=== Cancellazioni ===
Scrivere un programma 'cancellazioni.go' dotato di:
- una funzione func cancella(lista []string) []string che, per ogni numero n >0 (intero) presente in lista, cancella il numero stesso e gli n elementi successivi della lista (o quelli che ci sono per arrivare alla fine della lista) e restituisce la nuova lista così prodotta;

- una funzione main() che legge da file (il cui nome viene passato come parametro su linea di comando) un testo di parole e numeri non negativi, stampa la lista stessa e poi la lista ottenuta cancellando, per ogni numero n presente in lista, il numero stesso e gli n elementi successivi (vedi sopra).

Se il programma viene lanciato con un numero di argomenti
diverso da 1, deve terminare stampando "Fornire 1 nome di file".
Se invece il file non esiste, deve terminare stampando "File non accessibile".

Esempi
$ go run cancellazioni.go uno.input
[uno due 2 tre quattro cinque 1 sei sette 5 otto nove dieci]
[uno due cinque sette]

$ go run cancellazioni.go due.input
[uno due 2 tre quattro cinque 1 sei sette 1 otto nove dieci]
[uno due cinque sette nove dieci]

$ go run cancellazioni.go
Fornire 1 nome di file

$ go run cancellazioni.go FileNonEsistente.txt
File non accessibile

=== Cerchi ===
Scrivere un programma 'cerchi.go' dotato di:

- una struttura 'Cerchio' con campi:
  nome string
  x,y,raggio float64
    (dove x e y sono le coordinate del centro)

- una funzione 'newCircle(descr string) Cerchio' che data una stringa che descrive un cerchio (nome, coordinate x e y del centro, raggio separati da spazi) restituisce il cerchio corrispondente

- una funzione 'distanzaPunti(x1,y1,x2,y2 float64) float64 che restituisce la distanza tra i punti di coordinate (x1,y1) e (x2,y2)

- una funzione 'tocca(cerc1, cerc2 Cerchio) bool' che restituisce true se i due cerchi sono tangenti (internamente o esternamente); false altrimenti. Trattandosi di valori float, consideriamo una distanza trascurabile se è inferiore a 10^-6 (1e-6)

- una funzione maggiore(cerc1, cerc2 Cerchio) bool che restituisce true se cerc1 è più grande di cerc2; false altrimenti.

- una funzione main() che legge da standard input una sequenza (terminata da ctrl D) di righe che descrivono cerchi, del tipo:

uno 10.3 12.7 45.8
due 1.3 2.9 5.8
pippo 7.3 22.5 6.6

ciascuna contenente nome, coordinate del centro e raggio di un cerchio, in quest'ordine.

Il programma crea per ciascuna riga il cerchio corrispondente, lo stampa e stampa se il cerchio, rispetto a quello precedente, è tangente o no e maggiore o no.
Il primo confronto è fatto rispetto al cerchio "zero" ("", 0, 0, 0).

Esempio
(vengono marcate con '>' le righe digitate da tastiera,
le altre sono l'output del programma)

>uno 0.5 0 2.5
{uno 0.5 0 2.5} non tangente, maggiore
>due 0 0 3
{due 0 0 3} tangente, maggiore
>tre 10.2 -8.4 1.5
{tre 10.2 -8.4 1.5} non tangente, minore o uguale

==== Statistiche ====
Scrivere un programma in Go (il file deve chiamarsi 'statistiche.go') dotato di

- una funzione moda(serie []float64) float64 che, data una serie di numeri, restituisce la loro moda (*). In presenza di due (o più) mode si restituisca la più piccola.

- una funzione mediana (serie []float64) float64 che, data una serie di numeri, restituisce la loro mediana (**)

- una funzione main() che legge una serie di numeri float64 da standard input, uno per riga,
  (la lettura viene terminata con invio seguito da Ctrl D))
  ed emette su standard output la loro moda e la loro mediana.

Note
Sia la moda che la mediana sono indipendenti dall'ordine dei valori nella serie.
(*) La moda di una serie di numeri è il valore che compare più frequentemente nella serie.
(**) La mediana di una serie di numeri è
  - nel caso di serie di lunghezza dispari, il valore centrale della serie ordinata in ordine crescente
  - nel caso di serie di lunghezza pari, la media dei due valori centrali della serie ordinata in ordine crescente

Suggerimenti
Il package sort fornisce funzioni utili per ordinare una serie di valori

Esempi
1) data la serie in input:
1
2
5
4
3
si ottiene:
moda: 1
mediana: 3

2) data la serie in input:
4
3
2
1
si ottiene:
moda: 1
mediana: 2.5

3) data la serie in input:
4.1
2.7
4.1
2.7
2.7
4.1
2
5
si ottiene:

moda: 2.7
mediana: 3.4