# Webes és mobil programozás F#-ban 4. gyakorlat

## Berezvai Dániel jegyzete <http://elte.3ice.hu/>

Megint letöröltek mindent. Előző órán a pipe-ig jutottak el → ELTE\F#\2\Lesson2.fs.md

Admin jelszó: pcuser – Cloud14

## Típusrövidítés

**type** coord **=** int **\*** int

### Például

**type** x **=** int

**type** y **=** int

**type** coordinate **=** x **\*** y

### Használat

Kell a zárójel

**>** **((**0**,**0**):**coordinate**);;**

**val** it **:** coordinate **=** **(**0**,** 0**)**

**type** name **=** string

**type** phone **=** int

**type** contact **=** name **\*** phone

**>** **((**"John Smith"**,**21234567**):**contact**);;**

**val** it **:** contact **=** **(**"John Smith"**,** 21234567**)**

**>** snd it**;;**

**val** it **:** phone **=** 21234567

## Rekord

**>** **type** customer **=**

 **{**Name**:** string**;**

 Salary**:** int**;}**

**let** worker1 **=** **{**Name **=** "Andrew Smith"**;**

 Salary **=** 1000**};;**

**type** customer **=**

 **{**Name**:** string**;**

 Salary**:** int**;}**

**val** worker1 **:** customer **=** **{**Name **=** "Andrew Smith"**;**

 Salary **=** 1000**;}**

### Módosítás

**>** **type** customer **=**

 **{**Name**:** string**;**

 Salary**:** int**;}**

**let** worker1 **=** **{**Name **=** "Andrew Smith"**;**

 Salary **=** 1000**}**

**let** worker2 **=** **{**worker1 **with** Salary **=** 2500**};;**

**type** customer **=**

 **{**Name**:** string**;**

 Salary**:** int**;}**

**val** worker1 **:** customer **=** **{**Name **=** "Andrew Smith"**;**

 Salary **=** 1000**;}**

**val** worker2 **:** customer **=** **{**Name **=** "Andrew Smith"**;**

 Salary **=** 2500**;}**

Nem lehet, csak másolni.

Lehet ugyan változtathatóvá tenni a mezőket, de akkor imperatív, nem marad funkcionális a rekord.

## Diszkriminált unió

**>** **type** Literature **=**

 **|** Pol

 **|** SciFi

 **|** Crime**;;**

**type** Literature **=**

 **|** Pol

 **|** SciFi

 **|** Crime

### Menü

**>** **type** Name **=** string

**type** Cost **=** int

**type** Menu **=**

 **|**Soup **of** Name **\*** Cost

 **|**MainCourse **of** Name **\*** Cost

 **|**Dessert **of** Name **\*** Cost**;;**

**type** Name **=** string

**type** Cost **=** int

**type** Menu **=**

 **|** Soup **of** Name **\*** Cost

 **|** MainCourse **of** Name **\*** Cost

 **|** Dessert **of** Name **\*** Cost

### Funkció rá

**let** getSpeedOfVehicle v **=**

 **match** v **with**

 **|** Car **(\_,** s**)** **->** s

 **|** Bus **(\_,** **\_,** s**)** **->** s

 **|** Tram **(\_,** **\_,** s**)** **->** s

getSpeedOfVehicle**(**Car**(**1.0**,** 100**));;**

getSpeedOfVehicle**(**Bus**(**0.8**,** 60**));;**

getSpeedOfVehicle**(**Tram**(**0.4**,** 80**));;**

## Feladat: Mobil, Telefon, PC diszkriminátor, tulajdonságokkal. Tömegközlekedés diszkriminátor autó, villamos, busz, stb. +Mintaillesztés.

**type** FuelUsage **=** int

**type** PassengerCount **=** int

**type** TopSpeed **=** int

**type** TransportTool **=**

 **|**Car **of** FuelUsage **\*** PassengerCount **\*** TopSpeed

 **|**Bus **of** FuelUsage **\*** PassengerCount **\*** TopSpeed

 **|**Tram **of** FuelUsage **\*** PassengerCount **\*** TopSpeed**;;**

### Függvény, ami illeszt erre

**let** getSpeedOfVehicle v **=**

 **match** v **with**

 **|** Car **(\_,** s**)** **->** s

 **|** Bus **(\_,** **\_,** s**)** **->** s

 **|** Tram **(\_,** **\_,** s**)** **->** s

## Használata

**>** getSpeedOfVehicle**(**Car**(**1.0**,** 100**));;**

**val** it **:** TopSpeed **=** 100

**>** getSpeedOfVehicle**(**Bus**(**0.8**,** 60**,** 60**));;**

**val** it **:** TopSpeed **=** 60

**>** getSpeedOfVehicle**(**Tram**(**0.4**,** 100**,** 80**));;**

**val** it **:** TopSpeed **=** 80

## Beadandó

Beadandónál a főmenü rendszert fogjuk diszkriminált unióba összefoglalni. Valószínűleg nem lesz plusz tulajdonsága.

## Elnevezési konvenciók

Típusátnevezés kisbetű, diszkriminált unió és rekord nagybetű (CamelCase).

## Két hét van szünetig

Utána fejlesztünk.

Amiről szó lesz: Imperatív, OOP, aszinkron folyamatok/szálak. Már az indítóprogram aszinkron hívással kezdődik.

## Rekurzív diszkriminatív unió

### Peano számok

0= Zero

1= Succ Zero

2= Succ Succ Zero

## F#-ban

**type** Peano **=**

 **|** Zero

 **|** Succ **of** Peano

**let** **rec** evaluatePeano p**=**

 **match** p **with**

 **|**Zero**->**0

 **|**Succ p**->**1**+**evaluatePeano p**;;**

**let** one **=** Succ Zero**;;**

evaluatePeano one**;;**

**type** Peano **=**

 **|** Zero

 **|** Succ **of** Peano

**val** evaluatePeano **:** Peano **->** int

**val** one **:** Peano **=** Succ Zero

**val** it **:** int **=** 1

## Feladat: Ez alapján a Lista megírása

**type** List **=**

 **|** Nil

 **|** Cons **of** int **\*** List**;;**

### Lehet rá írni sum vagy produktumot:

**let** **rec** sumList l **=**

 **match** l **with**

 **|** Nil **->** 0

 **|** Cons **(**i**,** l**) ->** i **+** sumList l**;;**

## Feladat 2: Fa

Jobb fa bal fa vagy levél egy értékkel:

**type** Tree **=**

 **|** Leaf

 **|** Node **of** Tree **\*** Tree**;;**

Vagy jobb fa bal fa középen értéket tárol:

**type** Tree **=**

 **|** Leaf

 **|** Node **of** int **\*** Tree **\*** Tree**;;**

### Rendezetlen fát lehet rendezni…

Köszönöm nem:)

## Az Option típus is diszkriminált unió

**type** T Option **=**

 **|**None

 **|**Some **of** T

Gyakorlat vége.