

## Vizsga feladat - 2023.01.10.

**Kategória:**

Vizsgafeladatok

**Elérhető:**

2023. 01. 10. 12:20

**Pótolható határidő:**

**Végső határidő:**

2023. 01. 10. 13:50

**Maximum beküldési próbálkozások száma:**

Korlátlan

**Kiírta:**

Bozó István

**Leírás:**

### Előzetes tudnivalók

Használható segédanyagok:

- Haskell könyvtárak dokumentációja,
- Hoogle,
- a tárgy honlapja, és a
- Haskell szintaxis összefoglaló.

Ha bármilyen kérdés, észrevétel felmerül, azt a felügyelőknek kell jelezni, **nem** a diáktársaknak!

**FONTOS:** A megoldásban legalább az egyik (tetszőleges) függvényt rekurzívan kell megadni. Azaz a vizsga csak akkor érvényes, ha az egyik feladatot rekurzív függvénnyel adtátok meg és az helyes megoldása a feladatnak. A megoldást akkor is elfogadjuk, ha annak egy segédfüggvénye definiált rekurzívan. A könyvtári függvények ( `length` , `sum` , stb.) rekurzív definíciója nem fogadható el rekurzív megoldásként.

A feladatok tetszőleges sorrendben megoldhatóak. A pontozás szabályai a következők:

- Minden teszten átmenő, a feladat kikötéseinek megfelelő megoldás érhet teljes pontszámot.
- Egy feladatra beadott funkcionálisan hibás (valamelyik teszteseten megbukó) megoldás nem ér pontot.
- Fordítási hibás kód esetén a teljes vizsga 0 pontos.

Ha hiányos/hibás részek lennének a feltöltött megoldásban, azok kommentben szerepeljenek.

*Tekintve, hogy a tesztesetek, bár odafigyelés mellett íródnak, nem fedik le minden esetben a függvény teljes működését, határozottan javasolt még külön próbálgatni a megoldásokat beadás előtt vagy megkérdezni a felügyelőket!*

A *Visual Studio Code Haskell Syntax Highlighting* bővítmény a csatolt fájlok között megtalálható.

Telepítés:

- Bővítmények megnyitása bal oldalt (4 kicsi négyzet) ( `Ctrl + Shift + X` )
- `...` a megnyíló ablak jobb felső sarkában
- `Install from VSIX...` , majd a letöltött állomány kitallózása

## Feladatok

### 1. Páratlanok (2 pont)

Definiáld a `sumOdds :: Integral a => [a] -> a` függvényt, amely egy véges lista páratlan számait összeadja!

```
sumOdds [] == 0
sumOdds [1 :: Int,2,3,4,5] == 9
sumOdds [2,4,6,8,10,12,14,16] == 0
sumOdds [100] == 0
sumOdds [3] == 3
sumOdds [(-13) :: Integer,3,9,7,11,113,-101] == 29
sumOdds [1001,2002,3003,4,6,8,5005,10000,989,1,1] == 10000
```

## 2. Közlekedés (2 pont)

Egy város tömegközlekedését buszok és villamosok szolgálják ki. Sok járat van, minden járat sok, de véges számú megállóval rendelkezik, illetve egy megállóban akár több járat is megállhat. Definiáld a `Line` saját típust, amellyel a város tömegközlekedését reprezentáljuk. A típusnak van két konstruktora:

- `Tram :: Integer -> [String] -> Line`
- `Bus :: Integer -> [String] -> Line`

Mindkét esetben az `Integer` típusú érték a járatszámot, a `[String]` pedig a járat megállóinak a neveit tartalmazza.

