NOTE SULLO SVOLGIMENTO DELLA PROVA:

- Fare partire il Dev-C++ (dal Menù **`Avvio`** (o **`Start`**) nella barra degli strumenti in fondo allo schermo, selezionare Programmi e quindi Dev-C++);
- attraverso il menu `File->Open...` (o `apri` se il Dev-C++ è in italiano) aprire il progetto *APRIQUESTO.dev* presente nel directory *c:\esame\esaInf*;
- scrivere le funzioni richieste nel file *compito.cpp* già presente nel progetto;
- salvare spesso in modo da non perdere il lavoro nel caso in cui il PC abbia un malfunzionamento (per salvare premere il bottone a forma di dischetto del Dev-C++);
- 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).

Campeggio

Un programma per la gestione semplificata di un campeggio può essere realizzato a partire dalle seguenti costanti e dai seguenti tipi:

```
const int MAXS = 100;
const int MAXN = 200;
const int NUMR = 5;
const int NUMC = 8;

enum Alloggio {Camper, Roulotte, Tenda};

struct Data {
  int giorno;
  int mese;
  int anno;
};

struct Cliente {
  char nome[MAXS];
  Data arrivo;
  int persone;
  Alloggio all;
};

struct Piazzola {
  Cliente cli;
  bool occupata;
};

struct Listino {
  float prezzoPersona;
  float prezzoRoulotte;
  float prezzoCamper;
  float prezzoTenda;
};

struct Campeggio {
```

```
  Piazzola piaz[NUMR][NUMC];
  Listino list;
};
```

Il significato dei tipi `Alloggio` e `Data` è palese. Il tipo `Cliente` memorizza i dati del singolo cliente: il nome (una stringa), il numero di persone facenti capo a quel cliente (un intero), la data di arrivo nel campeggio (tipo `Data`), e il tipo di sistemazione dell'utente (tipo `Alloggio`). Il tipo `Piazzola` contiene i dati della singola piazzola: se è occupata oppure no e, nel caso in cui sia occupata, il cliente che in essa risiede (e le persone a lui associate). Il tipo `Listino` contiene il prezzo giornaliero applicato alla singola persona e per le diverse tipologie di alloggio. Il tipo `Campeggio` memorizza i dati di un campeggio. In particolare, un campeggio è organizzato come una matrice di piazzole (campo `piaz`) composta da `NUMR` righe e `NUMC` colonne; il campo `list` della struttura specifica invece i prezzi applicati da tale campeggio.

Realizzare le seguenti funzioni:

bool inizializza(Campeggio* pc) inizializza il campeggio (tutte le piazzole sono libere) e carica il listino prezzi. I prezzi sono specificati in un file di nome *listino.txt* avente il formato illustrato dal seguente esempio:

```
roulotte 12.0
tenda 10.0
camper 15.5
persona 5.0
```

Nota: le righe che compongono il file possono essere in un ordine qualunque, e non necessariamente nell'ordine specificato dall'esempio precedente.

void stampaMappa(const Campeggio* pc) stampa una mappa del campeggio: le piazzole vuote sono rappresentate da uno spazio, mentre quelle in cui è presente una tenda, una roulotte, o un camper sono rappresentate rispettivamente dai caratteri `T`, `R`, `C`. Esempio:

```
RTT   T C
   RRR CT
T R
      TT
CT  T  T T
```

bool daPiuTempo(const Campeggio* pc, Cliente* cl) trova il cliente che è nel campeggio da più tempo. Il cliente in questione viene restituito al chiamante riempiendo l'oggetto puntato da `cl`. La funzione restituisce *true* se nel campeggio è presente almeno un cliente, *false* altrimenti.

void maggiorCostoGiornaliero(const Campeggio* pc) stampa a video il nome del cliente che ha il maggior costo giornaliero. Per calcolare il costo giornaliero totale del singolo cliente è necessario tenere conto del numero di persone a lui associate e del tipo di alloggio. Il prezzo per persona e i prezzi per le diverse tipologie di alloggio sono specificati nel listino (per esempio, prendendo in considerazione il listino riportato in precedenza, una famiglia di 4 persone che alloggiano in una roulotte spenderanno 32.0).

int colonnaConPiuPersone(const Campeggio* pc) restituisce l'indice della colonna del campeggio in cui sono presenti più persone.