

Examen Programare Functionala

1 februarie 2023, ora 12:00

V2

DETALII ORGANIZATORICE

Mediul de lucru. La începutul examenului, va rugam să verificați conexiunea la internet și funcționarea interpretorului de Haskell. În toate laboratoarele limbajul Haskell este instalat în Windows. Puteti lansa interpretorul cu comanda `ghci` în Command Prompt. Dacă doriți să lucrați în VSCode, trebuie să vă asigurați că NU sunt extensii instalate pentru Haskell.

Atenție! În timpul examenului, accesul la internet și folosirea telefoanelor mobile sunt interzise! Orice încercare de fraudă va fi sancționată conform regulamentului FMI!

Materiale ajutatoare. Puteti avea ca materiale ajutătoare, printate sau în format electronic, suporturile de curs, de laborator, și notitele voastre (de exemplu, rezolvarile voastre de la laboratoare). Înainte de începerea examenului, va rugam să descărcați materialele ajutătoare în format electronic pe calculatorul pe care o să susțineți examenul.

Denumire foldere/fisiere. Va rugam să creați un director cu numele

Director: `TEST_PF2022_NumePrenume`

și să lucrați numai în acest director pe toată durata examenului. În acest director, creați un subdirector cu numele **MATERIALE**, care va conține materialele ajutătoare.

Fiecare problemă va fi rezolvată într-un fișier separat. Fiecare fișier va avea numele

Fișier: `Grupa_Varianta_NumePrenume_NrProblema.hs`

Varianta trebuie să fie V1 sau V2, iar NrProblema trebuie să fie P1, P2 sau P3.

De exemplu, `244_V1_PopescuDan_P1.hs`.

Încărcarea rezolvarilor. La sfârșitul examenului se va permite conectarea la internet pentru încărcarea rezolvarilor. Înainte să vă conectați la internet la sfârșitul examenului, va rugam să solicitați prezența unui supraveghetor, care va urmări încărcarea rezolvării.

Fiecare student va încărca câte un fișier diferit cu rezolvarea pentru fiecare problemă, la linkul corespunzător problemei, care va fi pe foaia cu subiecte.

Atenție! Fiecare student poate încărca un singur fișier pentru fiecare problemă. Dacă un student încarcă mai multe fișiere pentru o problemă, testul este anulat!

Linkuri pentru incarcarea rezolvarilor.

24 - P1 <https://www.dropbox.com/request/3FoDeiuZOKufmN4GgSPA>

24 - P2 <https://www.dropbox.com/request/872ABzNtLztGZ9b96KUe>

24 - P3 <https://www.dropbox.com/request/8Dw8k6VwHRf1dVBhjqGE>

252 - P1 <https://www.dropbox.com/request/GXhStF84ZpZGVjlLsg9>

252 - P2 <https://www.dropbox.com/request/SAJliy0vi31otOVhShtI>

252 - P3 <https://www.dropbox.com/request/Pg0O5yHX8YnXGdYVs60l>

Restantieri - P1 <https://www.dropbox.com/request/wQDUbx5J8pklDIzeBFT9>

Restantieri - P2 <https://www.dropbox.com/request/mJtJjEfvYw1eMUQAZUhY>

Restantieri - P3 <https://www.dropbox.com/request/MiMgwfc1PVTJswXsmuQV>

Rezolvarea subiectelor. Va rugam sa cititi cu atentie enunturile inainte de a face rezolvările. Eventualele intrebari i le puteti adresa supraveghetorului din sala. Puteți importa biblioteci, puteți folosi orice funcție predefinită și puteți scrie oricâte funcții ajutătoare doriți.

Se dau punctaje parțiale, deci încercați să scrieți funcții auxiliare pentru etape diferite de rezolvare a unei probleme și scrieți comentarii clarificatoare dacă este cazul.

SUBIECTE

P1 [2pct]

Se dau următoarele:

Un tip de date `Prop` ce reprezinta expresii booleene continand variabile, doua constante pentru `true` (T) si `false` (F) si operatorii de conjunctie (`&`) si disjunctie (`|`).

```
data Prop = V String | T | F | And Prop Prop | Or Prop Prop
  deriving (Show, Eq)
```

O clasa de tipuri `Operations` ce contine o functie de simplificare.

```
class Operations exp where
  simplify :: exp -> exp
```

Sa se scrie o instanta a clasei `Operations` pentru tipul de date `Prop`, astfel incat operatia de simplificare sa efectueze calculele booleene cu `true` si `false`, mai exact sa faca simplificarile

- `p && T = T && p = p`
- `F && p = p && F = F`
- `p || T = T || p = T`
- `p || F = F || p = p`

Puteti testa solutia pe urmatoarele expresii:

```
prop1 = And (Or (V "p") (V "q")) T
prop2 = Or prop1 (V "r")
prop3 = Or (And F (V "p")) (V "q")
prop4 = And prop3 (V "q")
```

P2 [3 pct]

Să se transforme un sir de caractere dupa codificarea urmatoare:

- literele mari se transforma in litere mici
- literele mici raman neschimbate
- cifrele se tranforma in '-'
- restul caracterelor raman neschimbate

Daca va ajuta, in rezolvare puteti folosi functiile `isUpper`, `isLower`, `toUpper`, `toLowerCase`, `isAlphaNum`, din pachetul `Data.Char`.

Sa se rezolve problema in doua moduri (o solutie fara monade si o solutie cu monade).

Exemple:

"Ana,2" -> "ana,-"

"/,," -> "/,,"

P3 [1 pct]

Se da tipul de date

```
newtype WriterM a = MW {getMW :: (Either String a, String)}
```

Sa se scrie instanta completa a clasei `Monad` pentru tipul `WriterM` astfel incat sa pastreze proprietatea de monada. Nu este nevoie sa faceti instante si pentru clasele `Applicative` si `Functor`.

Puteti folosi pentru testare urmatoarea expresie

```
testWriterM :: WriterM Int
testWriterM = ma >>= k
  where
    ma =
      MW (Right 7, "ana are mere ")
    k x =
      MW (if x `mod` 2==0 then Right x else Left "", "si pere!" )
```