Esame di Fondamenti di Informatica L-B
Ingegneria Gestionale e dei Processi Gestionali (L-Z)

Appello del 14/9/2009

Esercizio 1 (4 punti)

Modalità di gestione delle stringhe in Java.

Esercizio 2 (6 punti)

Siano dati i seguenti metodi Java:

```
public static int f(int V[], int M, int N) {
  int i=M-N, sum=0;
  while(i<=M+N)
    sum+=V[i++];
  return sum;
}

public static int g(int V[], int N) {
  int sum=0;
  for(int j=0; j<=N/2; j++)
    sum+=f(V, j, j);
  return sum;
}
```

1. Calcolare la complessità in passi base del metodo `f` nei termini dei parametri `M` ed `N`.
2. Calcolare la complessità in passi base del metodo `g` nei termini del parametro `N` (si supponga `N` dispari).
3. Calcolare la complessità asintotica del metodo `g` nei termini del parametro `N`.

Esercizio 3 (7 punti)

La cantina sociale di Santo Stefano al Monte vuole organizzare all'interno di un calcolatore le informazioni sui vini prodotti dalle cantine che partecipano alla cooperativa. In particolare, l'uva proveniente da ogni produttore viene registrata memorizzandone i dati relativi all'uvaggio (es. Cabernet Franc, Chardonnay, Sangiovese), la quantità (in quintali) di uva prodotta ed il prezzo al chilo. Si scriva una classe `Uva` per la cantina sociale di Santo Stefano al Monte che:

1. Possieda un opportuno costruttore con parametri.
2. Presenti opportuni metodi che permettano di accedere alle variabili di istanza dell'oggetto.
3. Possieda il metodo `costo` che calcoli il costo totale, come prezzo x peso.
4. Presenti il metodo `toString` che fornisca la descrizione dell'uva.
5. Possieda il metodo `equals` per stabilire l'uguaglianza con un altro oggetto `Uva` (l'uguaglianza va verificata solo sul tipo di uvaggio).
6. Implementi l'interfaccia `Comparable`, definendo il metodo `compareTo` per stabilire la precedenza con un oggetto `Uva` passato come parametro (la precedenza va data all'oggetto avente il minor costo totale).

Esercizio 4 (7 punti)

Si scriva una classe `Cantina` che memorizzi le informazioni relative alle uve prodotte da ciascuna cantina. Oltre al nome ed all'indirizzo della cantina, occorre memorizzare i dati relativi alle uve prodotte all'interno di un insieme. La classe `Cantina` deve inoltre:

1. Presentare un opportuno costruttore (inizialmente l'insieme delle uve prodotte è vuoto).
2. Possedere un metodo `aggiungi` che, dato un oggetto `Uva`, lo inserisca all'interno dell'insieme, controllando che l'insieme non contenga già un oggetto identico.
3. Presentare un metodo `cerca` che, dato il nome di un uvaggio, restituisca il prezzo al chilo delle uve di tale uvaggio, se esso è prodotto dalla cantina, ed un valore negativo, altrimenti.
4. Possedere un metodo `valoreTotale` che restituisca il costo totale di tutte le uve prodotte dalla cantina.
5. Possedere il metodo `toString` che restituisca una stringa che fornisca una descrizione della cantina, compresi i dati di tutte le uve prodotte.

Esercizio 5 (7 punti)

Si scriva un'applicazione per la cantina sociale di Santo Stefano al Monte che:

1. Crei una lista di oggetti `Cantina`.
2. Crei un oggetto `Cantina`, lette da tastiera le informazioni necessarie.
3. Inserisca l'oggetto di cui al punto 2. in coda alla lista di cui al punto 1.
4. Lette da tastiera le informazioni su una nuova `Uva`, inserisca tale uva tra quelle prodotte dalla cantina di cui al punto 2., indicando se l'inserimento è avvenuto con successo o meno.
5. Letto da tastiera il nome di un uvaggio, stampi il nome della cantina che produce tale uvaggio al minor prezzo al chilo.

