Informatica Generale II

A.A. 2005/2006

Esame: 8 settembre 2006

Un sistema di prenotazioni

Ogni anno il gestore di un albergo ha il problema di organizzare le prenotazioni delle stanze. In primo luogo vorrebbe evitare la sovrapposizione delle prenotazioni delle stanze, che in passato ha creato non pochi disagi ai sui clienti. Poi vorrebbe privilegiare l'utilizzo delle stanze situate ai piani più bassi, così da evitare dover climatizzare tutto l'albergo.

Stufo di gestire tutto manualmente, il gestore si rivolge al candidato perché sviluppi un programma che gli consenta di automatizzare l'allocazione delle stanze in base alle prenotazioni che gli giungono dai clienti.

Implementazione

L'albergo ha un certo numero di stanze tutte uguali. Le stanze sono distribuite su più piani, e tutti i piani contengono un numero uguale di stanze.

Ciascuna stanza ha un nome univoco costituito da un numero intero a due cifre, in cui la cifra delle decine rappresenta il numero del piano su cui la stanza è situata, e la cifra delle unità rappresenta il numero della stanza a quel piano. In questa denominazione si assume un massimo di 9 piani e 9 stanze per piano.

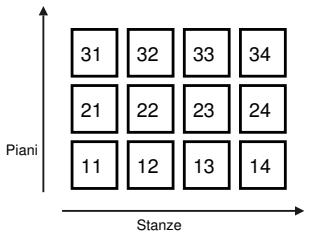

Figura 1: Esempio di albergo con 3 piani e 4 stanze per piano

Ogni *prenotazione* è caratterizzata da dati quali il nome e il cognome della persona che ha fatto la prenotazione, la data di inizio del soggiorno, e la durata del soggiorno stesso. Per esempio, una prenotazione potrebbe contenere i dati di Albert Einstein che ha prenotato una stanza per un soggiorno di 10 giorni a partire dal 10 settembre 2006.

Per semplicità, le date non verranno rappresentate con giorno, mese ed anno, ma con il numero di giorni trascorsi a partire da una data convenzionale, in modo da evitare calcoli altrimenti troppo complessi. La data convenzionale è il 1/1/2006. Quindi il valore 0 corrisponde al 1 gennaio 2006, 1 al 02/01/06, 30 al 31/01/06, 31 al 01/02/06, il 35 corrisponde al 05/02/06, 450 al 27/3/2007, ecc.

Ciascuna stanza è rappresentata da una lista concatenata semplice di prenotazioni. La lista corrisponde alle prenotazioni fatte per la stanza che rappresenta. La lista è ordinata in ordine *crescente* per data di prenotazione. Ad esempio la stanza 13 potrebbe avere una lista di prenotazioni come segue:

Nome	Cognome	Data	Giorni	(Periodo)
Paolino	Paperino	50	5	(20/2/2006 - 24/2/2006)
Fam.	Basettoni	58	2	(28/2/2006 - 1/3/2006)
Pippo	Depippis	180	7	(30/6/2006 - 6/7/2006)

Il periodo qui è riportato solo per maggiore leggibilità e chiarezza, e *non* è memorizzato nelle prenotazioni.

Come si vede la lista è ordinata in senso crescente per data. Inoltre la lista non deve mai contenere sovrapposizioni nelle prenotazioni. Ad esempio, la data di inizio soggiorno di "Basettonì' non potrebbe essere 53 o 49, perché sarebbe in sovrapposizione con la prenotazione di "Paperinò'.

L'insieme delle stanze dell'albergo è rappresentato mediante una matrice $P \times S$ dove P è il numero dei piani, e S è il numero di stanze per piano. Quindi ad esempio la posizione *(0,0)* della matrice corrisponde alla stanza 11, la posizione *(0,1)* alla stanza 12, la posizione *(2,4)* alla stanza 35 (terzo piano, stanza 5). In ciascuna delle celle della matrice, è memorizzata la lista delle prenotazioni per la corrispondente stanza. In sintesi, l'albergo è rappresentato mediante una matrice di liste concatenate di prenotazioni.

Quesiti

1. **(Punti: 2)** Si implementino le definizioni dei tipi che servono a rappresentare, tramite **strutture** le seguenti entità. I nomi dei campi e delle strutture devono essere rispettati rigorosamente.

   Prenotazione Che contenga una singola prenotazione, contenente i seguenti campi:

   nome (puntatore a carattere)

   cognome (puntatore a carattere)

   data Data di inizio del soggiorno relativa al 1/1/2006 (intero)

   giorni Numero di giorni della durata del soggiorno

   Si implementino anche uno o più costruttori.

   ListaPrenotazioni Lista concatenata semplice di *Prenotazioni*. I campi sono *dato* e *next*. Si implementino anche uno o più costruttori.

2. **(Punti: 5)** Si scrivano le funzioni:

   - *Prenotazione_stampa* che stampi una prenotazione nel formato

     ```
     <nome> <cognome>: da <data> per <giorni> giorni
     ```

   - *ListaPrenotazioni_stampa* che stampi una lista concatentata di prenotazioni. Il formato è una prenotazione per riga. Se la lista è vuota, non deve essere stampato nulla.
   - *Albergo_stampa* che prenda in input la matrice delle stanze e stampi ogni stanza con la sua lista delle prenotazioni. Il formato è il seguente:

     ```
     Stanza <p><s>:
     <lista prenotazioni>
     -------------------------
     ```

Dove *p* e *s* sono ripettivamente il piano e il numero della stanza su quel piano (valori a partire da 1).

Suggerimento: Le costanti *N_PIANI* e *N_STANZE* in `main.cpp` definiscono il numero di piani e di stanze per piano.

