11. gyak

A megoldáshoz ne használjuk a `Data.Maybe` modult, kizárólag azokat a függvényeket, amelyek a `Prelude`-ból és a `Data.List`-ből elérhetőek.

`Maybe`-s segédfüggvények

Adjuk meg azt a függvényt, amely eldönti egy `Maybe` típusú értékről, hogy az `Just`-tal lett-e konstruálva!

```
isJust :: Maybe a -> Bool
```

Adjuk meg azt a függvényt, amely egy `Just`-ból kiszedi a benne lévő értéket, `Nothing`-ra pedig hibát dob! (Ehhez használjuk az `error` függvényt!)

```
fromJust :: Maybe a -> a
```

Adjuk meg azt a függvényt, amely egy `Maybe`-ket tartalmazó listából kiválogatja a `Just`-okat, és egy olyan listát ad vissza, amelyben a `Just`-ok által tárolt értékek szerepelnek (`Nothing`-kat pedig elhagyja)! Példa: `catMaybes [Just 3, Nothing, Just 1] == [3,1]`

```
catMaybes :: [Maybe a] -> [a]
```

Adjuk meg azt a függvényt, amely `Maybe` típusba képező függvényt alkalmaz egy lista elemeire, majd kiválogatja az így kapott listából a `Just`-okat, és azokból egyesével kiszedi a bennük lévő értékeket!

```
mapMaybe :: (a -> Maybe b) -> [a] -> [b]
```

Adatbázis

A következő feladatokban egy adatbázis Haskelles reprezentációján kell majd különböző műveleteket implementálni.

Reprezentáció

Egy felhasználót a felhasználónevével és a jelszavál fogunk azonosítani.

```
type Username = String
type Password = String
```

Minden felhasználóhoz fog tartozni egy jogosultsági szint.

```
data Privilege = Simple | Admin
  deriving (Eq, Show)
```

Azt, hogy ki van bejelentkezve, cookie-k segítségével fogjuk számontartani.

```
data Cookie = LoggedOut | LoggedIn Username Privilege
  deriving (Eq, Show)
```

Az adatbázisban egy bejegyzés három dolgot fog tartalmazni: a felhasználóhoz tartozó jelszót, jogosultsági szintet és a felhasználó barátait.

```
data Entry = Entry Password Privilege [Username]
  deriving (Eq, Show)
```

Végül pedig az adatbázist kulcs-érték párok listájával fogjuk reprezentálni, ahol a kulcsok a felhasználónevek, az értékek pedig bejegyzések lesznek.

```
type Database = [(Username, Entry)]
```

Példa adatok

A következő néhány példa adaton fognak futni a tesztek, úgyhogy majd ezeket is másoljátok be a megoldásba.

```
richard, charlie, carol, david, kate :: (Username, Entry)
richard = ("Richard", Entry "password1" Admin  ["Kate"])
charlie = ("Charlie", Entry "password2" Simple ["Carol"])
carol   = ("Carol",   Entry "password3" Simple ["David", "Charlie"])
david   = ("David",   Entry "password4" Simple ["Carol"])
kate    = ("Kate",    Entry "password5" Simple ["Richard"])

testDB :: Database
testDB = [ richard, charlie, carol, david, kate ]

testDBWithoutCarol :: Database
testDBWithoutCarol =
  [ ("Richard", Entry "password1" Admin  ["Kate"])
  , ("Charlie", Entry "password2" Simple [])
  , ("David",   Entry "password4" Simple [])
  , ("Kate",    Entry "password5" Simple ["Richard"])
  ]
```

Szelektorfüggvények

Adjuk meg azt a függvényt, amely kiválasztja egy bejegyzésből a felasználó jelszavát!

```
password :: Entry -> Password
```

Adjuk meg azt a függvényt, amely kiválasztja egy bejegyzséből a felasználó jogosultsági szintjét!

```
privilege :: Entry -> Privilege
```

Adjuk meg azt a függvényt, amely kiválasztja egy bejegyzésből a felhasználó barátait!

```
friends :: Entry -> [Username]
```

