ESAME DI FONDAMENTI DI INFORMATICA T-2 del 17/01/2017

Proff. E. Denti – G. Zannoni

Tempo a disposizione: 4 ore

NB: il candidato troverà nell'archivio ZIP scaricato da Esamix anche il software "Start Kit"

NOME PROGETTO ECLIPSE: CognomeNome-matricola (es. RossiMario-0000123456)

L'agenzia pubblicitaria "Vendo & Rivendo" ha richiesto lo sviluppo di un'applicazione per la generazione automatica di annunci pubblicitari standard, da proporre ai propri clienti a basso costo.

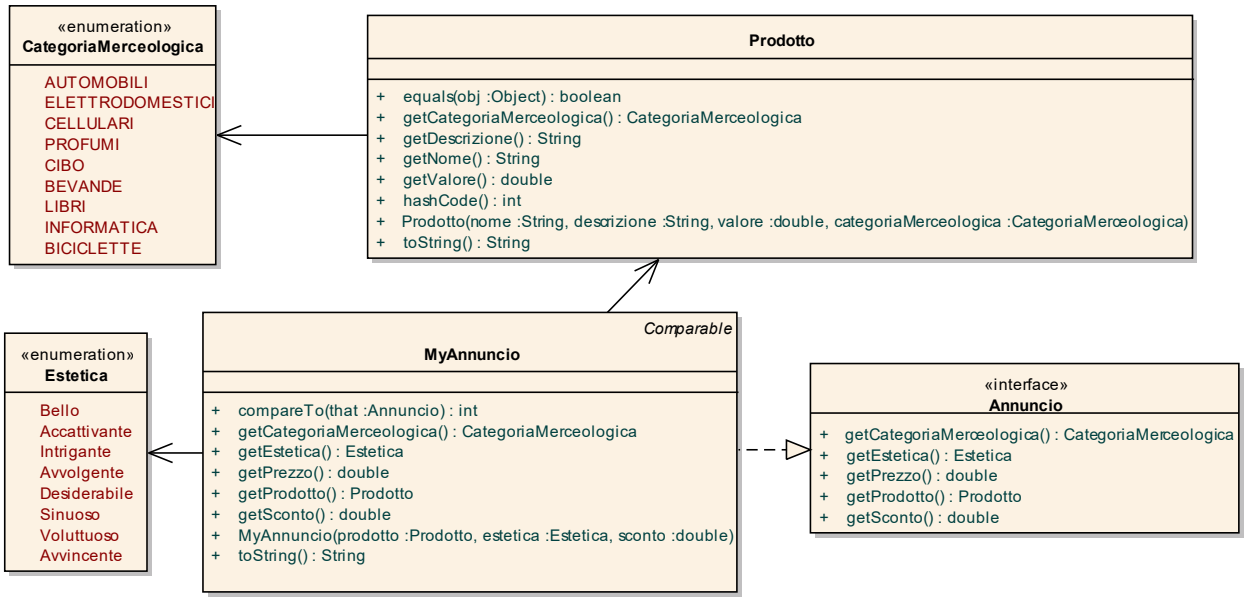
DESCRIZIONE DEL DOMINIO DEL PROBLEMA

Un *Prodotto* è un bene caratterizzato da nome, descrizione, valore e categoria merceologica. Ogni *Annuncio* si riferisce alla vendita di uno specifico *Prodotto* ed è caratterizzato, oltre che dal *Prodotto* stesso, ad una estetica (un aggettivo), uno sconto ed al prezzo di vendita il quale è calcolato partendo dal valore del *Prodotto* e dallo sconto.

Parte 1 (punti: 14)

Dati (package vendoerivendo.model) *(punti: 6)*

Il modello dei dati deve essere organizzato secondo il diagramma UML più sotto riportato.



SEMANTICA:

a) L'enumerativo ***CategoriaMerceologica*** (fornito nello start kit) rappresenta le diverse categorie merceologiche;

b) L'enumerativo ***Estetica*** (fornito) rappresenta una serie di aggettivi per definire l'estetica dell'oggetto;

c) La classe ***Prodotto*** (fornita) rappresenta un prodotto con le sue caratteristiche (nome, descrizione, valore in Euro e categoria merceologica);

d) L'interfaccia ***Annuncio*** (fornita), rappresenta un annuncio con le sue caratteristiche (Prodotto, estetica, sconto applicato, prezzo finale);

e) La classe ***MyAnnuncio* (da realizzare)** rappresenta un annuncio concreto, implementa le interfacce ***Annuncio*** e ***Comparable*** e prevede un solo costruttore ***MyAnnuncio(Prodotto, Estetica, sconto)*** che costruisce un annuncio per il prodotto descritto dal primo argomento, completandolo con un aggettivo per definirne l'estetica (enumerativo ***Estetica***) e lo sconto da applicare al valore del Prodotto per calcolarne il prezzo finale. È compito del costruttore verificare la correttezza formale degli argomenti, lanciando nel caso una ***IllegalArgumentException*** con specifico, idoneo messaggio d'errore.