Definiáld a `whichBusStop :: String -> [Line] -> [Integer]` függvényt, amely megadja, hogy a (véges) tömegközlekedési hálózat mely *busz*járata(i) áll(nak) meg egy adott megállóban!

```
null (whichBusStop "Alma utca" [])
whichBusStop "József utca" [Bus 111 ["Alma utca", "Károly utca", "József út", "Halom utca", "Mária utca"], Tram 10 ["József út", "Karinthy utca",
null (whichBusStop "Károly utca" [Tram 20 ["Alma utca", "Károly utca", "József út", "Halom utca", "Mária utca"], Tram 10 ["József út", "Karinthy u
whichBusStop "Mária utca" [Bus 111 ["Alma utca", "Károly utca", "József út", "Halom utca", "Mária utca"], Bus 10 ["József út", "Karinthy utca", "R
take 20 (whichBusStop "Nemes út" (Bus 5 ["József út", "Karinthy utca", "Róbert utca", "Templom tér", "Nemes út", "Őrház"] : Bus 10 ["József út", "
```

## 3. Szólánc (3 pont)

Definiáld a `wordChain :: String -> String` függvényt, amely egy véges szöveg szavait `String` -gé összefűzi egy darab szóközzel elválasztva a szavakat úgy, hogy kihagyja azokat a szavakat, amelyek nem a sorozatba tartozó előző szó utolsó karakterével kezdődnek! (Az első szó mindig benne van a láncban.)

```
null (wordChain [])
wordChain "alma barack ananasz zold" == "alma ananasz zold"
wordChain "szilva alma galamb barna alom malom maszat kalap" == "szilva alma alom malom maszat"
wordChain "kalap pisztacia alak kaszt torta" == "kalap pisztacia alak kaszt torta"
wordChain "lap alma liszt por ripsz szalma szilva szoja zacc fel" == "lap por ripsz zacc"
wordChain "Alma ALOM aloM Mesz zoLD DAL LOP POh tol kap kalap haskell" == "Alma aloM Mesz zoLD DAL LOP POh haskell"
```

## 4. Típusnév (3 pont)

Döntsd el egy karakterláncról, hogy típus-e, azaz teljesül-e rá, hogy csak nagybetűvel kezdődik és minden további eleme csak kisbetű, nagybetű, számjegy, vagy az alulvonás karakter. (A `String` -ről feltehető, hogy véges.)

```
isType :: String -> Bool

isType "Alma"
isType "A"
isType "T"
not (isType "alma")
not (isType "a")
not (isType "_")
not (isType "9")
isType "Alma_123"
isType "B1C2"
not (isType "Alma 123")
not (isType "_amlA")
not (isType "10 January")
not (isType "")
```

## 5. Két lehetőség (2 pont)

Definiáld a `partialApply :: (a -> a -> b) -> a -> (a -> b, a -> b)` függvényt, amely a paraméterül kapott függvényre parciálisan alkalmazza a kapott értéket a két lehetséges módon! (A sorrendjük nem számít!)

```
(\(f,g) -> (f 10, g 15)) (partialApply (+) 5) == (15,20)
( \(f,g) -> (f 4, g 5)) (partialApply (^) 3) == (81,125) || ( \(f,g) -> (f 4, g 5)) (partialApply (^) 3) == (64,243)
( \(f,g) -> (f [4,5], g [6,7,8])) (partialApply (++) [1,2,3]) == ([1,2,3,4,5],[6,7,8,1,2,3]) || ( \(f,g) -> (f [4,5], g [6,7,8])) (partialApply (++)
( \(f,g) -> (f 'o', g 't')) (partialApply (,) 'k') == (('k','o'),('t','k')) || ( \(f,g) -> (f 'o', g 't')) (partialApply (,) 'k') == (('o','k'),('k'
```

## Git tároló

A jelszóvédett feladatok esetében a 'git push' nem használható.

Útvonal:

https://tms.inf.elte.hu/git/4030/asacdn/w50040313d217c036e49eabdf4

Használat:

git clone https://tms.inf.elte.hu/git/4030/asacdn/w50040313d217c036e49eabdf4

Megoldás



Név:

Vizsga.zip

Feltöltés ideje:

2023. 01. 10. 13:39

Értékelés:

5

Státusz:

Elfogadva

Feltöltések száma:

2

Értékelte:

Bense Viktor

Megjegyzések:

4 pont beugró

12 pont gyak

+3 pont nagybead

----

19 pont

Automatikus tesztelés eredményei

Elért pontszám: 12/12 pont.

Teszteken sikeresen átmenő definíciók: sumOdds, whichBusStop, wordChain, isType, partialApply.

Minden teszt sikeresen lefutott.

Mellékelte fájlok



haskell-syntax-highlighting-3.6.0.vsix

