Un sistema per la gestione biglietti aerei può essere rappresentato attraverso le seguenti costanti e strutture:

```
const int MAXNUM = 100;
const int MAXL = 50;

enum Classe {PRIMA, BUSINESS, ECONOMICA};
enum Sesso {M, F};

struct Passeggero {
  char nome[MAXL];
  char cognome[MAXL];
  Sesso sex;
};

struct Biglietto {
  Passeggero pass;
  Classe cl;
  int volo;
  int pos;
};

struct GestioneBigl {
  Biglietto big[MAXNUM];
  int numBigl;
};
```

con **MAXNUM** numero massimo di biglietti che possono essere gestiti e **numBigl** il numero di biglietti effettivamente presenti nel sistema. Ogni biglietto è caratterizzato da: un campo **pass** che contiene i dati del passeggero, un campo **cl** (di tipo **Classe**) che rappresenta il tipo di biglietto, un intero **volo** che indica il numero del volo, un intero **pos** che indica la posizione del passeggero all'interno dell'aeromobile. Il tipo **Passeggero** si compone di due campi **nome** e **cognome** di tipo stringa (di lunghezza massima **MAXL**), e di un campo **sex** (di tipo **Sesso**) che codifica il sesso del passeggero.

Scrivere il corpo delle seguenti funzioni C++.

1. **int quanti(const GestioneBigl* gb, Sesso s)** che restituisce il numero di biglietti presenti relativi a passeggeri aventi il sesso indicato da **s**.
2. **int perClasse(const GestioneBigl* gb, Classe c, Biglietto vet[])** che copia in **vet** i dati dei biglietti presenti nel sistema che hanno la classe specificata da **c**; la funzione restituisce il numero di oggetti **Biglietto** inseriti in **vet**.
3. **bool presente(const GestioneBigl* gb, char n[], char c[], int* pv, int* pp)** che cerca all'interno del sistema la presenza di un biglietto relativo ad un passeggero avente il nome ed il cognome indicati da **n** e **c** rispettivamente; se un biglietto relativo a tale passeggero

è presente, la funzione comunica al chiamante, attraverso **`pv`** e **`pp`**, l'indicazione del numero di volo e della posizione (rispettivamente) e restituisce il valore *true*; altrimenti restituisce il valore *false*.

4. **`void salvaOrd(const GestioneBigl* gb, const char nf[], int v)`** che salva nel file il cui nome è specificato da **`nf`**, i campi **`cognome`** e **`nome`** dei passeggeri aventi un biglietto relativo al volo specificato da **`v`**, ordinandoli alfabeticamente per cognome.
5. **`int elimina(GestioneBigl* gb, int v)`** che elimina dal sistema tutti i biglietti relativi al volo indicato da **`v`**, lasciando compatto il vettore **`big`**; la funzione restituisce il numero di biglietti eliminati.

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'ELABORATO 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.