prof. Hromkovič

28. februára 2017 o 17.45 hod. v Aula Magna v budove FIIT STU

Jozef – luhn.scala

```
object Luhn {
  def cardnumber(s: String): Boolean = {
    val split = s.split("").reverse.map(_.toInt)
    (split.grouped(2).collect {
       case Array(a, b) =>
              a + (if (b * 2 > 9) b * 2 - 9 else b * 2)
       case Array(a) => a
    }.sum) % 10 == 0
  }
  def main(args: Array[String]): Unit = {
      println(Luhn.cardnumber("49927398716"))  -- true
      println(Luhn.cardnumber("49927398717"))  -- false
      println(Luhn.cardnumber("1234567812345678"))-- false
      println(Luhn.cardnumber("1234567812345670")) -- true
  }}
```

Prepísane do Haskellu

```
cardnumber :: Integer -> Bool
cardnumber s = (sum $
                    map (
                        \zoznam12 -> case zoznam12 of
                            [a,b] -> a + (if (b * 2 > 9) then b * 2 - 9 else b * 2)
                            [a] -> a
                    ) $ splitEvery 2 split ) `mod` 10 == 0
        where split = map (\x -> ord x - ord '0') $ reverse $ show s
        where split = map (\x -> ord x - ord '0') ( reverse ( show s ))

-- toto nájdete v Prelude
splitEvery :: Int -> [a] -> [[a]]
splitEvery _ [] = []
splitEvery n xs = as : splitEvery n bs
            where (as,bs) = splitAt n xs
```

Jozef – maxSucet.scala

```
case class Trojica(index: Int,slice: Array[Int],suma: Int)

def maxSucet(list: Array[Int]): (Int, Array[Int]) = {
  val partialResult = (1 until list.length).flatMap(size=>{
      (0 to (list.length - size)).map(index => {
        val slice = list.slice(index, index + size)
        Trojica(index, slice, slice.sum)
      })
    })
    if (partialResult.nonEmpty) {
      val res = partialResult.maxBy(_.suma)
      if (res.suma>0) (res.suma,res.slice) else (0,Array())
    }
    else (0, Array())
  }
```

Prepísane do Haskellu

```
maxSucet :: [Int] -> (Int, [Int])
maxSucet list = if partialResult == [] || fst maxSumPair < 0 then (0, [])
                else maxSumPair
    where
      -- pomocná funkcia, ktorá vyrobí python slice zo zoznamu
      slice index size = [list !! sliceIndex | sliceIndex <- [index..index+size-1]]
      partialResult =
                [ (sum (slice index size), slice index size)
                  | size<-[1..length list], index <- [0..length list - size]
                ]                          -- dvojitý for-cyklus
      maxSumPair = maximumBy (comparing fst) partialResult
                                 -- maximum dvojíc podľa druhej súradnice...
```

Coursea kurz, M.Odersky

https://www.coursera.org/learn/progfun1