Fondamenti di Informatica 27/09/2017

Contesto

Un file di testo memorizza le informazioni relative a dei percorsi di viaggi in automobile. Ciascun percorso viene descritto come una sequenza di tratti rettilinei di cui sono fornite le coordinate rispetto a un punto di origine.
Il formato con cui viene specificato ciascun viaggio nel file è il seguente:

- una prima riga riporta le coordinate X e Y (numeri interi, anche negativi) del punto di partenza;
- nelle righe successive vengono riportate le informazioni relative ad ogni tratta del viaggio.

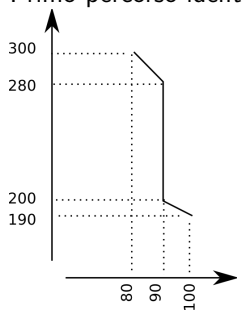
Per ogni tratta vengono forniti i seguenti dati, uno per riga:

- le coordinate X e Y (numeri interi, anche negativi) del punto di arrivo della tratta, raggiunto partendo dal punto indicato nella riga precedente;
- l'altimetria della tratta, specificata da un singolo carattere che può assumere i valori `'P'` o `'M'` (maiuscoli) per indicare rispettivamente una strada in pianura o di montagna;
- il tipo di strada percorsa, indicata da un singolo carattere che può assumere i valori `'A'`, `'E'`, o `'U'` (maiuscoli) ad indicare rispettivamente una tratta autostradale, extraurbana oppure urbana.

Esempio del contenuto del file:

```
100 190
90 200 P U
90 280 P A
80 300 M E
0 10
0 100 P A
10 110 M E
```

Primo percorso identificato nell'esempio a sinistra:



In un file possono essere contenuti uno o più percorsi. Ciascun percorso inizia con una riga contenente le coordinate del punto di partenza. Ciascun percorso può contenere al massimo 10 tratte.
Il precedente esempio contiene 2 percorsi, il primo composto da 3 tratte e il secondo da 2 tratte.

Informazioni sul programma richiesto

Si scriva un programma in linguaggio C in grado di elaborare un file avente il formato descritto, al fine di restituire i risultati indicati nei punti specificati di seguito. Il programma deve poter essere invocato da linea di comando. Un esempio di invocazione è la seguente:

```
./a.out nome_input_file
```

dove `a.out` è il nome del programma eseguibile da invocare; `nome_input_file` è il nome del file di dati da elaborare.
IMPORTANTE: il programma finale dovrà produrre la stampa di risultati esattamente col formato specificato nei vari punti. In particolare, *non aggiungere all'output del testo non richiesto*.
Eventuali righe di output aggiuntive che si vogliono generare in fase di debug, ma che si vogliono escludere dai test, possono essere stampate includendo in prima posizione il carattere $\#$.

Il buon funzionamento del programma può essere verificato col comando

```
./pvcheck ./a.out
```

dove `a.out` è il nome del file eseguibile.

RICHIESTE

1 Numero tratte per tipo altimetria

Stampare il *numero complessivo* di tratte percorse in pianura e in montagna. Stampare il risultato col seguente formato:

```
[ALTIMETRIA]
numero_tratte_in_pianura
numero_tratte_in_montagna
```

2 Distribuzione lunghezza tratte

Determinare il numero di tratte la cui lunghezza ricade negli intervalli $[10x, 10(x+1))$, con $x = [0 \dots 20]$. Stampare il valore massimo dei conteggi calcolati con il seguente formato:

```
[MAX-COUNT]
massimo_conteggio
```

3 Lunghezza del percorso

Calcolare la lunghezza totale di ciascun percorso, esprimendola in chilometri e arrotondando il valore all'intero più vicino. Determinare e stampare la lunghezza più elevata tra tutte quelle calcolate.
Stampare il risultato col seguente formato:

```
[LUNGHEZZA]
lunghezza_max
```

4 Ordinamento per distanza

Sia `n` il numero di tratte, indipendentemente dal percorso di appartenenza. Stamparle ordinandole in senso crescente di distanza tra il **punto di destinazione** e **l'origine del sistema di riferimento**. Il formato della tratta `i`-esima include le coordinate di destinazione `Xi` e `Yi`, l'altimetria `Ai` e il tipo di strada `Ti`, separati da singoli spazi:

```
[ORDINAMENTO]
X1 Y1 A1 T1
X2 Y2 A2 T2
...
Xn Yn An Tn
```

Note

- salvare il proprio programma nella directory di lavoro
- assegnare il nome del file in base al proprio cognome, chiamandolo **cognome.c**
- il primo commento del programma deve riportare **nome e cognome** e **numero di matricola**
- vengono valutati positivamente aspetti quali la leggibilità del programma, una buona formattazione del sorgente, l'uso appropriato dei commenti, modularità e generalità del codice
- è possibile far uso di manuali, testi, appunti e dispense, ma non di eserciziari (raccolte di esercizi risolti)