Deklaratív programozás, 2. gyakorlat (Elixir), 2021.09.21. -- 2019.09.23.

Az alábbi feladatok megoldására írjon olyan Elixir-függvényeket, amelyek megfelelnek a fejkommenteknek. Írjon mindegyikből több verziót. Gondolja át, melyik hatékonyabb és miért.

Használjon listajelölőt (komprehenziót), használja az Elixir és Erlang modulok függvényeit. A feladatok megoldásához használhatja a korábbi feladatokban definiált függvényeit.

A függvényeket az IEx-ben a modulnév megadásával kell meghívni (Mnév.fnév). Az alábbi példákban nem írjuk ki a modulnevet.

---

1. Összetett számok (javasolt: :lists.usort/1, listajelölő)

```
@spec osszetett(k::integer) :: xs::[integer]
# A 4..k*k közötti összetett számok növekvő listája, ismétlődés nélkül

osszetett(5) === [4, 6, 8, 9, 10, 12, 14, 15, 16, 18, 20, 21, 22, 24, 25]
```

---

2. Prímszámok (javasolt: osszetett/1, listajelölő)

```
@spec primek(k::integer) :: xs::[integer]
# A 2..k*k közötti prímszámok listája xs

primek(5) === [2, 3, 5, 7, 11, 13, 17, 19, 23]
```

---

3. Lista ismétlődő elemei (javasolt: Enum.zip/2, Enum.take/2, listajelölő)

```
@spec duplak(xs::[any]) :: ys::[any]
# ys az xs azon elemeinek listája, melyek egyenlők az őket követő elemmel

duplak([1, 2, 3])          === []
duplak([1, 1, 2, 3, 3, 3]) === [1, 3, 3]
```

---

4. Listaelemek különbözőségének vizsgálata (javasolt: (a) Enum.member?/2, (b) :lists.usort/1, (c) Enum.sort/1, Enum.zip/2, List.foldl/3)

```
@spec all_different(xs::[any]) :: b::boolean
# b igaz, ha az xs listában csupa különböző értékű elem van

all_different([1, 2, 3, 1]) === false
all_different([1, 2, 3])    === true
```

---

5. Lista összes elemének ellenőrzése (javasolt: (a) Enum.map/2, List.foldl/3, (b) List.foldl/3 map/2 nélkül, (c) rekurzióval, könyvtári függvények nélkül)

```
@spec all(p::(t::any -> boolean), xs::[t::any]) :: b::boolean
# b akkor és csak akkor igaz, ha p teljesül xs minden elemére

all(&is_atom/1, [:a, :b]) === true
all(&is_atom/1, [:a, 1])  === false
```

---

6. Platók hossza

Nevezzük platónak az egyenlő értékű elemek sorozatát egy listában

```
@spec platok_hossza(xs::[any]) :: phs::[{p::any, h::integer}]
# phs olyan {p, h} párok listája, amelyekben p a platót képező
# érték, h pedig e plató hossza (= azonos értékű elemeinek száma)

platok_hossza([:a,:a,:a,:b,:b,:d,:f,:f,:h]) === [{:a,3},{:b,2},{:d,1},{:f,2},{:h,1}]
```

---

7. Négyzetes mátrix (javasolt: listajelölő + String.length/1, String.pad/3, Integer.to_String/1, IO.write/1)

Nevezzük négyzetesnek a mátrixot, ha sorainak és oszlopainak száma is négyzetszám. Írjon egy függvényt a négyzetes mátrix előállítására, és egy másikat a kiíratására

```
    @type mx :: rs::[cs::[integer]]
    @spec negyzetes(k::integer) :: mss::mx
    # k^2 méretű mátrixok olyan k^4 méretű mátrixa mss, amelyben
    # az elemek értéke 1-től k^4-ig minden sorban felülről lefelé
    # haladva egyesével nő: a bal felső (1,1) indexű elem értéke 1,
    # alatta az (1,0) indexűé 2 stb., a (k^2,k^k) indexűé pedig k^4

    @spec matrix_ki(mss::mx) :: :ok
    # a negyzetes/1 függvény által előállított mátrix kiírása a
    # képernyőre úgy, hogy a mátrix minden sora új sorba kerüljön,
    # a számok pedig oszloponként balra legyenek illesztve, a
    # vezető 0-k helyén szóközzel

    negyzetes(2) === [[1, 5, 9, 13], [2, 6, 10, 14], [3, 7, 11, 15], [4, 8, 12, 16]]

    matrix_ki(negyzetes 2) == :ok
      1  5  9 13
      2  6 10 14
      3  7 11 15
      4  8 12 16
-------------------------------------------------------------------------------
 8. Mátrix részmátrixa (javasolt: listajelölő + (a) Enum.slice/3, (b) Enum.at/2

    @spec kozepe(mss::[[any]]) :: rss::[[any]]
    # rss az mss n*n-es négyzetes mátrix olyan (n/2)*(n/2) méretű részmátrixa,
    # mely az n/4+1. sor n/4+1. oszlopának elemétől kezdődik
    # négyzetes az olyan mátrix, amelynek n sora és n oszlopa van, ahol
    # az n>=4 négyzetszám

    kozepe([[:a,:b,:e,:f], [:c,:d,:g,:h], [:i,:j,:m,:n], [:k,:l,:o,:p]]) ===
      [[:d,:g],[:j,:m]]
-------------------------------------------------------------------------------
9.  Mátrix középső elemei (javasolt: (a) List.flatten/1, (b) Enum.at/2)

    @spec laposkozepe(mss::[[any]]) :: xs::[any]
    # Az xs lista az mss mátrix középső elemeinek listája
    def laposkozepe(mss), do: List.flatten(kozepe(mss))

    A megoldást valósítsa meg többféleképpen:
    a) lists:flatten/1 és kozepe/1 felhasználásával,
        Példa lists:flatten/1 használatára:
          lists:flatten([1,[2,3],[[[[4]]],[5,[6]]]]) === [1,2,3,4,5,6]
    b) Használja fel listanézetben a lists:nth/2 függvényt

    laposkozepe([[:a,:b,:e,:f], [:c,:d,:g,:h], [:i,:j,:m,:n], [:k,:l,:o,:p]]) ===
      [:d,:g,:j,:m]
-------------------------------------------------------------------------------
10. Részmátrix sor és oszlop elhagyásával (javasolt: listajelölő + Enum.at/2)

    @spec pivot(mss::[[any]],r::integer,c::integer) :: rss::[[any]]
    # rss az mss mátrix r-edik sorának és c-edik oszlopának elhagyásával áll elő

    pivot([[:a,:b,:e,:f], [:c,:d,:g,:h], [:i,:j,:m,:n], [:k,:l,:o,:p]], 1, 2) ===
      [[:a,:b,:f], [:i,:j,:n],[:k,:l,:p]]
-------------------------------------------------------------------------------
11. Mátrix transzponáltja

    Egy valódi mátrix transzponáltja sorainak és oszlopainak felcserélésével jön létre
    (a valódi mátrix minden sora egyforma hosszú)

    @spec transpose(mss::[[any]]) :: tss::[[any]]
    # az mss valódi mátrix transzpontáltja tss

    transpose([[:a,:b], [:c,:d], [:e,:f]]) === [[:a,:c,:e], [:b,:d,:f]]

----- $LastChangedDate: 2021-09-16 15:51:21 +0200 (cs, 16 szept 2021) $ ------
```