Programowanie funkcyjne – laboratorium 8
2.12.2009 r.

1. Napisz rekurencyjnie ogonowe wersje funkcji:

   - `len : 'a list -> int` obliczającej długość listy,
   - `sum : int list -> int` obliczającej sumę liczb na liście,
   - `flatten : 'a list list -> 'a list`
     `flatten [[1;2;3];[5;6];[];[7;5]] = [1;2;3;5;6;7;5]`
   - `cat : string list -> string` `cat ["ala";"ma";"kota"] = "ala ma kota "`

2. Napisz rekurencyjnie ogonową wersję funkcji `from_1`:
   `from_1 7 = [1;2;3;4;5;6;7]`

3. Napisz rekurencyjnie ogonową wersję funkcji `iter` taką, że:
   `iter f z n = [z; f z; f(f z); ...; f^(n-1) z]` Możesz użyć rekurencyjnie ogonowej wersji funkcji `rev`.

4. Napisz rekurencyjnie ogonową wersję funkcji `map`. Możesz użyć (rekurencyjnie ogonowej) funkcji `rev`.

5. Napisz procedurę `sześciany : int -> int list` taką, że wynikiem `sześciany n` jest lista postaci `[1^3; 2^3; ...; n^3]`. Rozwiązując to zadanie:

   - możesz korzystać wyłącznie z rekurencji ogonowej,
   - jedyne operacje na liczbach, z jakich możesz korzystać to: `+`, `-` oraz porównywanie
   - skorzystaj z tożsamości:

   $$(n+1)^3 = n^3 + 3n^2 + 3n + 1,$$
   $$(n+1)^2 = n^2 + 2n + 1.$$

   Twoje rozwiązanie powinno działać w czasie $\Theta(n)$.