ESERCIZIO FILTRO
Scrivere un programma che riceve una stringa da riga di comando e stampa la stringa di N caratteri in un quadrato di dimensione NXN. Si assuma che la stringa utilizzi solo caratteri ASCII.

```
Esempio d'esecuzione:
$ go run esercizio_filtro.go 30elode
3     3
 0   0
  e e
   l
  o o
 d   d
e     e

$ go run esercizio_3.go mano
m  m
 aa
 nn
o  o
```

ESERCIZIO 1
Scrivere un programma 'temperature.go' che legge da standard input una sequenza (terminata da EOF) di almeno una temperatura e stampa:
- la temperatura minima e il numero di volte che è stata registrata
- la temperatura massima e il numero di volte che stata registrata
- tutte le temperature con le occorrenze
- le temperature che si sono verificate con la massima frequenza.

Esempio
```
$ go run esercizio_1.go
2 3 4 -2 2 6 12 8 3 1 -3
0 -2 5 2 4 1 -1 0 7 11 8 3

produce in output:
max: 12, 1 volte
min: -3, 1 volte
-3:1 -2:2 -1:1 0:2 1:2 2:3 3:3 4:2 5:1 6:1 7:1 8:2 11:1 12:1
temperature [2 3] con frequenza 3
```

ESERCIZIO 2
Scrivere un programma che legga da riga di comando una sequenza e ne stampi in ordine alfabetico le sottosequenze che iniziano e finiscono con la stessa lettera,senza distinzione tra maiuscole e minuscole, e le occorrenze.
Si assuma che la sequenza sia formata da soli caratteri ASCII
A tale fine definire la funzione:
- `GeneraSottosequenze(sequenza []string) (sottosequenze map[string]int)` che riceve in input una sequenza di tipo `[]string` e restitisce una mappa contenente le sottosequenze e le occorrenze

```
Esempio d'esecuzione:
$ go run esercizio_2.go a b B a b
abba 1
bab 1
bb 1
```

bbab 1

ESERCIZIO 3

Parte 1

Il piano cartesiano è diviso in quattro quadranti: I, II, III e IV quadrante (senso antiorario, I in alto a dx).
Sul piano cartesiano, ad ogni punto individuato da una coppia di numeri reali, chiamati rispettivamente ascissa e ordinata, può essere associata un'etichetta simbolica, generalmente una lettera maiuscola.

Scrivere un programma che:
- legga da standard input una sequenza di righe di testo;
- termini la lettura quando, premendo la combinazione di tasti Ctrl+D , viene inserito da standard input l'indicatore End-Of-File (EOF).

Ogni riga di testo è una stringa che specifica l'etichetta del punto (ad es.: A , B , ...), l'ascissa e l'ordinata del punto nel seguente formato: etichetta;x;y

Esempio: Si ipotizzi che vengano inserite da standard input le seguenti di righe di testo:
```
A;10.0;2.0
B;11.5;3.0
C;8.0;1.0
D;14.0;-1.0
```
tali righe specificano 4 punti: A(10, 2) , B(11.5, 3) , C(8, 1) e D(14, -1) . Ogni tripla di punti letti in input definisce un triangolo che ha per vertici i punti stessi.

Esempio: dati i punti dell'esempio precedente, i triangoli che possono essere definiti sono 4: `ABC` , `ABD` , `ACD` e `BCD`.

Definire:
- la struttura `Punto` per memorizzare l' etichetta , l' ascissa e l' ordinata di un punto sul piano cartesiano;
- la struttura `Triangolo` per memorizzare le 3 istanze di tipo Punto che definiscono i 3 vertici di un triangolo.

Implementare le funzioni:
- `LeggiPunti() (punti []Punto)` che:
  - legge da standard input una sequenza di righe di testo nel formato etichetta;x;y , terminando la lettura quando viene letto l'indicatore End-Of-File (EOF);
  - restituisce un valore `[]Punto` nella variabile punti in cui è memorizzata la sequenza di istanze del tipo `Punto` inizializzate con i valori letti da standard input;
- `GeneraTriangoli(punti []Punto) (triangoli []Triangolo)` che riceve in input una slice di istanze di tipo `Punto` nella variabile punti e restituisce una slice di istanze di tipo `Triangolo` nella variabile `triangoli` in cui sono memorizzati tutti i triangoli che possono essere generati a partire dai punti in `punti`.