Questa classe espone inoltre i seguenti metodi:

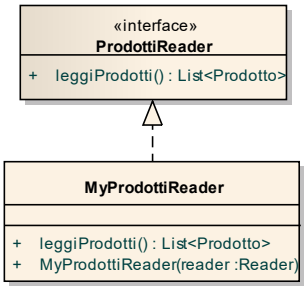
- ***compareTo*** che ordina gli annunci per categoria del prodotto (crescente), e in subordine per estetica (decrescente) e prezzo (crescente); si ricordi che gli enumerativi sono già confrontabili;
- ***toString*** che restituisce il testo completo dell'annuncio, ottenuto concatenando opportunamente la descrizione (***toString*** del ***Prodotto***), l'***Estetica*** e il prezzo proposto.

Persistenza (package vendoerivendo.persistence) ***(punti 8)***

Il file di testo Prodotti.txt contiene i dati di alcuni prodotti da pubblicizzare, uno per riga, con i rispettivi dati **separati da tabulazioni**: nell'ordine, per ogni prodotto sono elencati nome, prezzo, descrizione (che può contenere spazi) e categoria merceologica.

L'interfaccia ***ProdottiReader*** (fornita) dichiara il metodo ***leggiProdotti***, che legge una lista di Prodotti, lanciando ***IOException*** o ***BadFileFormatException*** secondo necessità.

La classe *MyProdottiReader* (da realizzare) implementa tale interfaccia: il costruttore riceve un Reader da cui effettuare la lettura.



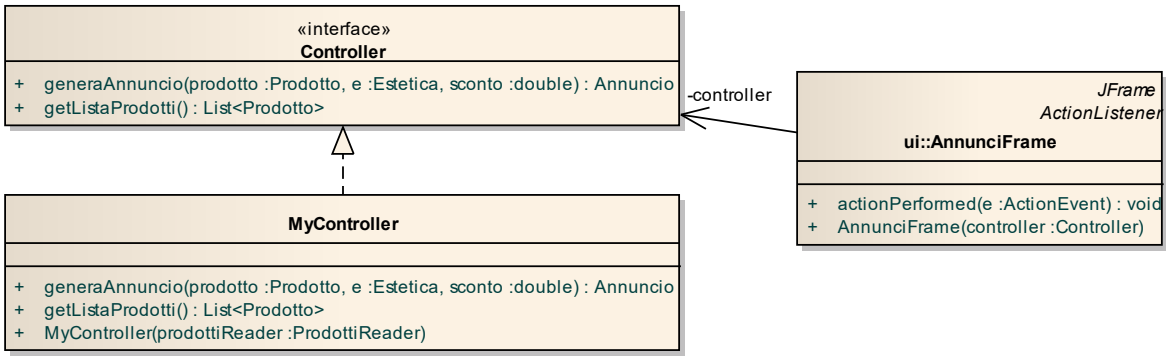
Parte 2 (punti: 16)

L'applicazione deve permettere all'utente di scegliere un prodotto (da una opportuna combobox precaricata), l'estetica, la fascia di prezzo in cui collocarlo e generare il corrispondente annuncio, mostrandolo in un'area di testo.

Controller (package vendoerivendo.controller) ***(punti 6)***

L'interfaccia ***Controller*** (fornita) dichiara i metodi del controller dell'applicazione; in particolare, il metodo ***generaAnnuncio*** produce un ***Annuncio*** per il prodotto specificato, mentre il metodo ***getListaProdotti*** restituisce la lista dei prodotti disponibili.

La classe ***MyController (da realizzare)*** implementa tale interfaccia: il costruttore riceve un ***ProdottiReader*** ed effettua la lettura, memorizzando internamente la lista dei prodotti letti, recuperabile tramite ***getListaProdotti***; eventuali eccezioni **non** devono essere catturate.



Interfaccia utente (package vendoerivendo.ui) ***(punti 10)***

L'interfaccia utente deve essere simile (non necessariamente identica) all'esempio mostrato di seguito.

La classe ***AnnunciFrame (da realizzare totalmente)*** realizza la GUI sotto illustrata: il suo costruttore riceve come argomento il Controller. Se il caricamento dei dati ha esito positivo, compare la finestra principale (Fig. 1. L'utente deve quindi scegliere prodotto ed estetica dalle opportune combobox e inserire uno sconto da applicare nel **JFormattedTextField** (Fig. 2) e poi premere il tasto **Genera annuncio**, che mostra il testo dell'annuncio richiesto nella

textarea (Fig. 3). Il pulsante **Stampa annuncio** provvede a generare l'annuncio e a salvarlo (senza mostrarlo nella textarea) tramite opportuno ***PrintWriter***, nel file di testo scelto dall'utente.

