LabProg 2010-2011 - TESTO ESAME
Andrea Trentini - DICo - UniMi
20 settembre 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 una "cassa" automatica che gestisce pagamenti ed emette resto.
Le classi da realizzare sono le seguenti (dettagli nelle sezioni successive):

- *Cassa*: accetta un pagamento in ingresso e fornisce un resto
- *Euro*: rappresenta i vari tagli
- *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 Cassa

Deve contenere un "serbatoio" di Euro di vari tagli. Una volta impostato il pagamento accetta monete/banconote in ingresso ed emette un resto in funzione deli tagli che ha a disposizione.
Es. una cassa che contenga 2x10 euro, 3x5 euro, 6x1 euro e debba accettare un pagamento di 12 euro e gli vengano immessi 20 euro (due tagli da 10). Deve accettare il pagamento (perche' la cifra immessa e' superiore al pagamento) e deve generare un resto di 8 euro in tagli da 1x5 euro e 3x1 euro.
Deve inoltre esporre i seguenti metodi **pubblici**:

- *void inserisci(Euro e)*: accetta temporaneamente un taglio in ingresso e lo conteggia
- *boolean paga()*: controlla che le monete/banconote immesse coprano la cifra da pagare e calcola il resto da emettere, se la cifra non basta restituisce *false*.
- *void impostaPagamento(int e)*: imposta la cifra da pagare
- *Euro emettiResto()*: da invocare piu' volte, fino a che tutto il resto non e' stato emesso, restituisce null quando il resto e' finito

Esempio:

```
Cassa c=new Cassa();
// deve partire gia' con
// alcuni euro in... cassa

c.impostaPagamento(12);

c.inserisci(new Banconota(10));

c.inserisci(new Banconota(5));

c.paga();
// deve tornare true
// perche' la cifra immessa
// e' maggiore del richiesto

System.out.println(c.emettiResto());
// 1 euro

System.out.println(c.emettiResto());
// 1 euro

System.out.println(c.emettiResto());
// 1 euro

System.out.println(c.emettiResto());
// null
```

1.2 Euro

(anche eventuali sottoclassi, ad es. Moneta e Banconota, e poi i tagli Dieci, Venti, etc.)
Rappresenta una moneta/banconota da X euro, il valore va impostato in sede di istanziazione e non puo' piu' essere cambiato.
Deve inoltre esporre (almeno, ma altri metodi, ad es. di confronto, potrebbero essere utili) i seguenti metodi **pubblici**:

- *int valore()*: legge il valore numerico della moneta/banconota

1.3 Main

Deve contenere il metodo *main()* in cui vanno istanziate le classi realizzate e opportunamente testate.

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.