NZH-MINTAFELADATOK

A zéhában az alábbi Elixir függvények fordulhatnak elő, a programozási feladatokban ezeket használhatja:

Kernel.+/2, Kernel.-/2, Kernel.*/2, Kernel.//2, Kernel.rem/2, Kernel.abs/1, Kernel.<=/2, Kernel.</2, Kernel.>/2, Kernel.>=/2, Kernel.===/2, Kernel.!==/2, Kernel.++/2, Kernel.<>/2, Kernel.../2, Kernel.&/1, Kernel.elem/2, Kernel.tl/1, Kernel.length/2 Kernel.is_atom, Kernel.is_list/1, Enum.to_list/1, Enum.concat/1, Enum.uniq/1, Enum.map/2, Enum.filter/2, Enum.reverse/1, List.flatten/1}

Az FP nagyzárthelyi 1. részében az alábbihoz hasonló "mitírki" feladatot kell megoldani Elixir nyelven.

---

1) Mi az X változó értéke? A deklarációk függetlenek egymástól. (9 pont)

```
x = { {:some, 6+1}, [?A] ++ 'B', length([&Kernel.elem/2]), (&Kernel.>/2).(2,3) }

x = Enum.map( Enum.concat( ['xy7','a5','8zx'] ), fn(x) -> x === ?x end )

f = &../2;
x = ( for x <- f.(1, 2), y <- f.(3, 1), x <= y, do: f.(x, y) )
    |> Enum.map(&Enum.to_list/1) |> Enum.concat |> Enum.uniq
```

Öt hasonló kis kérdés lesz a zéhában.

Az FP nagyzárthelyi 2. részében az alábbiakhoz hasonló programozási feladatokat kell megoldani Elixir nyelven.

A feladat két, egymásra épülő részfeladatból áll: először egy segédfüggvényt, majd ezt felhasználva egy, a teljes feladatot megoldó függvényt kell megírni.

2a1) Az M1 feladat segédeljárása (22 pont)

Írjon szigetek néven olyan függvényt, amely visszaadja egy lista pozitív elemekből álló, folytonos, semelyik irányba ki nem terjeszthető részlistáinak listáját.

```
@spec szigetek(xs::[integer]) :: rss::[[integer]]
# Az xs pozitív elemekből álló, folytonos, maximális
# hosszúságú részlistáinak listája az rss lista.
```

Példák:

```
szigetek([]) === []
szigetek([-1,0,-2,-3,0,0]) === []
szigetek([1,2,3,0,0,4,5,6]) === [[1,2,3],[4,5,6]]
szigetek([2,1,3,0,0,7,6,-1,-3,0,7,5,6]) === [[2,1,3],[7,6],[7,5,6]]
```

2b1) A teljes M1 feladat (12 pont)

Írjon szfold néven olyan függvényt (javaslat: a szigetek/1 függvény felhasználásával), amely egy negatív számokat nem tartalmazó egészlista szárazföldjeinek hosszából és legmagasabb pontjából álló párok listáját adja eredményül! Szárazföldnek a csupa pozitív számból álló, legalább egyelemű, maximális hosszúságú, folytonos részlistát nevezzük.

A zéhában specifikált szigetek/1 függvényt akkor is használhatja, ha nem írta meg. De olyan kódot is írhat, amelyik nem alkalmazza a szigetek/1 függvényt.

```
@spec szfold(zs::[integer]) :: ys::[ {hossz::integer, csucs::integer} ]
# A negatív számokat nem tartalmazó zs egészlistában előforduló szárazföldek
# hosszát és legmagasabb pontját leíró {hossz, csucs} párok listája ys.
```

```
    Példák:

    szfold([0,1,0,3,4,0,0,1]) === [{1,1}, {2,4}, {1,1}]
    szfold([0,0,30,4,10,0,0,100]) === [{3,30}, {1,100}]
    szfold([0,0,0]) === []
```

2a2) Az M2 feladat segédeljárása (22 pont)

```
    Írjon dec2hex néven olyan függvényt, amely kiszámolja egy decimális számnak
    a 0..9 és A..F karakterek sztringként ábrázolt hexadecimális alakját. A
    füzér utolsó karaktere legyen a legkisebb, egyes helyiértékű
    számjegy. Ügyeljen a 0 helyes kezelésére!

    @spec dec2hex(d::integer) :: h::charlist
    # A d decimális szám ?0-?9 és ?A-?F karakterek füzéreként ábrázolt
    # hexadecimális megfelelője h.

    Példák:

    dec2hex(0) === '0'
    dec2hex(7) === '7'
    dec2hex(14) === 'E'
    dec2hex(75) === '4B'
```

2b2) A teljes M2 feladat (12 pont)

```
    Írjon d2hfa néven olyan függvényt (javaslat: a dec2hex/1 függvény
    felhasználásával), amely a

        @type fa(elem) :: :e | {:n, elem, fa(elem), fa(elem)}

    típusspecifikációval megadott, fa(integer) típusú bemenő argumentumát
    fa(charlist) típusúvá alakítja úgy, hogy minden :n csomópontban található
    egész számot lecserél a füzérként ábrázolt hexadecimális megfelelőjére.

    A zéhában specifikált dec2hex/1 függvényt akkor is használhatja, ha nem írta
    meg. De olyan kódot is írhat, amelyik nem alkalmazza a dec2hex/1 függvényt.

    @spec d2hfa(fa0::fa(integer)) :: fa::fa(charlist)
    # A fa0 decimális számokat tartalmazó fa hexadecimális számokat
    # tartalmazó megfelelője fa.

    Példák:

    d2hfa(:e) === :e
    d2hfa({:n, 14, :e, :e}) === {:n, 'E', :e, :e}
    d2hfa({:n, 75, {:n, 200, :e, :e}, {:n, 35, :e, :e}})
                          === {:n, '4B', {:n, 'C8', :e, :e}, {:n,'23', :e, :e}}
```

----- $LastChangedDate: 2022-10-12 15:09:36 +0200 (sze, 12 okt 2022) $ ------