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

Appello del 17/6/2009

Esercizio 1 (4 punti)

Modalità di creazione di oggetti 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, sum=0;
  while(i<N)
    sum+=V[++i];
  return sum;
}

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

1. Calcolare la complessità in passi base della funzione `f` nei termini dei parametri `M` ed `N` (suggerimento: si distinguano i casi in cui `M` assume valori minori di `N` da quelli in cui assume valori maggiori o uguali a `N`).
2. Calcolare la complessità in passi base della funzione `g` nei termini dei parametri `M` ed `N` (suggerimento: si supponga `N>M`).
3. Calcolare la complessità asintotica della funzione `g` nei termini del parametro `M`.

Esercizio 3 (6 punti)

Il rettore dell'università "La Sorbolona" ha deciso di informatizzare la gestione dei libri contenuti all'interno della biblioteca centrale dell'ateneo. In particolare, per ogni libro in inventario occorre memorizzare l'autore, il titolo, la casa editrice e l'anno di pubblicazione. Si scriva una classe `Libro` per la biblioteca centrale de "La Sorbolona" che:

1. Possieda un opportuno costruttore con parametri.
2. Presenti opportuni metodi che permettano di accedere alle variabili di istanza dell'oggetto.
3. Presenti il metodo `toString` che fornisca la descrizione del libro.
4. Possieda il metodo `equals` per stabilire l'uguaglianza con un altro oggetto `Libro` (l'uguaglianza va verificata sul titolo e sull'autore).
5. Implementi l'interfaccia `Comparable`, definendo il metodo `compareTo` per stabilire la precedenza con un `Libro` passato come parametro (per autore ed anno di pubblicazione crescenti).

Esercizio 4 (8 punti)

Si scriva una classe `Sezione` che memorizzi le informazioni relative ai libri contenuti all'interno di ogni sezione della biblioteca. In particolare, oltre al codice della sezione ed al piano dell'edificio in cui tale sezione si trova, occorre memorizzare i libri contenuti nella sezione all'interno di una lista. La classe `Sezione` deve inoltre:

1. Presentare un opportuno costruttore (inizialmente la sezione non contiene alcun libro).
2. Possedere un metodo `aggiungi` che, dato un oggetto `Libro`, lo inserisca all'interno della lista mantenendola ordinata secondo il punto 5. dell'esercizio 4 (suggerimento: si utilizzi il metodo `add(int i, Libro l)` della classe `List<Libro>`).
3. Presentare un metodo `cerca` che, dato il nome di un autore, restituisca un insieme contenente tutti i libri di tale autore all'interno della sezione.
4. Possedere un metodo `cancella` che, dato il nome di un autore, rimuova dalla lista tutti i libri di tale autore.
5. Possedere il metodo `toString` che restituisca una stringa che fornisca una descrizione della sezione, compresi tutti i libri che ivi si trovano.

Esercizio 5 (7 punti)

Si scriva un'applicazione per la biblioteca centrale de "La Sorbolona" che:

1. Crei un vettore di oggetti `Sezione` (la cui dimensione deve essere richiesta all'utente).
2. Crei un oggetto `Libro`, lette da tastiera le informazioni necessarie.
3. Trovi la prima sezione all'interno del vettore di cui al punto 1. che non contiene alcun libro dell'autore del libro di cui al punto 2.
4. Inserisca il libro di cui al punto 2. nella sezione di cui al punto 3. (se tale sezione esiste).
5. Letto da tastiera il nome di un autore (di dubbia credibilità scientifica), provveda a rimuovere i libri di tale autore da tutte le sezioni della biblioteca.

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	
`while(i<N)`	`N-M`+1	o 1
`sum+=V[++i]`	`N-M`	o 0
Totale	2 `N` – 2 `M` + 3	o 3

Domanda 2:

2 assegnamenti	2
`(j<N)`	`N` + 1
`sum+=f(V, j, M)`	`N`
complessità di `f`	$M^2 + M + 3N$
`j++`	`N`
Totale	$M^2 + M + 6N + 3$

Domanda 3:

Complessità asintotica: $O(M^2)$

Soluzione Esercizio 3

```
class Libro implements Comparable<Libro> {
  private String autore, titolo, editore;
  private int anno;

  public Libro(String autore, String titolo, String editore, int anno) {
    this.autore=autore;
    this.titolo=titolo;
    this.editore=editore;
    this.anno=anno;
  }

  public String getAutore() { return autore; }
  public String getTitolo() { return titolo; }
  public String getEditore() { return editore; }
  public int getAnno() { return anno; }

  public String toString() {
    return autore + ": " + titolo + "(" + editore + "); " + anno;
  }

  public boolean equals(Object o) { return equals((Libro) o); }
  public boolean equals(Libro l) {
    return ((autore.equals(l.autore))&&(titolo.equals(l.titolo)));
  }

  public int compareTo(Libro l) {
    int ret=this.autore.compareTo(l.autore);
    if(ret==0) ret=this.anno-l.anno;
    return ret;
  }
}
```

Soluzione Esercizio 4

```
import java.util.*;
class Sezione {
  private int piano, codice;
  private List<Libro> libri;

  public Sezione(int piano, int codice) {
    this.piano=piano;
    this.codice=codice;
    libri = new ArrayList<Libro>();
  }

  public void aggiungi(Libro l) {
    int i=0;
    while((i<libri.size())&&(libri.get(i).compareTo(l)<0)) i++;
    libri.add(i,l);
  }

  public Set<Libro> cerca(String autore) {
    Set<Libro> s=new HashSet<Libro>();
    for(Libro l:libri)
      if(l.getAutore().equals(autore)) s.add(l);
    return s;
  }

  public void cancella(String autore) {
    for(Libro l:libri)
      if(l.getAutore().equals(autore)) libri.remove(l);
  }

  public String toString() {
    return codice + " (" + piano + ")\n" + libri.toString();
  }
}
```

Soluzione Esercizio 5

```
import fiji.io.*;
import java.util.*;
class Applicazione {
  public static void main(String[] args) {
    Sezione v[] = new Sezione[Lettore.in.leggiInt()]; // domanda 1
    Libro l= new Libro(Lettore.in.leggiLinea(), Lettore.in.leggiLinea(),
      Lettore.in.leggiLinea(), Lettore.in.leggiInt()); // domanda 2
    int i=0;

    while((i<v.length)&&(!v[i].cerca(l.getAutore()).isEmpty()))
      i++; // domanda 3
    if(i<v.length) v[i].aggiungi(l); // domanda 4
    String autore=Lettore.in.leggiLinea();
    for(i=0; i<v.length; i++)
      v[i].cancella(autore); // domanda 5
  }
}
```