Il **JFormattedTextField** prende nel costruttore un **java.text.NumberFormat**. In questo caso, è sufficiente crearne uno (metodo **getPercentInstance** – tale formattatore considera percentuali valide solo i numeri seguiti dal segno %) configurandolo per formattare numeri con nessuna cifra decimale. Per ottenere dal **JFormattedTextField** il valore inserito dall'utente, usare il metodo **getValue** che restituisce un **Number**; se il valore inserito dall'utente non è una percentuale valida, allora **getValue** restituisce **null**. Il metodo **doubleValue** di **Number** consente di ottenere il valore desiderato.

Nota bene: si ricordi che il formattatore di percentuali consente all'utente di inserire valori percentuali (ad es. 50%, 80%, ecc.) ma restituisce tali valori divisi per 100 (ad es. 0.5, 0.8, ecc.).

Per qualsiasi scopo si generi l'annuncio (visualizzazione o stampa), occorre verificare che l'annuncio sia effettivamente generabile; nel caso non lo sia (Prodotto non selezionato, Estetica non selezionata, sconto non inserito correttamente o con valore non compreso fra 0 e 100 escluso), va segnalato l'errore all'utente mediante l'uso di un opportuno **JOptionPane** (metodo **showMessageDialog**).

[SUGGERIMENTO: utilizzare un JFileChooser per consentire all'utente di selezionare un file di testo, sfruttando il metodo showSaveDialog]

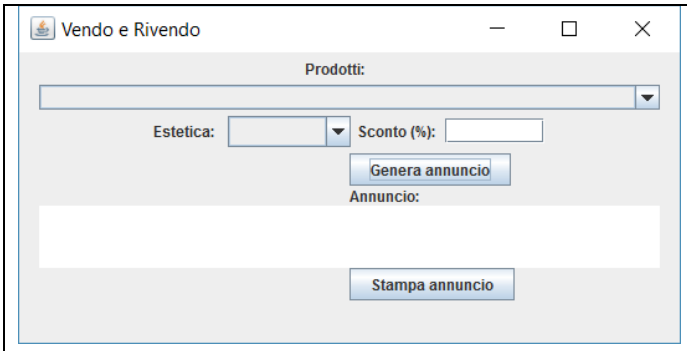


Fig. 1

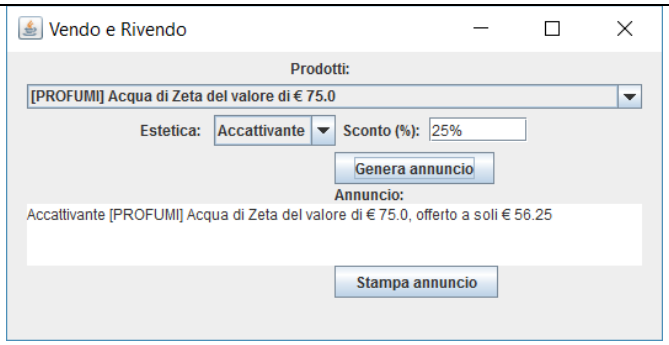


Fig. 2

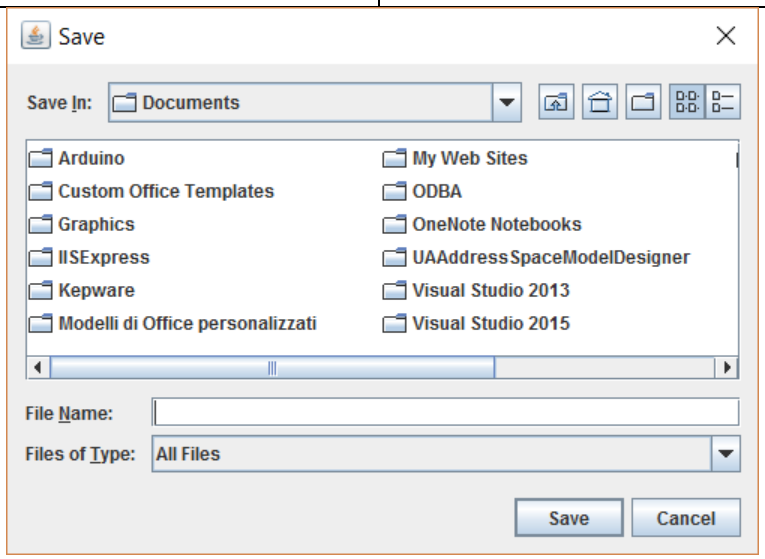


Fig. 4

La classe ***Program*** (non mostrata nel diagramma UML, ma fornita nello start kit) contiene il main di partenza dell'intera applicazione; la classe accessoria ***GUITest*** contiene un main ausiliario, in grado di far girare l'interfaccia grafica *anche in assenza di persistenza funzionante*.