Informatica Generale II

A.A. 2003/2004

Esame: 3 maggio 2004

Si vuole informatizzare la gestione di una piccola biblioteca. Si sa a priori che la biblioteca dovrà gestire al massimo 50 utenti che possono prendere libri in prestito. Il programma di gestione dovrà tenere l'archivio dei libri posseduti e l'archivio dei prestiti.
L'archivio dei libri posseduti è memorizzato in una lista concatenata `archivio`. Gli elementi della lista sono dei record di tipo `DatiVolume`, ognuno dei quali contiene i seguenti campi (il candidato noti che nel programma si assume che questi campi abbiano i nomi come riportati sotto):

- `autore` (max 30 caratteri);
- `titolo` (max 50 caratteri);
- `prezzo` (intero);
- `num` (intero) numero inventario;
- `prestito` (booleano).

I 50 utenti sono identificati con un numero N tale che $1 \leq N \leq 50$, ed ogni utente può avere in prestito più libri.
L'archivio dei prestiti è memorizzato in un vettore di liste (memorizzato in una variabile `prestiti` nel programma principale). Il componente `prestiti[i-1]` è il puntatore alla lista di libri attualmente in prestito all'utente i-esimo. La lista dei libri prestati ad un utente contiene solo il numero di inventario, che identifica il volume in modo univoco.
Al candidato è richiesto di:

1. **[Punti 3]** definire le seguenti strutture dati:
   - `DatiVolume` per memorizzare i campi (autore, titolo, ...);
   - `ListaVolumi` per memorizzare l'archivio dei libri della biblioteca. La definizione dovra' provvedere il costruttore base senza argomenti e il costruttore con due argomenti, il primo corrispondente al volume, il secondo puntatore al volume successivo (si veda funzione `LeggiArchivio` nel file `main.cpp`);
   - `ListaNumeriInventario`,, per memorizzare i numeri di inventario dei libri in prestito. La definizione dovra' provvedere il costruttore base senza argomenti e il costruttore con due argomenti, il primo corrispondente al numero di inventario, il secondo puntatore all'elemento successivo (potrebbe essere utile nella funzione `AggiornaArchivio` richiesta nel seguito);
2. **[Punti 4]** scrivere la dichiarazione e la definizione di una funzione `ListaVolumiLength` che prenda in input la lista rappresentante l'archivio dei libri (`ListaVolumi * archivio`) e restituisca il numero di volumi (ovvero la lunghezza della lista).
3. **[Punti 5]** scrivere la dichiarazione e la definizione di una funzione `InArchivio` che:
   - riceve in input la lista rappresentante l'archivio dei libri (`ListaVolumi * archivio`) e un numero di inventario (`int num`);
   - restituisca:
     - il valore `true` se il libro corrispondente al numero di inventario è presente nell'archivio e non è in prestito;
     - il valore `false` se il libro non è presente in archivio o se è in prestito.
4. **[Punti 8]** scrivere la dichiarazione e la definizione di una funzione `AggiornaArchivio` che:

- riceve in input la lista rappresentante l'archivio dei libri (`ListaVolumi * archivio`), il vettore dei prestiti (`ListaNumeriInventario * prestiti[50]`), un numero di inventario (`int num`), ed un numero utente (`int utente`);
- usa la funzione `InArchivio` definita al punto 2 per verificare se il volume può essere preso in prestito;
- aggiorna l'archivio dei prestiti dell'utente (`prestito[utente - 1]`), la lista rappresentante l'archivio dei libri della biblioteca (`archivio`) per tenere conto del nuovo prestito nel caso in cui esso sia possibile e stampi il messaggio

  "`prestito non possibile per il libro #`"

  Dove # è il numero di inventario del libro che si richiede in prestito.

Ai candidati viene fornito un file `main.cpp` contenente il main e alcune funzioni di utilità per creare l'archivio e per liberare la memoria dell'archivio e dei prestiti. Nelle liste si assume che il campo che punta al nodo successivo si chiami `next`. I candidati devono inserire i propri dati negli appositi campi e procedere a inserire il codice relativo ai punti di cui sopra nei predisposti punti all'interno del file `main.cpp`. Di seguito è riportato un frammento del contenuto del file `main.cpp` relativo alle parti da completare. Nella directory di lavoro si trovera' anche un file `archivio.dat` da cui il programma legge l'archivio iniziale mediante le funzioni gia' fornite nel file `main.cpp` e che quindi non devono essere re-implementate dal candidato.

```
/*
  NOME:
  CONGNOME:
  MATRICOLA:
  CODICE CALCOLATORE:
*/

#include <iostream>
#include <cstdlib>
#include <cstring>
#include <fstream>

using namespace std;

/* PUNTO 1 */

/* definizione del tipo DatiVolume */

/* definizione del tipo ListaVolumi */
/* Nota: il campo che punta al nodo successivo si dovra' chiamare next */

/* definizione del tipo ListaNumeriInventario */
/* Nota: il campo che punta al nodo successivo si dovra' chiamare next */


/* PUNTO 2 */
/* dichiarazione funzione ListaVolumiLength */

/* definizione funzione ListaVolumiLength */


/* PUNTO 3 */
/* dichiarazione funzione InArchivio */

/* definizione funzione InArchivio */


/* PUNTO 4 */
/* dichiarazione funzione AggiornaArchivio */

/* definizione funzione AggiornaArchivio */
```