Metodi Avanzati di Programmazione
Corso di Laurea in Informatica
Anno Accademico 2011/2012

Prova scritta del 04/09/2012 ore 14:30-17:30

1) Descrivere la astrazione di funzione e spiegare l'uso delle funzioni quali cittadini di prima classe o seconda classe o terza classe nei linguaggi di programmazione. **(4 punti)**
2) Dare le specifiche algebriche (semantiche e di restrizione) per il tipo astratto ReteComputer di cui si forniscono le seguenti specifiche sintattiche:

Tipi:
ReteComputer,Computer, ListaComputer, Intero, Booleano;

Operatori:

creaRete ()→ReteComputer
insComputer (ReteComputer, Computer)→ReteComputer (*)
collegaComputer(ReteComputer, Computer,Computer)→ReteComputer (**)
cancellaComputer(ReteComputer,Computer) →ReteComputer
appartieneComputer(ReteComputer,Computer)→Booleano *// verifica se un computer appartiene alla rete*
contaLegami(ReteComputer,Computer)→Intero *//conta il numero di legami che interessano direttamente il parametro computer nella rete*
listaCollegamenti(ReteComputer,Computer))→ListaComputer *//crea una lista che contiene tutti i computer direttamente collegati al parametro computer all'interno della rete*
differenza(ReteComputer,ReteComputer) → ListaComputer *// restituisce la lista dei computer che appaiono nella prima rete, ma non nella seconda*
(*) insComputer() aggiunge un Computer che non ha legami con alcun altro Computer nella rete
(**)collegaComputer () crea un legame tra due Computer che fanno già parte della rete.
Si assume l'esistenza degli operatori creaLista()→ListaComputer e insLista(ListaComputer,Computer) → ListaComputer **(7 punti)**

3) Descrivere le forme di **polimorfismo ad hoc** nella classificazione di Cardelli-Wegner **(4 punti)**
4) Spiegare la serializzazione in Java. Con riferimento alle classi definite del seguito:

```
class Computer
{
        String ip;
}
class Rete{
        static int  generatoreCodice=0;
        transient String password;
        codice= generatoreCodice++;
        HashMap<Computer,List<Computer>> rete = new HashMap<Computer,ArrayList<Computer>>();
        Rete (Stringa psw){
                    password=psw;

        }
        void insComputer(Computer c){
                    rete.add(c, new ArrayList<Computer>());
        }
        void collegaComputer(Computer c1, Computer c2)
        {
                    rete.get(c1).add(c2);
                    rete.get(c2).add(c1);
        }
}
```

Scrivere il codice Java dei metodi *salva(...)* e *carica(...)* che fanno parte della classe Rete e permettono di salvare l'oggetto corrente su un file (salva) e caricare una istanza di Rete da file per restituirla (carica). Scrivere inoltre un main che sia in grado di caricare una istanza di Rete da file e aggiungervi nuovi computer e legami. **(5 punti)**

5) Spiegare il meccanismo di gestione delle eccezioni in Java. Con riferimento al codice riportato nell'esercizio 4 , definire la eccezione controllata ComputerException da sollevare sia se si tenta di inserire nella rete un computer già esistente sia se si prova a collegare due computer inesistenti nella rete. Modificare il codice scritto al fine di sollevare e gestire la suddetta eccezione. **(5 punti)**

6) Spiegare il ciclo di vita di un applet. Creare un applet JAVA (comprensiva di tag per la visualizzazione tramite appletviewer) che abbia una interfaccia utente di tipo **responsive**. Tale applet deve includere una JTextField T1 e due JButton B1 e B2 (**allineati orizzontalmente**). T1 inizialmente mostra un contatore che viene incrementato di uno ogni 100 millisec. Se l'utente preme su B1 allora si raddoppia la velocità di attesa corrente tra un numero e il successivo, se preme B2 allora si dimezza la velocità di attesa corrente tra un numero e il successivo. **Commentare il codice scritto.** **(8 punti)**