Поправителен изпит по Функционално програмиране - 20.08.2022 г.
Специалност Информационни системи, 1-ви курс

Задача 1

Да се дефинира предикат **`isPerfectSq :: Int -> Bool`**, който за подадено естествено число проверява дали то е точен квадрат. Да се реализира линеен итеративен процес. При невалиден вход да се извежда грешка.

Примери:

```
isPerfectSq 9    → True
isPerfectSq 18   → False
isPerfectSq (-1) → error "Argument has to be a natural number!"
```

Задача 2

Даден е списък от двойки, представящи оценките на ученици от даден клас по даден предмет. Първият елемент на всяка двойка е номерът на ученика в класа, а вторият – получената оценка по предмета (приемаща стойности от 0 до 100).

Да се дефинира процедура **`studAvg :: [(Int, Double)] -> [(Int, Double)]`**, която приема списък от горепосочения вид и изчислява средната стойност на най-добрите пет оценки на всеки ученик. Резултатът да е сортиран спрямо номерата на учениците. Може да се приеме, че всеки ученик има поне пет оценки.

Примери:

```
studAvg [(1, 100), (1, 50), (2, 100), (2, 93), (1, 39), (2, 87), (1, 89), (1, 87), (1, 90), (2, 100), (2, 76)] → [(1, 83.2), (2, 91.2)]
studAvg [(3, 55), (2, 50), (1, 21), (3, 53), (2, 48), (1, 3), (3, 4), (2, 28), (1, 10), (3, 80), (2, 68), (1, 15), (3, 91), (2, 45), (1, 49)] → [(1, 19.6),(2, 47.8),(3, 56.6)]
```

Задача 3

Нека разгледаме думата *abode*. В нея буквата *a* е на позиция *1*, а *b* е на позиция *2*. В английската азбуката *a* и *b* също са на позиции съответно *1* и *2*. Забелязваме също, че и *d* и *e* в *abode* заемат позициите, които биха заели в азбуката: съответно *4* и *5*.

Да се дефинира **на функционално ниво** функция **`solve :: [String] -> [Int]`** , която приема списък от думи и връща списък с броя букви, които заемат своите позиции в азбуката за всяка дума от дадения. Входните данни са съставени само от думи, включващи главни и малки букви от английската азбука.

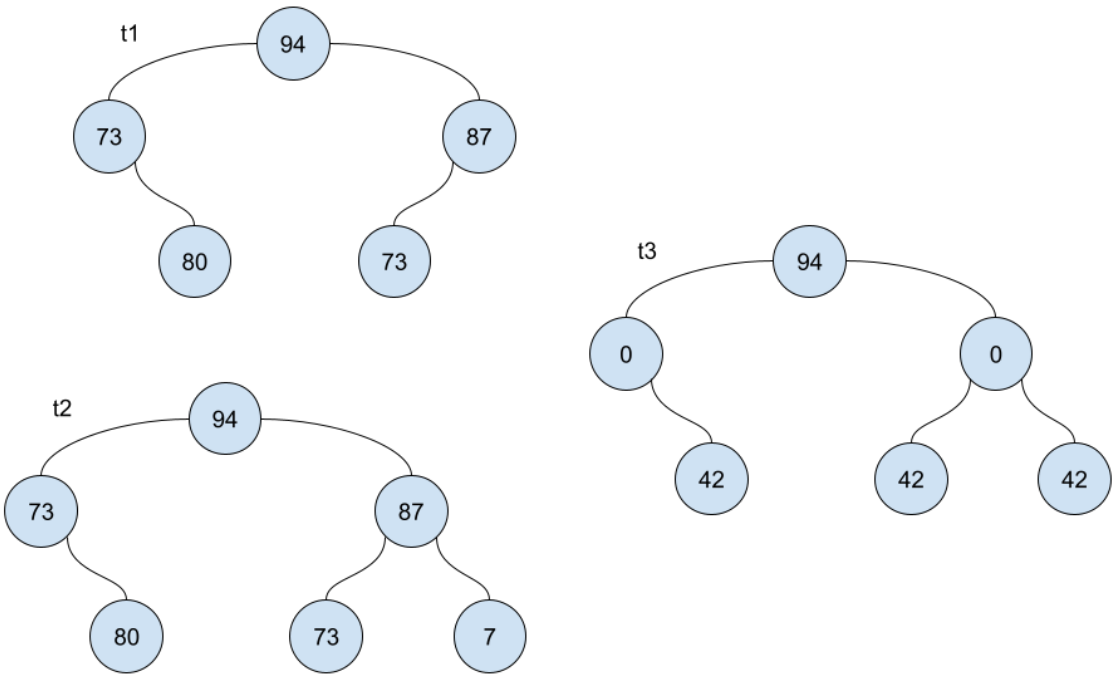
Примери:

```
solve ["abode","ABc","xyzD"]                        → [4,3,1]
solve ["abide","ABc","xyz"]                         → [4,3,0]
solve ["IAMDEFANDJKL","thedefgh","xyzDEFghijabc"] → [6,5,7]
solve ["encode","abc","xyzD","ABmD"]                → [1,3,1,3]
```

Задача 4

Дефиниран е полиморфен алгебричен тип **`BTree a`**, описващ двоично дърво:
`data BTree a = Nil | Node a (BTree a) (BTree a)`.
Да се дефинира функция **`maximumLevel :: (Num a) => BTree a -> Int`**, която намира нивото на дървото, на което сумата от стойностите във възлите е максимална. Коренът на дървото се намира на ниво 1. При няколко нива с еднаква сума, която е и максимална, да се връща нивото, което е най-отдалечено от корена.

Примери:




```
t1 = Node 94 (Node 73 Nil (Node 80 Nil Nil)) (Node 87 (Node 73 Nil
Nil) Nil)
t2 = Node 94 (Node 73 Nil (Node 80 Nil Nil)) (Node 87 (Node 73 Nil
Nil) (Node 7 Nil Nil))
t3 = Node 94 (Node 0 Nil (Node 42 Nil Nil)) (Node 0 (Node 42 Nil
Nil) (Node 42 Nil Nil))

maximumLevel t1 → 2
maximumLevel t2 → 3
maximumLevel t3 → 3
```