Transakcje Narodowego Banku Polskiego

V Haskellu naprogramujte několik jednoduchých programů, které z jednoho druhu logů z banky zvládnou vytáhnout různé informace. Protože logy jsou z Polské banky, nejsou nijak organizované ani setříděné.

Konkrétně, log je seznam záznamů, kde každý záznam obsahuje:

- Čas akce
- Jméno uživatele
- Akci uživatele, ta může být:
    - Přihlášení uživatele
    - Odhlášení uživatele
    - Výběr nějaké částky z účtu
    - Přípsání nějaké částky na účet

Úkol

Napište jednoduché funkce (celkem 5 + 3 doporučené bonusové), které z tohoto logu dostanou následující informace:

- Seznam uživatelů (bez duplicit), seřazený podle abecedy
- Časy deseti největších výběrů
- Jména uživatelů, kterým přišlo deset nejmenších přípisů
- Jméno uživatele, který je nejaktivnější (tj. má v logu nejvíc záznamů)
- Jméno uživatele, kterému na účtu přibylo nejvíc peněz (tj. má maximální součet příjmů mínus součet výdajů)
- Bonus: Průměrnou částku (oříznutou na celé číslo), kterou vybrali uživatelé začínající od J
- Bonus: Jméno uživatele, který provedl nejvíc akcí za sebou bez toho, aby jakýkoliv jiný uživatel cokoliv udělal (tj. po seřazení logu podle času bude mít “nejvíc řádků po sobě”)
- Bonus: Počet výběrů z účtů uživatelů s pětipísmennými jmény, u kterých se nedá prokázat, že byli přihlášení. (Správně by výběry měly probíhat je po zaznamenaném přihlášení, ideálně bez toho aby se uživatel v mezičase odhlásil.)

Při řešení **nepoužívejte ručně definovanou rekurzi**.

Návod

Stáhněte si testovací data a kostru programu v Haskellu. Kostra programu se stará o načítání a parsování dat (zatím si nemusíte všímat toho, jak to přesně dělá).

Vaše řešení jednotlivých bodů zadání vložte místo undefined do definic odpovídajících funkcí (označeno komentářem), případně si doplňte libovolné množství pomocných funkcí. Do kostry jinak pokudmožno nezasahujte.

Místo ruční rekurze použijte následující:

- filter, map, foldr, foldl, zip, zipWith a ostatní funkce, které definují většinu rozumných tvarů rekurze
- množinovou syntaxi pro vytváření seznamů (tj. např. [a*2 | a <- 1..10]), hodí se především pro pattern-matching na typech transakcí (prakticky jako kombinace filter a map; budeme se tím zabývat na devátém cvičení)
- může se hodit i group, groupBy, nub, sort, sortOn nebo sortBy z knihovny Data.List (na začátek programu připište import Data.List).

Poznámky:

- časovou složitost není potřeba moc řešit — většina otázek jde zodpovědět v O(n), ale O(n log n) úplně stačí (můžete předpokládat, že sort a nub oba fungují v O(n log n))
- hodí se vyrobit si malé “accesory”, tj. funkce které vám umožní nějakou hodnotu vytáhnout bez pattern matchingu. Např. jmeno :: Zaznam -> String jde např. použít pro získání seznamu všech jmen jako map jmeno log
- z jednodušších funkcí je jednoduché tečkou vyrábět složitější: např. obdoba unixového |sort|uniq se v Haskellu napíše nub . sort

Definice a typy všech funkcí (i jiných z Data.List) si můžete najít na Hoogle.

Může se hodit

```
> sortOn negate [1..10]
[10,9,8,7,6,5,4,3,2,1]
> sortOn (`mod` 3) [1..10]
[3,6,9,1,4,7,10,2,5,8]
> groupBy (\x y -> x `div` 3 == y `div` 3) [1..10]
```

[[1,2],[3,4,5],[6,7,8],[9,10]]