Metodi Avanzati di Programmazione
Corso di Laurea in Informatica
Anno Accademico 2013/2014
Prova scritta del 09/06/2014 ore 9:00-12:00

1) Fornire le specifiche algebriche (semantiche e di restrizione), **in forma di equazioni**, per il tipo astratto *Matrice* di numeri interi di cui si forniscono le seguenti specifiche sintattiche:

Tipi:

Matrice, Intero, Boolean

Operatori:

creaMatrice(Intero) --> Matrice *// crea una matrice quadrata di zeri, il numero di righe e colonne è passato come argomento*

assegna(Matrice, Intero, Intero,Intero)-->Matrice *// assegna nella posizione individuata dai primi due parametri (riga e colonna), il valore intero passato come terzo parametro*

leggi(Matrice,Intero,Intero)-->Intero *// restituisce l'intero memorizzato nella posizione specificata dai parametri*

diagonale(Matrice)-->Vettore *// restituisce il vettore degli interi lungo la diagonale principale*

contaNonZeri(Matrice)-->Intero *// conta i valori diversi da zero*

uguale(Matrice, Matrice)--> Booleano *// restituisce vero se le due matrici contengono gli stessi valori nelle stesse posizioni, falso altrimenti*

somma(Matrice,Matrice)--> Matrice *// restituisce la matrice che è somma delle due passate come parametri.*

Si assuma l'esistenza di *creaVettore*(Intero)--> Vettore (crea un vettori di zeri, la dimensione del vettore è specificata come parametro) e *assegnaVettore* (Vettore,Intero,Intero)--> Vettore (assegna il valore passato come terzo parametro nella posizione specificata come secondo parametro del vettore)

(7 punti)

2) Fornire una definizione di tipo astratto in ADA. Scrivere il codice ADA per tipo astratto **Matrice** realizzazione di un dato astratto avente specifiche sintattiche:
creaMatrice(Intero) --> Matrice **assegna**(Matrice, Intero,Intero,Intero)-->Matrice **leggi**(Matrice,Intero,Intero)-->Intero **uguale**(Matrice, Matrice)--> Booleano e **somma**(Matrice,Matrice)--> Matrice
Spiegare come usare questo tipo astratto e le operazioni per esso definite. Riportare le istruzioni per questo uso in ADA. **Commentare il codice scritto.** **(7 punti)**

3) Definire la relazione di **ereditarietà per implementazione** nel paradigma OO. Fornire un esempio UML di classi definite usando e la relazione di ereditarietà per implementazione. Tradurre tale esempio in Java. **(5 punti)**

4) Identificare il contenitore più appropriato per la modellazione di un **insieme** di prodotti **ordinati** rispetto a un qualche criterio (InsiemeProdottiOrdinato). Ciascun Prodotto è modellato dai campi, codice prodotto, nome, prezzo.
I criteri per l'ordinamento possibili sono due: ordinare rispetto a nome oppure ordinare rispetto a codice prodotto. Implementare la classe InsiemeProdottiOrdinato e le classi ad esso correlate in Java. Mostrare l'istanziazione di tali classi (con ciascuno dei criteri di ordinamento fissato), il popolamento degli insiemi e la stampa del contenuto degli insiemi. **Commentare il codice scritto.**

(7 punti)

5) Descrivere il ciclo di vita di un Thread. Scrivere un programma che crei cinque **Thread daemon** usando l'interfaccia **Runnable**. Ciascun Thread stampa a video la progressione di base X (X è un parametro) definita come segue.

X X+1 X+2 X+3 … **(7 punti)**