Adatbázis műveletek

Adjuk meg azt a függvényt, amely kap egy felhasználónevet, egy jelszót, és egy adatbázis bejegyzést, és ha bejegyzésben lévő jelszó megegyezik a kapott jelszóval, akkor egy `LoggedIn` cookie-t ad vissza a felhasználó számára, egyébként pedig egy `LoggedOut` cookie-t.

```
mkCookie :: Username -> Password -> Entry -> Cookie
```

Adjuk meg azt a függvényt, amely bejelentkeztet egy felhasználót! A felhasználót úgy jelentkeztetjük be, hogy ha szerepel az adatbázisban, és helyes a megadott jelszó, akkor visszaadunk egy cookie-t a felhasználónevével és a jogosultsági szintjével, ha pedig nem szerepel az adatbázisban, vagy a megadott jelszó nem egyezik az adatbázisban lévővel, akkor kijelentkezteve tartjuk. **Segítség**: Használjuk a `lookup`, `maybe` és `mkCookie` függvényeket!

```
login :: Username -> Password -> Database -> Cookie
```

Adjuk meg azt a függvényt, amely a következőképpen működik. Kap egy törlendő felhasználót, és egy adatbázisban szereplő kulcs-érték párt. Ha a kulcs éppen a törlendő felhasználó, akkor térjen vissza `Nothing`-gal. Egyébként pedig törölje a bejegyzésben lévő barátok közül közül a törlendő felhasználót, és az eredményt egy `Just`-ba csomagolva adja vissza.

```
updateEntry :: Username -> (Username, Entry) -> Maybe (Username, Entry)
```

Adjuk meg azt a függvényt, amely töröl egy felhasználót az adatbázisból! A törlést csak akkor végezzük el, ha az azt kérő felhasználó (akinek az adatai a kapott cookie-ban vannak) `Admin` jogosultsággal rendelkezik. **Segítség**: Használjuk az `updateEntry` és `mapMaybe` függvényeket!

```
deleteUser :: Cookie -> Username -> Database -> Database
```

```
isJust (Just 5)  == True
isJust (Just []) == True
isJust Nothing   == False

fromJust (Just 5)  == 5
fromJust (Just []) == []

catMaybes []                          == []
catMaybes [Nothing]                   == []
catMaybes [Nothing, Nothing]          == []
catMaybes [Nothing, Just 5, Nothing] == [5]
catMaybes [Just 1, Just 5, Nothing]  == [1,5]
catMaybes [Nothing, Just 5, Just 1]  == [5,1]
catMaybes [Just 0, Just 5, Just 1]   == [0,5,1]
```

```
mapMaybe Just [1..10] == [1..10]
mapMaybe (const Nothing) [1..10] == []
mapMaybe (\n -> if odd n then Just n else Nothing) [1..10] == [1,3..10]

password (snd richard)  == "password1"
privilege (snd richard) == Admin
friends (snd carol)     ==  ["David", "Charlie"]

mkCookie "Richard" "password1" (snd richard) == LoggedIn "Richard" Admin
mkCookie "Charlie" "password2" (snd charlie) == LoggedIn "Charlie" Simple
mkCookie "Charlie" "password2" (snd richard) == LoggedOut
mkCookie "Richard" "wrong_pw" (snd richard) == LoggedOut

login "Kate"    "forgot"    testDB == LoggedOut
login "Kate"    "password5" testDB == LoggedIn "Kate" Simple
login "Richard" "password1" testDB == LoggedIn "Richard" Admin

updateEntry "Kate"  kate    == Nothing
updateEntry "Kate"  richard == Just ("Richard", Entry "password1" Admin [])
updateEntry "David" carol   == Just ("Carol", Entry "password3" Simple ["Charlie"])

deleteUser LoggedOut "Kate" testDB                      == testDB
deleteUser (LoggedIn "Carol" Simple) "Kate" testDB      == testDB
deleteUser (LoggedIn "Richard" Admin) "Jonathan" testDB == testDB
deleteUser (LoggedIn "Richard" Admin) "Carol" testDB    == testDBWithoutCarol
```