Per la lettura di dati da tastiera è possibile utilizzare l'oggetto `Lettore.in`, definito all'interno del package `fiji.io`, che possiede i seguenti metodi:

- `boolean leggiBoolean()` Legge un boolean (delimitato da spazi).
- `char leggiChar()` Legge un singolo carattere.
- `double leggiDouble()` Legge un numero razionale (delimitato da spazi).
- `float leggiFloat()` Legge un numero razionale (delimitato da spazi).
- `int leggiInt()` Legge un intero (delimitato da spazi).
- `String leggiLinea()` Legge una linea di testo.
- `String leggiString()` Legge una parola senza spazi al suo interno.

Soluzione Esercizio 2

Domanda 1:

2 assegnamenti	2
`i<M`	2 N + 2
`sum+=V[i++]`	2 N + 1
Totale	4 N + 5

Domanda 2:

2 assegnamenti	2
`(j<N/2)`	N/2 + 3/2
`sum+=f(V, j, j)`	N/2 + 1/2
complessità di `f`	$N^2/2 + 5N/2 + 2$
`j++;`	N/2 + 1/2
Totale	$N^2/2 + 4N + 13/2$

Domanda 3:
Complessità asintotica: $O(N^2)$

Soluzione Esercizio 3

```
class Uva implements Comparable<Uva> {
  private String tipo;
  private float peso, prezzo;

  public Uva(String tipo, float peso, float prezzo) {
    this.tipo=tipo;
    this.peso=peso;
    this.prezzo=prezzo;
  }

  public String getTipo() { return tipo; }
  public float getPeso() { return peso; }
  public float getPrezzo() { return prezzo; }
  public float costo() { return peso*prezzo*100; }

  public String toString() {
    return tipo + "(" + peso + "): " + prezzo;
  }

  public boolean equals(Object o) { return equals((Uva) o); }
  public boolean equals(Uva u) { return (tipo.equals(u.tipo)); }

  public int compareTo(Uva u) {
    float diff=costo()-u.costo();
    if(diff==0) return 0;
    if(diff<0) return -1;
    return 1;
  }
}
```

Soluzione Esercizio 4

```
import java.util.*;
class Cantina {
  private String nome, indirizzo;
  private Set<Uva> uve;

  public Cantina(String nome, String indirizzo) {
    this.nome=nome; this.indirizzo=indirizzo;
    uve=new HashSet<Uva>();
  }

  public String getNome() { return nome; }

  public boolean aggiungi(Uva u) { return uve.add(u); }
  public float cerca(String uvaggio) {
    for(Uva u:uve) if(u.getTipo().equals(uvaggio)) return u.getPrezzo();
    return -1;
  }

  public float valoreTotale() {
    float v=0;
    for(Uva u:uve) v+=u.costo();
    return v;
  }

  public String toString() {
    return nome + ", " + indirizzo + "\n" + uve.toString();
  }
}
```

Soluzione Esercizio 5

```
import java.util.*;
import fiji.io.*;
class Applicazione {
  public static void main(String[] args) {
    List<Cantina> l=new ArrayList<Cantina>(); // domanda 1
    Cantina c=new Cantina(Lettore.in.leggiLinea(),
      Lettore.in.leggiLinea()); // domanda 2
    l.add(c); // domanda 3
    if(!c.aggiungi(new Uva(Lettore.in.leggiLinea(),
      Lettore.in.leggiFloat(), Lettore.in.leggiFloat()))
        System.out.println("uva già presente") // domanda 4
  String nome=Lettore.in.leggiLinea();
    Cantina min=null;
    float pmin=0;
    for(Cantina can: l) {
      float p=can.cerca(nome);
      if((p>0)&&((min==null)||(p<pmin))) {
        min=can; pmin=p;
      }
    }
    System.out.println(min.getNome()); // domanda 5
  }
}
```