Examen Programare Functionala

1 februarie 2023, ora 10:00

V2

DETALII ORGANIZATORICE

Mediul de lucru. La începutul examenului, va rugam să verificati 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. Daca doriti sa lucrati in VSCode, trebuie sa va asigurati ca 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, si notitele voastre (de exemplu, rezolvarile voastre de la laboratoare). Înainte de începerea examenului, va rugam sa descărcati materialele ajutătoare în format electronic pe calculatorul pe care o sa sustineti examenul.

Denumire foldere/fisiere. Va rugam sa creati un director cu numele

Director: TEST_PF2022_NumePrenume

și sa lucrati numai în acest director pe toată durata examenului. In acest director, creati un subdirector cu numele **MATERIALE**, care va conține materialele ajutătoare.

Fiecare problema va fi rezolvata intr-un fisier separat. Fiecare fișier va avea numele

Fisier: Grupa_Varianta_NumePrenume_NrProblema.hs

Varianta trebuie sa fie V1 sau V2, iar NrProblema trebuie sa fie P1, P2 sau P3.
De exemplu, 244_V1_PopescuDan_P1.hs.

Incarcarea rezolvarilor. La sfârșitul examenului se va permite conectarea la internet pentru încărcarea rezolvărilor. Înainte a va conecta la internet la sfarsitul examenului, va rugam sa solicitati prezența unui supraveghetor, care va urmări încarcarea rezolvării.

Fiecare student va încărca cate un fișier diferit cu rezolvarea pentru fiecare problema, la linkul corespunzator problemei, care va fi pe foaia cu subiecte.

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

Linkuri pentru incarcarea rezolvarilor.

23 - P1 https://www.dropbox.com/request/9g4eZVTWFNeNWstvyiay

23 - P2 https://www.dropbox.com/request/s8LjS1it2NUV4Q247cuB

23 - P3 https://www.dropbox.com/request/9svjXX3XJ7zUaBlZRREs

251 - P1 https://www.dropbox.com/request/VXc3TnPOx4rBIbNdkDTD

251 - P2 https://www.dropbox.com/request/36LyEgYij0Gpty1LiwD8

251 - P3 https://www.dropbox.com/request/oD0UXBpfp6CfNlAlSwQH

Rezolvarea subiectelor. Va rugam sa cititi cu atenție enunturile înainte 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 `Expr` ce reprezinta expresii aritmetice continand variabile, valori intregi si operatorii de adunare si inmultire.

```
data Expr = Var String | Val Int | Plus Expr Expr | Mult Expr Expr
            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 clase `Operations` pentru tipul de date `Expr`, astfel incat operatia de simplificare sa efectueze calculele cu valorile 0 si 1, mai exact sa faca simplificarile:

- a + 0 = 0 + a = a
- a * 0 = 0 * a = 0
- a * 1 = 1 * a = a

Puteti testa solutia pe urmatoarele expresii:

```
ex1 = Mult (Plus (Val 1) (Var "x")) (Val 1)
ex2 = Plus ex1 (Val 3)
ex3 = Plus (Mult (Val 0) (Val 2)) (Val 3)
ex4 = Mult ex3 (Val 5)
```

P2 [3 pct]

Să se scrie o funcție care transformă un text (șir de caractere) în varianta lui din limba păsărească.
Transformarea unui caracter se va face astfel:

- dacă caracterul este o consoana, se adaugă un 'P' si consoana din nou, e.g., 'm' -> "mPm"
- dacă caracterul nu este o consoana, ramane nemodificat, e.g., 'a' -> "a"

Daca va ajuta, puteti folosi functia `isAlpha` din pachetul `Data.Char`.

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

Exemple:
"Mi-e foame" -> "MPMi-e fPfoamPme"
"Ana" - "AnPna"

P3 [1 pct]

Se da tipul de date

```
newtype ReaderM env a = ReaderM { runReaderM :: env -> Either String a }
```

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

Puteti testa solutia pe urmatoarea expresie:

```
testReaderM :: ReaderM String String
testReaderM = ma >>= k
   where
     ma =
      ReaderM
        (\ str -> if length str > 10 then Right (length str) else Left "")
     k val =
      ReaderM
        (\ str -> if val `mod` 2 == 0 then Right "par" else Left "")
```