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).

Treni

Un programma per la gestione degli orari e dei percorsi dei treni può essere realizzato usando le seguenti costanti e i seguenti tipi:

```
const int MAXS = 100;
const int MAXT = 50;
const int MAXP = 80;

const float COSTO_KM_REG = 0.1f;
const float COSTO_KM_IC = 0.2f;
const float COSTO_KM_FRECCIA = 0.3f;

enum Tipo {REG, IC, FRECCIA};

struct Orario {
  int minuti;
  int ora;
};

struct Segmento {
  char stazionePart[MAXS];
  Orario oraPart;
  char stazioneArr[MAXS];
  Orario oraArr;
  float km;
};

struct Treno {
  int id;
  Tipo tipo;
  Segmento segm[MAXT];
  int numSegm;
};

struct Programmazione {
  Treno tre[MAXP];
  int numTre;
};
```

Il tipo `Tipo` viene usato per per rappresentare la tipologia di un treno (regionale, intercity, etc). Le costanti `COSTO_KM_...` indicano il costo chilometrico per i diversi tipi di treno. Il significato della struttura `Orario` è evidente. La struttura `Segmento` contiene le informazioni relative a un tratto tra due stazioni consecutive. In particolare vengono memorizzate: la stazione di partenza, l'orario di partenza, la stazione di arrivo, l'orario di arrivo e la distanza percorsa (in km). La struttura `Treno` memorizza i dati di un treno (il campo `id` è un identificatore univoco, il campo `tipo` indica la tipologia del treno) e il suo percorso. Il percorso di un treno è composto da un certo numero di segmenti memorizzati nell'array `segm`; il campo `numSegm` indica il numero di segmenti che compongono il percorso di un treno. La struttura `Programmazione` contiene i dati di tutti i treni (nell'array `tre` e il campo `numTre` ne indica il numero effettivo).

Scrivere il corpo delle seguenti funzioni C++.

1. **`float costoBiglietto(const Programmazione* pp, int id)`** calcola e restituisce il costo del biglietto del treno con identificatore `id` (per andare dalla stazione di origine a quella finale).

2. **`int partenza(const Programmazione* pp, const char staz[], Orario inizio, Orario fine, int vett[])`** riempie l'array `vett` con gli id di tutti i treni che partono dalla stazione `staz` in un momento qualunque compreso tra gli istanti `inizio` e `fine` (estremi inclusi). La stazione `staz` può essere sia la stazione di origine del treno che una stazione intermedia. La funzione restituisce il numero di valori inseriti in `vett`.

3. **`bool salva(const Programmazione* pp, const char nomef[], const char p[], const char a[])`** salva nel file di nome `nomef` l'elenco di tutti i treni che possono essere usati per andare da `p` ad `a`. La funzione restituisce *true* se l'operazione va a buon fine, *false* altrimenti. L'elenco deve avere il formato indicato dal seguente esempio:

   ```
   4455 IC Pisa 11:23 Grosseto 13:11
   9999 FRECCIA Pisa 9:10 Grosseto 10:20
   8888 FRECCIA Pisa 10:42 Grosseto 11:30
   ```

4. **`void stampaOrd(const Programmazione* pp)`** stampa a video l'elenco di tutti i treni contenuti nella programmazione e per ognuno di essi la stazione di origine. L'elenco deve essere ordinato per valori alfabeticamente crescenti delle stazioni di origine e deve avere il formato indicato dal seguente esempio:

   ```
   7890 Livorno
   8888 Massa
   3312 Pisa
   4455 Pisa
   2211 Pisa
   1123 Pisa
   9999 Pisa
   5555 Pisa
   7777 Pisa
   ```

5. **`Tipo tipoPiuNumeroso(const Programmazione* pp)`** restituisce il tipo di treno che appare più frequentemente nella programmazione.