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 *ProgettoEsame.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).

Inquinamento

Un sistema che gestisce i rilevamenti prodotti da un insieme di centraline per il monitoraggio del livello di inquinamento può essere rappresentato attraverso le seguenti costanti e i seguenti tipi:

```
const int MAXL = 100;
const int MAXS = 100;
const int MAXO = 100;

struct Centralina {
  int id;
  float x;
  float y;
  char nome[MAXL];
};

struct Rilevamento {
  float pm10;
  float pm5;
  int centId;
};

struct SistemaInq {
  Centralina cent[MAXS];
  int numCent;
  Rilevamento ril[MAXO];
  int numRil;
};
```

La struttura `Centralina` memorizza: il nome della centralina (come una stringa), le sue coordinate x e y (due float), e un identificatore univoco (come un intero). La struttura `Rilevamento` contiene i dati rilevati da una centralina. In particolare, i campi `pm10` e `pm5` riportano rispettivamente i valori di PM10 e PM5 registrati, e `centId` contiene l'identificatore univoco della centralina che ha generato il rilevamento. La struttura

`SistemaInq` è dotata dei seguenti campi: un array `cent`, che contiene i dati delle centraline (al più `MAXS`), `numCent` che indica il numero di centraline effettivamente presenti, un array `ril` che contiene i rilevamenti (al più `MAXO`) e `numRil` che indica il numero di rilevamenti effettivamante presenti.

Scrivere il corpo delle seguenti funzioni C++.

1. **`bool carica(SistemaInq* psi, const char nomef[])`** che inserisce nel sistema i dati delle centraline contenuti nel file di nome `nomef`. Il file contiene una riga per ogni centralina e in ogni riga sono contenuti i seguenti dati:

   nome_centralina coordinata_x coordinata_y identificatore

   La funzione restituisce *true* se il caricamento va a buon fine, *false* se si verifica un errore.

2. **`bool inserisciRil(SistemaInq* psi, Rilevamento o)`** che, se possibile, inserisce il rilevamento `o` nel sistema. Affinché il rilevamento `o` possa essere inserito è necessario che: i) l'array `ril` non sia pieno; ii) nel sistema sia già presente una centralina con identificatore pari all'identificatore di centralina di `o`. Se l'inserimento va a buon fine la funzione restituisce *true*, altrimenti restituisce *false*.

3. **`bool maxPm10AndPm5(const SistemaInq* psm, float* ppm10, float* ppm5)`** che trova i valori massimi di PM10 e PM5 tra i rilevamenti contenuti nel sistema. La funzione restuituisce i valori trovati riempiendo gli oggetti puntati da `ppm10` e `ppm5`, da usare rispettivamente per il PM10 e il PM5. La funzione restituisce *true* se nel sistema c'è almeno un rilevamento, *false* altrimenti.

4. **`void eliminaRil(SistemaInq* psi, int sid)`** che elimina dal sistema tutte i rilevamenti che hanno identificatore di centralina pari a `sid` (la funzione provvede a compattare opportunamente il vettore `ril`).

5. **`int piuVicine(const SistemaInq* psm, Centralina s, Centralina vett[])`** che cerca le tre centraline più vicine alla centralina `s`. Il risultato viene lasciato nel vettore `vett`. La funzione restituisce un valore che indica quanti elementi di `vett` sono stati effettivamente riempiti (potrebbero essere meno di tre nel caso in cui il sistema contenga meno di tre centraline).