Programowanie funkcyjne — kolokwium nr 1, 7.12.2016

Instrukcja: Rozwiązania zadań należy przesłać do godziny 9:30 na adres `kolokwium.pf@gmail.com` (decyduje data stempla googlowego). Każde zadanie należy przesłać w oddzielnym pliku: zadanie1.hs, zadanie2.hs i zadanie3.hs. Plików nie należy zipować. W rozwiązaniach nie można korzystać z modułów innych niż standardowe; niedozwolone jest użycie polecenia `import`. Rozwiązania nie spełniające powyższych wymogów nie będą oceniane. Punktacja: 10 punktów za każde zadanie. Uwaga: korzystanie z internetu poza wyznaczonym czasem skutkuje automatycznym otrzymaniem 0 punktów.

Zadanie 1. Napisać funkcję pownum :: Integer → [Integer], która dla podanego $n \geqslant 1$ zwraca (być może nieskończoną) listę składającą się ze wszystkich liczb takich, że suma n-tych potęg cyfr równa jest tej liczbie. Zatem np.

```
pownum 5 = [1, 4150, ...]
```

gdyż $4^5 + 1^5 + 5^5 + 0^5 = 4150$.

Zadanie 2. Napisać funkcję ps, która dla podanej listy zwraca listę zawierającą wszystkie jej prefiksy — w kolejności od najkrótszego (jednoelementowego) do najdłuższego (cała lista) — a następnie kolejno coraz krótsze sufiksy tej listy. Przykładowo:

```
ps "Test" = ["T", "Te", "Tes", "Test", "est", "st", "t"]
ps [3,5,2] = [[3], [3,5], [3,5,2], [5,2], [2]]
```

Dla pustej listy wynik może być dowolny. W rozwiązaniu należy w istotny sposób użyć funkcji foldl lub foldr.

Zadanie 3. Rododendronem nazywamy drzewo, w którym każdy wierzchołek może mieć dowolną liczbę potomków (być może zero). Rododendron nie może być pusty. Stworzyć typ Rd a, przechowujący elementy typu a w rododendronie, i zdefiniować funkcje:

```
el :: Eq a => Rd a -> a -> Bool
subst :: Eq a => a -> a -> Rd a -> Rd a
rd2list :: Rd a -> [a]
```

Funkcje mają, odpowiednio: sprawdzać czy podany element należy do rododendronu, zamieniać wszystkie wystąpienia pierwszego podanego elementu na drugi oraz zamieniać rododendron na listę zgodnie z porządkiem preorder.