Filtro
Scrivere un programma 'filtro.go' dotato di:
- una funzione DrawPoint(c byte, k int) string che restituisce una stringa formata da k spazi bianchi seguiti dal carattere c

- una funzione DrawSegment(c byte, k, l int) string che restituisce una stringa formata da k spazi bianchi seguiti da l caratteri c

- una funzione main() che legge come parametri da linea di comando (in quest'ordine) due numeri interi l e n e un carattere c (byte), e, se l e n sono > 0, produce su standard output una scala di n gradini di lunghezza e altezza l disegnati usando il carattere c (vedi esempi sotto), altrimenti non fa niente.

Si può assumere che il programma venga lanciato con tre parametri dei tipi attesi (non occorre cioè fare controlli).

Esempi
```
$ go run filtro.go 3 1 x
xxx
   x
   x

$ go run filtro.go 3 2 +
+++
   +
   +
   +++
      +
      +
```

Appuntamenti
- una struttura Appuntamento con tre campi int (giorno, oraInizio, oraFine) che rappresentano il giorno dell'anno dell'appuntamento, l'ora di inizio e quella di fine, rispettivamente. Si considerano per semplicità solo appuntamenti che iniziano e finiscono nella stessa giornata e a ore precise (intere).

- una funzione NewAppuntamento(gg, h1, h2 int) (Appuntamento, bool) che, se i parametri sono validi (giorno tra 1 e 366, ora inizio precedente o uguale a ora di fine e tutte e due tra 0 e 24) crea un appuntamento con quei dati e lo restituisce insieme a 'true', altrimenti restituisce un Appuntamento zero-value e 'false'.

- una funzione CheckSovrapposizioneAppuntamenti(app1, app2 Appuntamento) bool che, dati due appuntamenti (si supponga siano validi), restituisce 'true' se si sovrappongono (anche parzialmente), 'false' altrimenti.

- una funzione AddAppuntamento(app Appuntamento, agenda *[]Appuntamento) bool
che, se l'appuntamento app non si sovrappone ad altri già presenti in agenda, lo aggiunge all'agenda e restituisce 'true', altrimenti restituisce 'false'

- una funzione AppuntamentiGiornata(gg int, agenda []Appuntamento) []Appuntamento che, dati un giorno e una agenda di appuntamenti, estrae e restituisce gli appuntamenti del giorno passato come parametro

- una funzione main() che legge da standard input righe che contengono tre interi che rappresentano nell'ordine giorno ora-inizio ora-fine (separatore è solo whitespace) e per ciascuna aggiunge (se valido e non in sovrapposizione) il corrispondente appuntamento.

La lettura termina con EOF (CTRL D) e a quel punto il programma visualizza l'agenda degli appuntamenti (in ordine qualsiasi) su standard output.

Sistema di numerazione n-ario

Si consideri il sistema di numerazione posizionale S3 in base 3 avente come cifre i tre simboli

z u d

i cui valori sono rispettivamente:

val(z) = 0
val(u) = 1
val(d) = 2

Realizzare un programma s3toint.go che, dato un numerale num in S3 come argomento sulla linea di comando, stampi il valore decimale di num.

Il programma deve (unicamente) controllare che l'argomento sia un numerale in S3 e stampare "input non valido" nel caso non lo sia. Non sono richiesti altri controlli (si può dare per scontato che l'argomento ci sia).

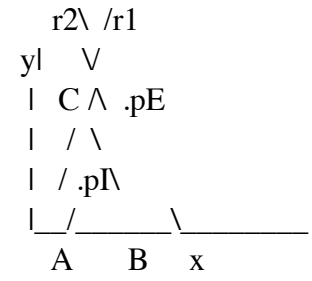
E` possibile dotare il programma di funzioni aggiuntive oltre a main().

Nota. Si ricorda che un sistema di numerazione si dice posizionale se i simboli (cifre) usati per scrivere i numeri assumono valori diversi a seconda della posizione che occupano. Il sistema di numerazione che usiamo solitamente è un sistema posizionale in base 10.

Esempi

```
------
$ go run S3ToInt.go uzz
9
$ go run S3ToInt.go dddd
80
$ go run S3ToInt.go duz
21
$ go run S3ToInt.go uzc
input non valido
$ go run S3ToInt.go udzzzduuz
11001
```

Triangolo
Scrivere un programma triangolo.go che verifica l'appartenenza
(cioè all'interno o sul suo perimetro) di un punto (x,y) a un triangolo ABC,
la cui base AB si trova sull'asse delle ascisse e di cui sono date
le equazioni delle rette su cui stanno i lati AC e BC (vedi figura).



Nota: Una retta è specificata attraverso il coefficiente angolare m
e il termine noto (intercetta) q della sua equazione y = m·x+q.

In particolare il programma stampa il punto e uno dei messaggi
(vedi anche esempi sotto)

- fuori
- dentro

a seconda che il punto appartenga o no.

Il programma deve essere dotato di

- una struttura Punto con due campi float64
per le coordinate x e y del punto

- una struttura Retta con due campi float64
per il coefficiente angolare m e l'intercetta q della retta

- una funzione
  Sotto(p Punto, r Retta) bool
che restituisce 'true' se il Punto p è sotto o sulla Retta r,
'false' altrimenti

- una funzione main()
che legge da standard input, nell'ordine,
  la m e la q della prima retta
  la m e la q della seconda retta
  la x e la y di un punto
verifica se il punto appartiene o no
al triangolo e stampa il messaggio opportuno.

Non si devono fare controlli sulle rette, si assuma cioè che definiscano un triangolo,

come mostrato nella figura sopra.

ESEMPI

go run triangolo.go
2 -2
-2 20
3 5
{3 5} fuori

go run triangolo.go
1 4
-1 12
1 1
{1 1} dentro

Vocali

Scrivere un programma vocali.go che analizza un testo e conta
le occorrenze delle vocali (sia minuscole che maiuscole, ma non le accentate)
nel testo.
In particolare il programma è dotato di:

- una funzione per contare le occorrenze delle diverse vocali in una stringa
  func ContaVocali(s string, vocali map[rune]int)
che, data una stringa s e una mappa vocali, aggiorna opportunamente la mappa vocali
aggiungendo eventuali vocali non ancora presenti / incrementandone i valori

- una funzione
  func main()
che legge un file, il cui nome è passato su linea di comando, produce
una mappa tra vocali presenti nel testo e il numero delle loro occorrenze nel testo,
e la stampa.
In mancanza di argomenti a linea di comando, il programma deve stampare:
  Mancano argomenti

ESEMPIO

se il file contiene:

jdhkas c'è dkasjhkdjashkdh askdh ksah @@@ €€€ ### Ħ wi) Ø
qwqwe qwyewquteuqwte q 312312 2312wweqe €łttttt sdasdas
AA JKJLKLJLKJ LIIIIII
U u ù
aeiou
AEIOU

l'output è (salvo l'ordine di stampa):
o : 1
E : 1
u : 4
U : 2
e : 7
A : 3
I : 7
O : 1
a : 8
i : 2