Programowanie funkcyjne — kolokwium nr 1, 4.12.2018

Instrukcja: Rozwiązania należy przesłać do godziny 9:45, w jednym mailu, na adres kolokwium.pf@gmail.com. Każde zadanie należy przesłać w oddzielnym pliku: zadanie1.hs, zadanie2.hs i zadanie3.hs. Plików nie należy zipować. Rozwiązania muszą się poprawnie kompilować. 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. Rozważmy ciągi $(a_n), (b_n)$, które spełniają następującą zależność rekurencyjną

$$\begin{aligned} a_n &= (n-1)b_{n-1} - 3a_{n-1} \\ b_n &= 3b_{n-1} + (n-1)^2 a_{n-1} - (n-1)^2 \end{aligned}$$

Dodatkowo wiemy, że $a_0 = b_0 = 1$. Napisać funkcję seqIndex m, która zwraca *najmniejsze* k takie, że $a_0 + a_1 + \ldots + a_k \geqslant m$. Na przykład seqIndex 100 = 4, seqIndex 1000000 = 8.

Zadanie 2. Rozważmy typ danych

```
data Expr a = Value a
            | Add (Expr a) (Expr a)
            | Mul (Expr a) (Expr a)
            | Sub (Expr a) (Expr a)
            | P
```

przechowujący *częściowe* wyrażenia, tzn. wyrażenia, które zawierają operacje dodawania (Add), mnożenia (Mul) i odejmowania (Sub) oraz wartość P, która oznacza, iż konkretny argument nie jest jeszcze znany. Argumenty do operacji arytmetycznych przechowujemy za pomocą Value. Napisać funkcję eq :: (Eq a) $\rightarrow$ Expr $a \rightarrow$ Expr $a \rightarrow$ Bool, która zwraca True, jeśli wyrażenia są takie same, i False w przeciwnym wypadku. Przyjmujemy, że dwa wyrażenia są takie same, jeśli jedno można otrzymać z drugiego przez zamiany P na dowolne inne wyrażenia. Na przykład

```
eq (Add (Value 1) (Value 2)) (Add (Value 1) (Value 3)) = False
eq (Add (Value 1) (Value 2)) P = True
```

Zadanie 3. Napisać funkcję cykl, która dla podanej niepustej listy zwraca listę jej wszystkich przesunięć cyklicznych, w dowolnej kolejności. Podać najogólniejszą możliwą sygnaturę. Funkcja ma w nietrywialny sposób korzystać z foldl lub foldr, przy czym fold musi stanowić najbardziej zewnętrzną część definicji, np. cykl ℓ = foldl ... Przykładowo, wywołanie cykl [1,2,3] powinno zwrócić [[1,2,3], [2,3,1], [3,1,2]] lub dowolną permutację takiej listy.