincoli.
Definire costruttori e set&get opportuni.

public class Revisore

Sottoclasse di Persona, aggiunge solo il campo 'competenze' che non deve essere né vuoto né nullo, lanciare eccezione se non sono verificati questi vincoli.
Definire costruttori e set&get opportuni.

Raccomandazioni

Affinché l'elaborato sia valutato, è richiesto che sia le classi sviluppate che i test risultino *compilabili*. A tal fine i metodi/costruttori che non saranno sviluppati dovranno comuque avere una implementazione fittizia come la seguente:

```
public void nomeDelMetodo () {
   throw new UnsupportedOperationException();
}
```

Si suggerisce quindi di dotare da subito le classi di tutti i metodi richiesti, implementandoli in modo fittizio, e poi di sostituire man mano le implementazioni fittizie con implementazioni che rispettino le specifiche.

Consegna

Si ricorda che le classi devono essere tutte *public* e che vanno consegnati tutti (e soli) i file *.java* prodotti. NON vanno consegnati i *.class*. NON vanno consegnati i file relativi al meccanismo di autovalutazione (*Test_*.java, AbstractTest.java, *.sh*). Per la consegna, eseguite l'upload dei SINGOLI file sorgente (NON un file archivio!) dalla pagina web: http://upload.di.unimi.it nella sessione del vostro docente.

*** ATTENZIONE!!! ***

NON VERRANNO VALUTATI GLI ELABORATI CON ERRORI DI COMPILAZIONE O LE CONSEGNE CHE NON RISPETTANO LE SPECIFICHE (ad esempio consegnare un archivio zippato è sbagliato, come anche consegnare ad un docente diverso dal proprio assegnato).
UN SINGOLO ERRORE DI COMPILAZIONE O DI PROCEDURA INVALIDA **TUTTO** L'ELABORATO.

Per ritirarsi fare l'upload di un file vuoto di nome `ritirato.txt`. Se avete caricato dei file nella sessione del docente sbagliato, caricate lí un file vuoto di nome `erratoConsegna.txt` e caricate poi i file nella sessione giusta.