

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).

Autobus

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

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

enum Tipo {urbana, extraurbana};

struct Coord {
    double x;
    double y;
};

struct Fermata {
    int id;
    char indirizzo[MAXS];
    Coord coord;
    Tipo tipo;
};

struct Linea {
    char nome[MAXS];
    int idferm[MAXF];
    int numIdferm;
};

struct Piano {
    Linea lin[MAXP];
    Fermata fer[MAXF];
    int numLinee;
    int numFermate;
};
```

Il tipo `Coord` memorizza le coordinate delle fermate degli autobus in un piano cartesiano. La struttura `Fermata` contiene i dati relativi ad una fermata dell'autobus: l'id univoco della fermata (un intero positivo), l'indirizzo in cui la fermata si trova (una stringa), le coordinate della fermata, e il tipo di fermata (urbana o extraurbana). Il tipo `Linea` contiene le informazioni relative a una linea autobus: il nome della linea (una stringa), gli id delle fermate in cui l'autobus ferma (array contenente gli id delle fermate), e il numero di fermate relative a tale linea. In particolare il numero massimo di fermate è pari a `MAXF`,

mentre il numero effettivo per la specifica linea è quello indicato da `numIdferm`. Le fermate sono indicate mediante il loro identificatore univoco e l'ordine è quello in cui sono memorizzate nell'array `idferm`. La struttura `Piano` contiene le informazioni relative a un piano trasporti: le linee sono memorizzate nell'array `lin` (al più `MAXP`), mentre le fermate sono memorizzate nell'array `fer` (al più `MAXF`); il numero effettivo di linee e di fermate dello specifico piano trasporti è indicato rispettivamente da `numLinee` e `numFermate`.

Realizzare le seguenti funzioni:

bool caricaPiano(Piano* pp, const char filefermate[], const char filelinee[]) carica da file i dati del piano trasporti. I dati delle fermate sono contenuti nel file `filefermate` mentre quelli delle linee sono contenuti nel file `filelinee`. Il file `filefermate` ha il formato illustrato dal seguente esempio:

```
23 via_mazzini      32.0 44.0 U
44 via_verdi        55.4 21.1 U
55 pza_garibaldi    11.1 44.5 U
98 via_del_bren.     32.4 65.4 E
76 via_rindi        22.1 48.5 U
57 via_aurelia       24.4 56.6 E
88 via_oberdan       34.5 55.5 U
```

ogni riga contiene i dati di una fermata, nell'ordine l'id della fermata (un intero), l'indirizzo (una parola), le coordinate x e y (due reali), il tipo (carattere U per urbana, carattere E per extraurbana).

Il file `filelinee` ha il formato illustrato dal seguente esempio:

```
lam_verde 4 23 44 76 88
lam_rossa 3 55 44 23
lam_blu 5 98 76 23 44 57
```

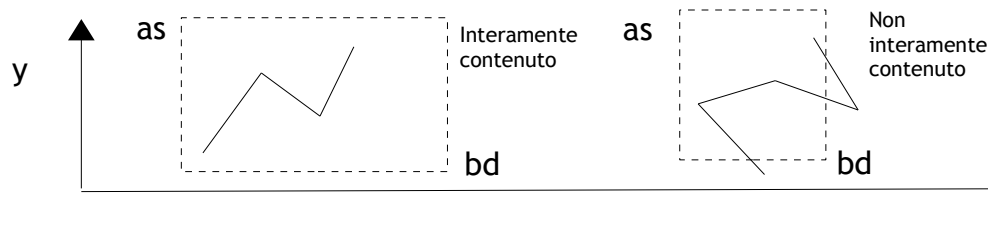
ogni riga contiene i dati di una linea autobus, nell'ordine il nome della linea (una parola), il numero di fermate relative a tale linea (un intero) e gli identificatori delle fermate (sequenza di interi).

La funzione restituisce *true* se il caricamento va a buon fine, *false* altrimenti.

bool percorsoPiuLungo(const Piano* pp, char ris[]) riempie l'array `ris` con il nome della linea che fa il percorso più lungo. La distanza tra due fermate può essere approssimata con la distanza euclidea. La funzione restituisce *true* se c'è almeno una linea, *false* altrimenti.

void perIndirizzo(const Piano* pp, const char ind[]) stampa a video il nome di tutte le linee di autobus che fermano all'indirizzo specificato da `ind`.

int lineeInArea(const Piano* pp, const Coord as, const Coord bd) restituisce il numero di linee il cui percorso è interamente contenuto nell'area rettangolare individuata da `as` e `bd` (`as` specifica le coordinate dell'angolo in alto a sinistra, `bd` le coordinate dell'angolo in basso a destra).



void stampa(const Piano* pp) stampa a video una rappresentazione del piano trasporti come illustrato dal seguente esempio:

via_mazzini	U	lam_verde	lam_rossa	lam_blu
via_verdi	U	lam_verde	lam_rossa	lam_blu
pza_garibaldi	U	lam_rossa		
via_del_bren.	E	lam_blu		
via_rindi	U	lam_verde	lam_blu	
via_aurelia	E	lam_blu		
via_oberdan	U	lam_verde		

Ogni riga corrisponde ad una fermata e vengono stampati nell'ordine: l'indirizzo associato alla fermata, il carattere U/E a seconda che la fermata sia urbana o extraurbana, e i nomi delle linee che passano da tale fermata.