3. **(Punti: 8)** Si implementi la funzione *ListaPrenotazioni_inserisci_prenotazione* che:

   - Prenda in input una *ListaPrenotazioni* e una *Prenotazione*;
   - Restituisca un valore booleano.

   La funzione deve inserire la prenotazione nella lista in ordine di data di inizio soggiorno, ma solamente se la prenotazione non si sovrappone con quelle già esistenti nella lista.

   Se la prenotazione viene inserita nella lista la funzione deve restituire *true*, altrimenti deve restituire *false*.

   Suggerimento: Si sfruttino gli schemi già visti per l'inserimento in ordine.

   Suggerimento: Al momento di verificare la sovrappoizione, si tenga presente che la lista è ordinata.

4. **(Punti: 5)** Si scriva la funzione *Albergo_prenota_stanza* che dati la matrice delle stanze, e una prenotazione, prenoti una stanza dell'albergo automaticamente.

   La funzione deve:

   (a) Scandire la matrice a partire dai piani più bassi.

   (b) Effettuare la prenotazione mediante la funzione *ListaPrenotazioni_inserisci_prenotazione*

   (c) Se la prenotazione è avvenuta, uscire restituendo *true*

   (d) Altrimenti continuare per le altre stanze

   (e) Se alla fine tutte le stanze risultano non utilizzabili per la prenotazione, restituire *false*

Note importanti

1. Nella directory di lavoro $\sim$`/Esame` vi è il file `main.cpp` . che il candidato dovrà completare nelle sue parti mancanti e con i dati personali.
2. La stessa directory contiene inoltre il file di testo `prenotazioni` che la funzione *main* utilizza per caricare le prenotazioni nell'albergo. Se modificato, il file deve mantenere il formato del ile originale, secondo il quale ogni prenotazione deve essere costituita da nome, cognome, data e giorni separati da uno o piu spazi.
3. Il file `main.cpp` contiene già del codice che si può sfruttare per la risoluzione dei problemi proposti.
4. Il file `main.cpp` così comè fornito non può essere compilato senza ottenere degli errori di compilazione. È compito dello studente modificarlo affinché possa essere correttamente compilato.
5. Al fine della valutazione verrà preso in considerazione unicamente il file sim`/Esame/main.cpp` . Ogni altro file verrà ignorato. Il candidato dovrà perciò assicurarsi di modificare e salvare unicamente tale file.

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

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

using namespace std;

const int N_PIANI = 2;
const int N_STANZE = 3;
const char* FNAME = "prenotazioni";


/* ---------------------------- INIZIO PUNTO 1  ------------------------- */
/*                     Scrivere qui sotto la soluzione                    */
/*    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv   */

/**********************************************************************
           !!!!!!!!!!!   A T T E N Z I O N E   !!!!!!!!!!!
       Compilare con i propri dati l'intestazione del file *PRIMA* di
       cominciare! (e poi _cancellare_ questo testo)
**********************************************************************/

/*    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^   */
/* ---------------------------- FINE PUNTO 1  ------------------------- */



/* ---------------------------- INIZIO PUNTO 2  ------------------------- */
/*                     Scrivere qui sotto la soluzione                    */
/*    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv   */

/*    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^   */
/* ---------------------------- FINE PUNTO 2 ------------------------- */



/* ---------------------------- INIZIO PUNTO 3  ------------------------- */
/*                     Scrivere qui sotto la soluzione                    */
/*    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv   */

/*    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^   */
/* ---------------------------- FINE PUNTO 3 ------------------------- */



/* ---------------------------- INIZIO PUNTO 4  ------------------------- */
/*                     Scrivere qui sotto la soluzione                    */
/*    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv   */
bool Albergo_prenota_stanza(ListaPrenotazioni* alb[N_PIANI][N_STANZE],
                            const Prenotazione& pren)
{
  /* ... */

  return false;
}

/*    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^   */
/* ---------------------------- FINE PUNTO 4 ------------------------- */



/* Inizializza l'albergo inserendo in tutte le stanze una lista di
   prenotazione vuota. */
void Albergo_init(ListaPrenotazioni* alb[N_PIANI][N_STANZE])
{
  for (int p=0; p < N_PIANI; ++p) {
    for (int s=0; s < N_STANZE; ++s) {
      alb[p][s] = NULL;
    }
  }
```

```
}


/* legge da file una singola prenotazione */
bool leggi_prenotazione(istream& is, Prenotazione &p)
{
  string str;
  is >> str; p.nome = strdup(str.c_str());
  if (is.fail()) return false;

  is >> str; p.cognome = strdup(str.c_str());
  if (is.fail()) return false;

  is >> p.data;
  if (is.fail()) return false;

  is >> p.giorni;
  if (is.fail()) return false;

  return true;
}

/* carica da un file tutte le prenotazioni nell'albergo */
void Albergo_carica_prenotazioni(ListaPrenotazioni* alb[N_PIANI][N_STANZE])
{
  ifstream ifs(FNAME);
  while (!ifs.eof()) {
    Prenotazione p;
    if (!leggi_prenotazione(ifs, p)) return;

    if (! Albergo_prenota_stanza(alb, p)) {
      cout << "La prenotazione:\n   ";
      Prenotazione_stampa(p);
      cout << endl << "non puo' essere soddisfatta.\n\n";
    }
  }
}


// ----------------------------------------------------------------------
// Main del programma
// ----------------------------------------------------------------------

int main()
{
  ListaPrenotazioni* albergo[N_PIANI][N_STANZE];
  Albergo_init(albergo);
  Albergo_carica_prenotazioni(albergo);

  Albergo_stampa(albergo);

  return 0;
}
```