Metodologie di Programmazione - Canale P-Z
A.A. 2008/09 - Prova intermedia del 22/04/2009

Avvertenze Le soluzioni devono essere scritte in modo chiaro e ordinato. Il codice deve essere indentato per facilitarne la lettura. Si possono consultare libri e appunti ma non è permesso usare computer.

Ogni esercizio o parte di esercizio ha un punteggio massimo. Il voto (in trentesimi) sarà determinato dalla somma dei punteggi ottenuti nei vari esercizi. (Notare che la somma dei punti degli esercizi proposti supera 30. Quindi, per ottenere il massimo non è necessario svolgere e ottenere il massimo dei punti in tutti gli esercizi.)

Esercizio 1 [4 punti]

Definire una classe `Corso` che contiene le informazioni di base relative ad un corso universitario. Le informazioni che ogni istanza della classe deve mantenere sono: il nome, un codice numerico identificativo (un valore intero non negativo), gli obiettivi formativi, un programma sintetico. **[2 punti]**

Il codice e il nome del corso sono obbligatori e immutabili, ovvero, ogni istanza deve contenere un valore per questi campi e il loro valore non può essere cambiato dopo l'inizializzazione. Gli altri campi non sono richiesti al momento della creazione dell'istanza e possono essere aggiornati successivamente.

Tutti i campi della classe non devono essere accessibili direttamente, ma tramite opportuni getters e, se modificabili, setters.

Riscrivere il metodo `toString` in modo che restituisca una stringa contenente tutti i dati di un corso. **[2 punti]**

Esercizio 2 [16 punti]

Definire una gerarchia di classi per memorizzare i dati relativi agli studenti e ai docenti dei corsi.

Per prima cosa si definisca una classe `Persona` che contiene i seguenti dati: nome, cognome, ed email. Ad eccezione dell'email, tutti i dati della classe `Persona` sono obbligatori e immutabili. Tutti i campi sono privati e accessibili solo attraverso opportuni getters/setters. Riscrivere il metodo `toString` in modo che restituisca una stringa contenente tutti i dati di una persona. **[2 punti]**

Si definiscano poi due sottoclassi `Studente` e `Docente` che contengono le informazioni specifiche dei corrispondenti ruoli.

La classe `Studente` deve contenere la matricola dello studente (valore obbligatorio e immutabile) e una lista (ad esempio, un `ArrayList`) contenente i codici dei corsi frequentati dallo studente. I corsi frequentati non devono necessariamente essere inseriti alla creazione di ogni istanza di `Studente`, ma accessibili e modificabili mediante i metodi richiesti di seguito. **[1 punti]**

Scrivere un metodo `freqCorso` che, ricevuto come parametro un codice, verifica se uno studente frequenta il corrispondente corso. **[2 punti]**

Scrivere un metodo `addCorso` che, dopo aver verificato che un codice di corso non è presente nella lista dei corsi frequentati dallo studente, lo aggiunge a tale lista. **[3 punti]**

Scrivere un metodo `delCorso` che, se presente, elimina codice di corso dalla lista dei corsi frequentati da uno studente. **[3 punti]**

I metodi `addCorso` e `delCorso` devono restituire `true` se la corrispondente operazione ha successo e `false` in caso contrario.

La classe `Docente` deve contenere un codice numerico identificativo del docente (obbligatorio e immutabile) e un array (attenzione, non una lista) contenente i codici dei corsi insegnati dal docente. I corsi non devono necessariamente essere inseriti alla creazione di ogni istanza di `Docente` ma accessibili e modificabili in blocco, ovvero, oltre ai getters per leggere i valori dei campi, si deve prevedere un metodo che ricevuto un array di codici di corsi lo assegna ad un docente. **[1 punti]**

Si scriva un metodo `insegnaCorso` che ricevuto come parametro un codice, verifica se un docente insegna il corrispondente corso. **[2 punti]**

Nelle sottoclassi di `Persona`, sovrascrivere il metodo `toString` in modo che aggiunga i dati specifici della sottoclasse alla stringa prodotta dalla superclasse. **[2 punti]**

Esercizio 3 [4 punti]

Data una istanza di `ArrayList<Persona>` e un codice di corso, scrivere una funzione che costruisce la lista delle email degli studenti che frequentano il corso. **[4 punti]**

Esercizio 4 [12 punti]

Supponiamo che per memorizzare delle liste di studenti su file di tipo testo si utilizzi un formato che prevede di scrivere i dati di ogni studente su di una riga rispettando il seguente pattern:

matricola : *cognome* : *nome* : *email*

Ad esempio, la riga corrispondente allo studente Carlo Maria Rossi, con matricola 23136 ed email `cmrossi@gigimail.com` è

```
23136:Rossi:Carlo Maria:cmrossi@gigimail.com
```

1. Scrivere una funzione che prende come parametro di input il nome di un file (una stringa) contenente una lista di studenti nel formato precedentemente descritto e, usando le classi `FileReader` e `Scanner`, crea una istanza di una classe che implementa l'interfaccia `List` (ad esempio `ArrayList` o `LinkedList`) con la lista degli studenti. **[4 punti]**

   La funzione deve catturare e gestire le eventuali eccezioni sollevate dai metodi usati per la lettura. **[2 punti]**

2. Scrivere una funzione che, ricevuto come input una istanza di `ArrayList<Studenti>` e una stringa `str`, utilizza la classe `PrintWriter` per scrivere su di un file di nome `str` (sovrascrivendo un eventuale file con lo stesso nome già esistente) contenente la lista degli studenti nel formato precedentemente descritto. **[4 punti]**

   La funzione deve propagare le eventuali eccezioni sollevate dai metodi usati per la scrittura, garantendo comunque (usando un blocco `try-finally`) la corretta chiusura dei file aperto in scrittura. **[2 punti]**

Esercizio 5 [4 punti]

Supponiamo di avere le seguenti classi, di cui si riportano solo i campi e i costruttori.

```
public class Punto2 {
  private int x;
  private int y;
  public Punto2(int x, int y) {
   this.x = x;
   this.y = y;
  }
  ...
}

public class Punto3 extends Punto2 {
  private int z;
  public Punto3(int z) {
    this.z = z;
  }
  public Punto3(int x, inty y, int z) {
    this.z = z;
    super(x, y);
  }
  ...
}
```

Quali sono i problemi nella definizione dei costruttori? **[2 punti]**

Indicare una possibile soluzione. **[2 punti]**