INFORMATICA E CALCOLATORI (ELETTRONICA) **Scritto 13 luglio 2004**

INFORMATICA (BIOMEDICA)

Un sistema semplificato per la gestione degli studenti di un Corso di Laurea può essere realizzato con le strutture seguenti:

```
struct studente
{    char cognome [LUNG]; int matricola; int anni;
     int numEsami; float media;
};

struct CorsoLaurea
{      studenti stu[NUM]; int numStu;
};
```

con **NUM** il numero massimo di studenti gestibili con il sistema di gestione e **numStu** il numero di studenti effettivamente iscritti al Corso di Laurea. Ogni studente è individuato dal suo cognome (al massimo di **LUNG** caratteri), dalla matricola, dal numero di anni da cui è iscritto al Corso, dal numero di esami sostenuti e dalla media dei voti con cui ha superato gli esami sostenuti. Quando un'istanza della struttura **CorsoLaurea** è creata, il valore di **numStu** è inizializzato a zero.
Scrivere il corpo delle seguenti funzioni C++.

1. **void build(const char nome[], CorsoLaurea* c)** che, apre in lettura il file il cui nome è passato come primo argomento alla funzione, e fino a raggiungere la fine del file ripete le azioni seguenti: a) legge una stringa, tre numeri interi ed un numero reale che rappresentano, rispettivamente, il cognome di uno studente, il suo numero di matricola, gli anni da cui è iscritto al Corso, il numero di esami sostenuti e la media dei voti riportati, e b) li inserisce nel sistema di gestione.
2. **void ordina(CorsoLaurea* c)** che ordina gli studenti per numero crescente di anni da cui sono iscritti al Corso ed all'interno di ogni anno per valori alfabeticamente crescenti del campo **cognome**;
3. **bool laureato(CorsoLaurea* c, int matricola)** che elimina dal sistema di gestione lo studente la cui matricola è passata come secondo argomento alla funzione e ricompatta il vettore **stu** in modo tale che il vettore rimanga ordinato come descritto nella funzione precedente; la funzione restituisce **false** se non è presente nessun studente con quel numero di matricola, altrimenti restituisce **true**;
4. **int numStu(const CorsoLaurea* c, int numAnni, float m)** che restituisce il numero di studenti che sono iscritti da un numero di anni pari al secondo argomento passato alla funzione ed hanno conseguito una media superiore al valore passato come terzo argomento alla funzione.
5. **void rendimento(const CorsoLaurea* c)** stampa, ordinandoli per valori alfabeticamente crescenti del campo **cognome**, il cognome di ogni studente ed il suo rendimento. Il rendimento è calcolato come la media tra il rendimento RA ed il rendimento RV. Il rendimento RA è definito come il rapporto tra il numero di esami sostenuti dallo studente ed il numero totale di esami che avrebbe dovuto sostenere negli anni in cui è stato iscritto (si supponga che in ogni anno, anche quelli fuori corso, ci siano 6 esami). Il rendimento RV corrisponde al rapporto tra la media dei voti ed il voto massimo (nel nostro caso 30).

NOTE SULLO SVOLGIMENTO DELLA PROVA:

- PER SVOLGERE L'ELABORATO, APRIRE il Dev-C++ (dal Menù **`Avvio`** (o **`Start`**) nella barra degli strumenti in fondo allo schermo, selezionare Programmi e quindi Dev-C++);
- PRIMA DI INIZIARE LO SVOLGIMENTO DELL'ELABORATO, selezionare la voce **`Identifica studente`** nel menù **`Strumenti`** all'interno dell'ambiente Dev-C++ e inserire i dati richiesti;
- Per svolgere l'elaborato, aprire il progetto *esaInf.dev* presente nel directory *c:\esame\esaInf* e scrivere le funzioni richieste nel file *compito.cpp*, già presente nel progetto;
- per una corretta stampa dell'elaborato bisogna mantenere il codice entro i margini imposti dall'ambiente Dev-C++ (linea verticale presente alla destra della pagina);
- SE L'ELEBAORATO È STATO COMPILATO SENZA ERRORI, PRIMA DELLA CONSEGNA, selezionare la voce **`Consegna Compito`** nel menù **`Strumenti`** all'interno dell'ambiente Dev-C++ e premere il tasto INVIO fino a quando non sparisce la finestra che è stata attivata.