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).

Points of interest

Un programma per la gestione dei punti di interesse (points of interest) di una città può essere realizzato usando le seguenti costanti e i seguenti tipi:

```
const int MAXS = 100;
const int MAXP = 200;

enum Type {CAFE, RESTAURANT, STATION, HOTEL, MUSEUM};

struct Coord {
  double x;
  double y;
};

struct POI {
  Coord coord;
  Type type;
  char name[MAXS];
  int stars;
};

struct Map {
  POI pois[MAXP];
  int numPois;
};
```

Il tipo `Coord` viene usato per usato per rappresentare le coordinate di un punto di interesse all'interno di un piano cartesiano. Le coordinate `x` e `y` hanno valori compresi nell'intervallo [0, 100). La struttura `POI` rappresenta un punto di interesse e ne memorizza il nome (campo `name`), le coordinate (campo `coord`), il tipo (campo `type`), il grado di interesse (campo `stars`, un intero compreso tra 1 e 5 estremi inclusi). La struttura `Map` rappresenta la mappa della città e memorizza i dati dei punti di interesse nell'array `pois;` il campo `numPois` ne indica il numero effettivo.

Scrivere il corpo delle seguenti funzioni C++.

1. **`bool findCloser(const Map* pm, const Coord c, POI* pp)`** riempie l'oggetto `POI` puntato da `pp` con i dati del punto di interesse più vicino al punto le cui coordinate sono indicate da `c`. Il concetto di più vicino deve essere inteso nel senso di distanza euclidea. La funzione restituisce *true* se l'oggetto puntato da `pp` viene riempito, *false* altrimenti (non ci sono punti di interesse nella mappa).

2. **`bool saveByCategory(const Map* pm, const char fn[])`** salva il contenuto della mappa (i punti di interesse in essa contenuti) nel file indicato da `fn`. I dati devono essere raggruppati per categoria e il formato deve essere quello indicato dal seguente esempio:

   Cafe Brazeel 12.4, 18.5 5
   Cafe Ottimelli 66.1, 88.3 4
   Cafe CeccoRivolta 88.8, 77.7 5
   Restaurant Numero_dodici 56.1, 22.3 4
   Station S.Rossore 70.4, 80 4
   Station Centrale 23.2, 11.1 5
   Station Aeroporto 5.3, 3.3 3
   Hotel La_torre 77.4, 99.7 5
   Museum Sinopie 46.6, 89.4 2
   Museum Palazzo_blu 34.5, 56.6 3

   La funzione restituisce *true* se l'operazione va a buon fine, *false* altrimenti.

3. **`bool distances(const Map* pm, const char name1[], const char name2[], double* euc, double* man)`** calcola la distanza euclidea e la distanza manhattan[1] tra i due punti di interesse aventi i nomi indicati da `name1` e `name2`. Si assuma che non ci siano casi di omonimia. La funzione restituisce il valore *true* se le distanze possono essere calcolate, *false* altrimenti (almeno un punto di interesse è assente). La distanza euclidea e la distanza manhattan vengono restituite al chiamante riempiendo gli oggetti puntati da `euc` e `man` rispettivamente.

4. **`int remove(Map* pm, int s)`** rimuove dalla mappa tutti i punti di interesse aventi un campo `stars` con valore minore o uguale a `s`. La funzione restituisce il numero di POI rimossi.

5. **`void draw(const Map* pm)`** stampa a video una rappresentazione della mappa. La mappa è divisa in 10 righe e 10 colonne aventi la stessa dimensione e viene stampata a video come una matrice di 10x10 caratteri. Il carattere '.' indica che in una zona della mappa non ci sono punti di interesse, mentre il carattere 'X' indica che c'è almeno un punto di interesse. La mappa copre la regione del piano [0, 100) x [0, 100). Esempio:

```
X........X
.XX.......
.....X....
..........
..........
...X......
..........
..........
....X..X..
.......X.X
```

1 Si definisce distanza manhattan il cammino che si percorre da un punto ad un altro quando ci si può muovere solo ed esclusivamente con direzioni parallele ad uno degli assi cartesiani. Più formalmente la distanza L_1 tra il punto P_1 di coordinate (x_1,y_1) e il punto P_2 di coordinate (x_2,y_2) è

$$L_1(P_1,P_2) = |x_1 - x_2| + |y_1 - y_2|$$