TPFI 2020/21
Hw 3: Prove e Liste Infinite

assegnato: 15 aprile 2021, consegna 2 maggio 2021

Esercizio 1 (Liste Infinite I) Scrivere un programma Haskell che genera la lista infinita di caratteri `insonnia = "1 sheep 2 sheep 3 sheep 4 sheep ..."`. Provare a scrivere un "one-liner", cioè un programma che semplicemente compone opportunamente funzioni. Può essere utile la funzione `show :: Show a => a => String` che trasforma un elemento di qualsiasi tipo che implementa la classe `Show` in una stringa, cioè una lista di caratteri.

Esercizio 2 (Funtori) Considerare la definizione del tipo alberi n-ari:

```
data Tree a = T a [Tree a] | E
```

(i costruttori sono `T` ed `E` per l'albero vuoto). Definire le funzioni `map` e `foldr` per questo tipo di dato e dare alcuni esempi significativi di uso di queste due funzioni.

Esercizio 3 (Liste Infinite II) Definite in Haskell la lista infinita di liste `tartaglia`, tale che `tartaglia!!n!!k` sia il coefficiente binomiale $\binom{n}{k}$.

Esercizio 4 (Proofs) Considerare la seguente specifica della funzione `scanr`:

```
scan r f e = map (foldr f e) .  tails
```

Derivare una definizione efficiente (=lineare nella lunghezza della lista) facendo manipolazioni algebriche, in analogia con quanto visto per `scanl` (slide lezione **8**, verrà rivisto nella lezione **13**).

Esercizio 5* (Liste Infinite III) Dare una definizione circolare in Haskell della lista infinita dei numeri di Ulam (lezione **11**), selezionandoli tra le possibili somme tra numeri di Ulam, cioè una definizione nella forma:

```
allSums (x:xs) = map (x+) xs :  allSums xs
      ulams = 1:2:f (allSums ulams)
```

per una certa funzione `f`. Ovviamente, `f` può usare altre funzioni ausiliarie.