Второ контролно по Функционално програмиране

спец. Информатика и Софтуерно инженерство, 17.01.2018 г. *Вариант А*

Задача 1. (10 т.) Да се напише функция `mostFrequent`, която по даден списък от списъци от числа връща числото, което е сред най-често срещаните числа във всички списъци, ако такова има, или 0 иначе.
Пример: `mostFrequent [[1,1,3,2],[1,1,5],[1,5],[1,1,1,3]] → 1`
Пример: `mostFrequent [[1,1,3,2],[1,5,5],[1,5],[1,1,1,3]] → 0`

Задача 2.

1. (6 т.) Да се напише функция `grow t x`, която по дадено двоично дърво от числа `t` получава ново, в което към всяко листо на `t` добавя по две нови листа със зададена стойност `x`.
2. (6 т.) Двоично дърво наричаме "пълно", ако има $\mathbf{2^n}$ елемента на ниво **n**. Да се напише функция `growingTrees`, която генерира безкраен поток от пълни дървета с височини съответно 1, 2, 3,..., като всички елементи на ниво **n** са със стойност **n**.

Задача 3. Телевизионно предаване се представя с наредена тройка от име (низ), начален час (наредена двойка от час и минути) и продължителност (брой минути). Телевизионна програма наричаме последователност от предавания, чиито интервали на излъчвания са подредени в нарастващ ред и не се пресичат.

1. (5 т.) Да се напише ункция `lastShow`, което по списък от предавания връща името на това, което завършва най-късно.
2. (10 т.) Да се напише функция `longestProgram`, която по даден списък от предавания генерира възможно най-дълга телевизионна програма, т.е. сумата от продължителностите на предаванията в нея е максимална.

Примери:

```
shows = [("A",(11,0),120),("B",(12,0),15),("C",(10,30),90)]
lastShow shows → "B"
longestProgram shows → [("A",(11,0),120))]
```

Забележка: използването на всички стандартни функции в R^5RS, както и на функциите accumulate, filter, foldr и foldl е позволено, но не е задължително.

Второ контролно по Функционално програмиране

спец. Информатика и Софтуерно инженерство, 17.01.2018 г. *Вариант Б*

Задача 1. (10 т.) Да се напише функция `extremum`, която по даден списък от списъци от числа намира число, което е минимално или максимално във всеки от списъците, ако има такова, или 0 иначе.
Пример: `extremum [[1,2,3,2],[3,5],[3,3],[1,1,3,3]] → 3`
Пример: `extremum [[1,2,3,2],[2,3,5],[3,3],[2,2,3,3]] → 0`

Задача 2.

1. (6 т.) Да се напише функция `clone t x y`, която по дадено двоично дърво от числа `t` получава ново със зададен корен `x` и две поддървета, получени от `t` чрез увеличаване на всичките му елементи със зададено число `y`.
2. (6 т.) Двоично дърво наричаме "пълно", ако има $\mathbf{2^n}$ елемента на ниво **n**. Да се напише функция `cloningTrees`, която генерира безкраен поток от пълни дървета с височини съответно 1, 2, 3,..., като всички елементи на ниво **n** са със стойност **n**.

Задача 3. Телевизионно предаване се представя с наредена тройка от име (низ), начален час (наредена двойка от час и минути) и продължителност (брой минути). Телевизионна програма наричаме последователност от предавания, чиито интервали на излъчвания са подредени в нарастващ ред и не се пресичат.

1. (5 т.) Да се напише функция `isProgram`, което проверява дали даден списък от предавания е телевизионна програма.
2. (10 т.) Да се напише функция `diversestProgram`, която по даден списък от телевизионни предавания генерира възможно най-разнообразна телевизионна програма, т.е. броят на различните (по име) предавания в нея е в максимален.

Пример:
```
shows = [("A",(10,30),90),("B",(11,0),120)),("C",(12,0),15)]
isProgram shows → False
diversestProgram shows → [("A",(10,30),90),("C",(12,0),15)]
```

Забележка: използването на всички стандартни функции в R^5RS, както и на функциите accumulate, filter, foldr и foldl е позволено, но не е задължително.

Задача 2. (12 т.) Нека е даден ориентиран граф със символи по върховете, който е представен чрез списъци от наследници по следния начин:

```
Scheme: (define G '((a b c) (b a c) (c b)))
Haskell: g = [('a', "bc"]), ('b', "ac"), ('c', "b")]
```

Да се напише функция `pathsFrom`, която връща поток от всички пътища от даден връх, подредени в нарастващ ред по дължина.

Пример: `pathsFrom g 'a' → ["a","ab","ac","aba","abc",...]`

Задача 2. (12 т.) Нека е даден ориентиран граф със символи по върховете, който е представен чрез списъци от наследници по следния начин:

```
Scheme: (define G '((a b c) (b c) (c b)))
Haskell: g = [('a', "bc"]), ('b', "c"), ('c', "b")]
```

Да се напише функция `infinitePath`, която връща поток, представящ един безкраен път в графа или празен поток, ако такъв няма.

Пример: `infinitePath g` → ['a','b','c','b','c','b','c',...]