LabProg 2010-2011 - TESTO ESAME
Andrea Trentini - DICo - UniMi
28 febbraio 2011

Avvertenza. Non è ammesso l'uso delle classi del package *prog.io* allegato al libro di testo del corso.

1 Testo esame

Lo scopo è realizzare la gestione di un ristorante, per la parte riguardante il menu e il conto. Le classi da realizzare sono le seguenti (dettagli nel seguito):

- *Menu*: il menu delle pietanze disponibili
- *(abstract) VoceDimenu*: una generica voce di menu (può essere un Piatto o una Bevanda)
- *(abstract) Piatto*: un piatto disponibile per l'ordinazione
- *(abstract) Bevanda*: una bevanda disponibile per l'ordinazione
- *Tavolo*: un tavolo, numerato
- *Main*, contiene il solo metodo *main()* con alcune invocazioni di test (importante è riuscire a testare TUTTI i metodi almeno una volta)

Tali classi dovranno esporre almeno i metodi specificati nelle sezioni seguenti. Eventuali metodi di servizio possono essere aggiunti a piacimento. Ogni classe deve avere il *toString()* che rappresenti lo stato delle istanze e i **costruttori** adeguati per gli attributi che vengono dichiarati. Creare opportunamente i metodi di accesso agli attributi (set&get). Si suggerisce, anche dove non segnalato, di utilizzare, se esistenti e se applicabili, le classi parametriche (es. *Vector<E>* invece di *Vector*). Alcuni controlli di coerenza vengono suggeriti nel testo, potrebbero essercene altri a discrezione. I tipi di ritorno possono essere variati (ad es. boolean invece di void se si vuole ottenere un feedback sul successo dell'operazione) previa autorizzazione del docente. Consiglio di posporre l'implementazione dei controlli di coerenza, fatelo come ultima operazione, prima realizzate un sistema funzionante.

1.1 Menu

Contiene degli attributi (di tipo opportuno) per memorizzare la lista delle voci di menù, e i metodi di accesso alla lista stessa. Oltre ai costruttori opportuni deve esporre i metodi **pubblici** di accesso ai dati, con controlli di coerenza. Deve inoltre esporre i seguenti metodi **pubblici**:

- *VoceDiMenu[] getItem(String keyword)*: restituisce un elenco delle voci di menu che "contengono" la parola chiave fornita

Suggerimento per chi vuole dimostrare particolare abilita' programmativa: implementare un metodo *iteratore()* che fornisca un iteratore degli item nel menu. Non impelagatevi in questo punto se non vi sentite sicuri di voi!

1.2 (abstract) VoceDiMenu

Serve come superclasse generica per Piatto e Bevanda. Deve contenere la rappresentazione del prezzo (con i relativi metodi set&get) e una stringa per la descrizione dell'item (servirà anche per la ricerca per chiave del *getItem()* di Menu. Deve inoltre esporre i seguenti metodi **pubblici**:

- *void ordina(Tavolo t)*: assegna questa voce al tavolo "t" (cioe' rappresenta un'ordinazione di un item di questo tipo per il tavolo), simmetrico del *ordina(VoceDiMenu v)* di Tavolo

1.3 (abstract) Piatto+DERIVATE

Sottoclasse di VoceDiMenu che rappresenta la topclass dei piatti (cibarie). Pressochè vuota. DEVE essere abstract perché in realtà vivrà sotto forma di sottoclassi (**che dovete creare**) concrete, ad es.: Spaghetti, Bistecca, Carne, Pesce, etc.

1.4 (abstract) Bevanda+DERIVATE

Contiene degli attributi (di tipo opportuno) per rappresentare una bevanda generica. DEVE essere abstract perché in realtà vivrà sotto forma di sottoclassi (**che dovete creare**) concrete, ad es.: Acqua, Vino, Bibita, etc.

1.5 Tavolo

Contiene degli attributi (di tipo opportuno) per memorizzare nr. del tavolo, persone sedute (coperti) e le ordinazioni. Oltre ai costruttori opportuni deve esporre i metodi **pubblici** di accesso ai dati, con controlli di coerenza. Deve esporre i seguenti metodi **pubblici**:

- *void conto()*: stampa a video il dettaglio delle pietanze ordinate e il conto totale (col nr. degli item consumati)
- *VoceDiMenu[] getOrdinazioni()*: restituisce un elenco delle voci di menu associate a questo tavolo (pietanze ordinate)
- *void ordina(VoceDiMenu v)*: assegna una voce al tavolo (cioe' rappresenta un'ordinazione di un item di questo tipo per il tavolo), simmetrico del *ordina(Tavolo t)* di VoceDiMenu

1.6 Main

Deve contenere il metodo *main()* in cui vanno istanziate le classi realizzate e opportunamente testate, ad es. creando almeno un tavolo che ordina alcuni piatti e popolando le istanze per avere una situazione reale a regime, con generazione del conto finale.

2 Consegna

Ricordo che le classi devono essere tutte *public* e che vanno consegnati tutti i file *.java* prodotti. NON vanno consegnati i .class Per la consegna, eseguite l'upload dei singoli file sorgente dalla pagina web: http://upload.dico.unimi.it.