Ereditarietà – esercizio 1

PARTE 1

Definire una opportuna gerarchia di classi che rappresenti contribuenti fiscali. La radice di tale gerarchia `e la classe pubblica astratta **Contribuente**, contenuta nel package **contribuente**, contenente i campi privati entrate e uscite di tipo double ed (unicamente) i seguenti metodi:

- public Contribuente(double entrate, double uscite): crea un oggetto di classe Contribuente ed assegna a ciascun campo dati il valore del parametro omonimo;
- abstract public String id(): restituisce l'identificativo del contribuente;
- protected double imponibile(): restituisce l'ammontare dell'imponibile, calcolato secondo la formula imponibile = (entrate−uscite)∗fds, dove fds rappresenta un fattore di sconto dipendente dal tipo di contribuente (v. sotto);
- abstract protected double fattoreDiSconto(): restituisce il fattore di sconto del contribuente (valore compreso tra 0 ed 1);
- abstract protected double aliquota(): restituisce l'aliquota del contribuente (valore com- preso tra 0 ed 1, estremi inclusi);
- public double daPagare(): restituisce l'ammontare totale delle tasse da pagare, ovvero il prodotto dell'imponibile per l'aliquota.

Dopo aver realizzato tale classe, definire nel package **contribuente.persone** la classe pubblica **PersonaFisica**, implementazione di Contribuente, che rappresenta i contribuenti di tipo "persona fisica", contenente un campo dati privato codiceFiscale di tipo String ed il costruttore a tre argo- menti public PersonaFisica(double entrate, double uscite, String codiceFiscale,String nome, String cognome) che crea un oggetto di classe PersonaFisica ed assegna a ciascun campo dati il valore del parametro omonimo. Per l'implementazione dei metodi astratti si considerino i seguenti requisiti:

- l'id di una persona fisica `e il suo codice fiscale;
- il fattore di sconto di una persona fisica `e 0.8;
- l'aliquota di una persona `e pari a: 0 se il suo imponibile `e ≤ 5000, 0.3 altrimenti

Deve inoltre contenere i metodi accessori getNome() e getCognome().

Successivamente, definire nel package **contribuente.aziende** la classe pubblica **Azienda**, implementazione di Contribuente, che rappresenta i contribuenti di tipo "azienda", contenente un campo dati privato partitaIva di tipo String ed il costruttore a tre argomenti public Azienda(double entrate, double uscite, String partitaIva) che crea un oggetto di classe Azienda ed assegna a ciascun campo dati il valore del parametro omonimo. Per l'implementazione dei metodi astratti ereditati dalla classe Contribuente si considerino i seguenti requisiti:

- l'id di un'azienda `e la sua partita IVA;
- il fattore di sconto di un'azienda `e 0.9;
- l'aliquota di un'azienda `e 0.4.

Realizzare infine una opportuna classe di prova che costruisce 4 oggetti di tipo PersonaFisica e 4 oggetti di tipo Azienda ed effettua il test di correttezza dei vari metodi implementati.

PARTE 2

Infine, realizzare nel package **contribuente.utilita** la classe pubblica **Utilita**, contenente i seguenti metodi statici:

- public static **Contribuente[] debitori(Contribuente[] contribuenti, Double[] pagamentiPervenuti)** che, data una lista contribuenti di oggetti di classe Contribuente ed una lista pagamentiPervenuti di oggetti di classe Double, restituisce l'insieme degli elementi c di contribuenti il cui valore restituito da c.daPagare() sia minore di quello contenuto nell'elemento che in pagamentiPervenuti occupa la stessa posizione di c. Ad esempio, se per il terzo elemento di contribuenti il valore di daPagare() `e 10 ed il terzo elemento di pagamentiPervenuti contiene 11, allora il terzo elemento di contribuenti dovr`a essere incluso nell'insieme risultato. Si pu`o assumere che contribuenti e pagamentiPervenuti siano della stessa dimensione.
- public static **Azienda[] aziendeDebitrici(Contribuente[] debitori)** che, dato un insieme debitori di oggetti di classe Contribuente, restituisce tutti gli oggetti di debitori che sono anche di classe Azienda.
- public static **PersonaFisica cerca(String nome, String cognome,Contribuente[] lista)** che dato il nome e cognome, restituisce la prima PersonaFisica con quel nome e cognome presente nella lista o null se non ne esiste nessuna.