

Programowanie funkcyjne — kolokwium nr 1, 13.12.2023

Instrukcja: Każde zadanie należy przesłać na Pegaza 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 znaków złożone z liter `a`, `b` i `c`, w których liczba wystąpień litery `a` jest dwa razy mniejsza niż suma liczb wystąpień liter `b` i `c`. Napisać funkcję, która dla argumentu naturalnego n zwróci listę wszystkich takich ciągów znaków o długości $3n$. Przykładowo dla argumentu 1 powinna być to lista złożona z ciągów `"abb"`, `"abc"`, `"acb"`, `"acc"`, `"bab"`, `"bac"`, `"cab"`, `"cac"`, `"bba"`, `"bca"`, `"cba"`, `"cca"`. Kolejność ciągów w liście jest dowolna.

Zadanie 2. Rozważmy funkcję f określoną na liczbach naturalnych dodatnich, która zwraca $n/2$ dla liczby n parzystej oraz $3n + 1$ dla liczby n nieparzystej. Hipoteza Collatza mówi, że jeśli zaczniemy od dowolnej liczby naturalnej, to wielokrotnie wywołując na niej f zawsze otrzymamy w końcu 1. Zaimplementować funkcję o sygnaturze

```
collatz :: Integer -> [Integer],
```

która wywołana na n zwraca listę zawierającą minimalną liczbę wywołań funkcji f dla kolejnych liczb naturalnych od 1 do n , po której zostanie osiągnięty wynik 1. Wywołanie nigdy się nie kończy, jeśli w liczonym zakresie istnieje kontrprzykład. Przykładowo, `collatz 5` powinno zwrócić `[0, 1, 7, 2, 5]`, ponieważ $f(3) = 10$, $f(10) = 5$, $f(5) = 16$, $f(16) = 8$, $f(8) = 4$, $f(4) = 2$ i $f(2) = 1$.

Zadanie 3. *Wieloszczet nierychliwy* to struktura danych, która gromadzi dane liczbowe wskazanego typu z podziałem na segmenty. Wieloszczet nierychliwy pusty składa się z jednego, pustego segmentu; operacja `wp` zwraca taki właśnie wieloszczet. Operacja `dl` dołącza liczbę do bieżącego (najpóźniej utworzonego) segmentu, zaś `ts` tworzy nowy segment i powoduje, że kolejne operacje `dl` działają na owym nowym segmencie (chyba że bieżący segment jest pusty; wówczas `ts` zwraca wieloszczet bez zmian). Operacja `wmax` podaje listę maksimumów z kolejnych segmentów, poczynając od ostatnio dołączonego (jeżeli bieżący segment jest pusty, to jest pomijany, zatem lista maksimumów jest wówczas o jeden krótsza niż liczba segmentów). Podobnie `wmin` podaje listę minimumów. Operacje `wmax` i `wmin` powinny być wykonalne w czasie liniowym względem liczby segmentów, pozostałe operacje — w czasie stałym.

Zdefiniować typ `Wnr a`, służący do przechowywania elementów typu `a` w wieloszczecie nierychliwym, oraz następujące funkcje, realizujące opisane wyżej operacje z odpowiednią złożonością:

```
wp :: Wnr a
dl :: Num a => Wnr a -> a -> Wnr a
ts :: Wnr a -> Wnr a
wmax :: Wnr a -> [a]
wmin :: Wnr a -> [a]
```