Esame del Corso di Tecniche di Programmazione

Canale di Ingegneria delle Reti e dei Sistemi Informatici - Polo di Rieti)

Prof. Paolo Romano

Appello del 20/12/2007

Si vogliono gestire le collezioni di libri di biblioteche. Ogni oggetto *Biblioteca* ha un nome (una stringa) ed una collezione di libri. Ciascun libro è caratterizzato da un:

- *codice univoco* (un intero);
- *autore* (una stringa);
- *titolo* (una stringa).

Gli oggetti *Biblioteca* supportano le seguenti funzionalità:

- *crea:* che, data una stringa *n* che rappresenta il nome della biblioteca, crea un oggetto *Biblioteca* con nome *n* che inizialmente non ha alcun libro;
- *nome:* che restituisce il nome della *Biblioteca*;
- *aggiungi:* dati il codice, l'autore e il titolo di un libro, lo aggiunge alla biblioteca; se è già presente un libro con lo stesso codice genera una eccezione;
- *autore:* che, dato il codice di un libro, ne restituisce l'autore; se non esiste un libro con quel codice genera una eccezione;
- *titolo:* che, dato il codice di un libro, ne restituisce il titolo; se non esiste un libro con quel codice genera una eccezione;
- *elimina:* che, dato il codice di un libro, lo elimina dalla biblioteca; se il libro non è presente nella biblioteca genera una eccezione;
- *quantiLibriAutore:* che, dato un autore, restituisce il numero di suoi libri presenti nella biblioteca;
- *eliminaLibriAutore:* che, dato un autore, elimina dalla biblioteca tutti i suoi libri;
- *tuttiLibriAutore:* che, dato un autore, restituisce un array di interi contenente i codici di tutti i libri dell'autore specificato; oppure *null* se non esistono libri per l'autore specificato;
- *tuttiAutori:* che restituisce un array di stringhe contenente i nomi di tutti gli autori i cui libri sono presenti nella biblioteca;
- *totaleLibri*: che restituisce il numero totale di libri nella biblioteca.

Domanda 1. Scrivere una classe Java `Biblioteca` per rappresentare oggetti *Biblioteca*. Fornire il costo asintotico in tempo e spazio di ciascun metodo indicando esplicitamente i parametri di input della funzione di costo, l'istruzione dominante e descrivendo il caso peggiore (qualora necessario).

Domanda 2. Descrivere come modificare la classe Biblioteca per garantire che la collezione di libro sia ordinata rispetto al titolo.

Domanda 3. Scrivere un metodo statico `verificaAutori`, cliente della classe `Biblioteca`, che data una lista di stringhe *l* rappresentante una lista di autori di libri, restituisca una lista di nomi di autori (appartenenti alla lista *l*) di cui la biblioteca non contiene alcun libro. Il metodo deve avere un'implementazione funzionale. Le liste di autori sono memorizzate in oggetti della classe `Lista`,

avente la seguente interfaccia (sostituire a <TYPE> il tipo opportuno):

```
class Lista {
        public Lista();
        public boolean vuota(); //restituisce true se la lista è vuota
        public <TYPE>  primo(); //restituisce il primo elemento della lista
        public Lista  inserisciPrimo(<TYPE> x); //restituisce una lista ottenuta inserendo x come primo elemento
        public Lista  eliminaPrimo(); //restituisce una lista ottenuta eliminando il primo elemento
}
```

Specificare il costo asintotico in tempo e spazio del metodo indicando esplicitamente la dimensione dell'input, l'istruzione dominante e descrivendo il caso peggiore quando necessario. Le operazioni della classe Lista hanno costo O(1).

Domanda 4. Realizzare un metodo statico pubblico `SommaNonFoglie`, che, dato il riferimento alla radice di un albero binario *alb*, i cui nodi contengono interi, restituisca la somma di tutti gli interi contenuti nei nodi di *alb* che 1) non sono foglie e, 2) il cui valore sia maggiore della somma dei propri figli. Fornire il costo in tempo e spazio di memoria del metodo realizzato, motivando la risposta (indicando esplicitamente i parametri di input della funzione di costo ed il caso peggiore).

Domanda 5. Cosa si intende per aggregazione e per composizione di oggetti? Si fornisca almeno un esempio di entrambe le tipologie di associazioni.

Domanda 6. Si descriva il meccanismo del poliformismo in linguaggi object oriented, discutendone i principali vantaggi e fornendo un esempio. Si illustrino quindi i principali meccanismi impiegati a run-time dalla JVM a supporto del polimorfismo.