Si assuma che:
- le righe di testo lette da standard input siano nel formato corretto;
- la tripla di valori presente in ogni riga specifichi correttamente un punto sul piano cartesiano;
- vengano lette da standard input almeno 3 righe di testo che specificano 3 punti distinti sul piano cartesiano.

Parte 2

La lunghezza di ciascun lato di un triangolo è pari alla distanza euclidea tra gli estremi del lato. Per esempio, la lunghezza del lato AB del triangolo ABD è pari alla distanza euclidea tra i punti A e B : $\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2)}$

Una volta terminata la fase di lettura delle istanze di tipo Triangolo , il programma deve stampare a video (come mostrato nell'Esempio di esecuzione) la rappresentazione string del triangolo rettangolo tale che:

1. uno dei due cateti del triangolo rettangolo sia parallelo all'asse delle ascisse (l'altro cateto deve essere quindi parallelo all'asse delle ordinate);

2. i vertici del triangolo rettangolo giacciano tutti nello stesso quadrante del piano cartesiano;

3. abbia area maggiore tra tutti i triangoli rettangoli che soddisfano le condizioni 1 e 2.
   Si ricorda che è possibile calcolare l'area di un triangolo rettangolo come: area = cateto1 cateto2 / 2 oppure con la formula di Erone: area = (p (p - l1) (p - l2) (p - l3))1/2

Se non esistono triangoli rettangoli che soddisfano le condizioni 1 e 2, il programma non deve stampare nulla. Oltre alla funzione `main()`, devono essere definite ed utilizzate almeno le seguenti funzioni:

- `Distanza(p1, p2 Punto) float64` che riceve in input due istanze di tipo `Punto` nei parametri `p1` e `p2` e restituisce un valore `float64` pari alla distanza euclidea tra i punti rappresentati da `p1` e `p2` ;
- `StringPunto(p Punto) string` che riceve in input un'instanza del tipo `Punto` nel parametro `p` e restituisce un valore `string` che corrisponde alla rappresentazione `string` di `p` nel formato `etichetta = (x, y)` ;
- `StringTriangolo(t Triangolo) string` che riceve in input un'instanza del tipo `Triangolo` nel parametro `t` e restituisce un valore `string` che corrisponde alla rappresentazione `string` di `t` nel formato `Triangolo rettangolo con vertici VERTICE_1, VERTICE_2 e VERTICE_3, ed area AREA.` , dove `VERTICE_1` , `VERTICE_2` e `VERTICE_3` sono le rappresentazioni `string` delle istanze del tipo `Punto` che rappresentano i vertici di `t` , mentre `AREA` è il valore `float64` che specifica l'area di `t` .

Esempio d'esecuzione:

```
$ cat punti1.txt
A;4;8
B;4;2
C;7;2
D;5;2
E;4;9
F;5;9
G;-5;2
H;-4;9
I;4;-3
L;-5;-3
M;-5;3

$ go run esercizio_3.go < punti1.txt
Triangolo rettangolo con vertici B = (4.0, 2.0), C = (7.0, 2.0) e E =
(4.0, 9.0), ed area 10.5.

$ cat punti2.txt
A;4;8
B;4;3
C;7;2
D;5;-2
E;7;7
F;-5;9
G;-5;2
H;-4;9
```

```
I;4;-3
L;5;-3
M;-5;3

$ go run esercizio_3.go < punti2.txt
Triangolo rettangolo con vertici F = (-5.0, 9.0), G = (-5.0, 2.0) e H =
(-4.0, 9.0), ed area 3.5.

$ cat punti3.txt
A;4;8
B;5;3
C;7;2

$ go run esercizio_3.go < punti3.txt
```