

Examen Programare Functionala

1 februarie 2023, ora 10:00

V1

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 ajutoare. Puteti avea ca materiale ajutoare, 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 ajutoare 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 ajutoare.

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

Fisier: **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 problema, la linkul corespunzător problemei, care va fi pe foaia cu subiecte.

Atenție! Fiecare student poate încărca 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/9svjXX3XJ7zUaBIZRREs>

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

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

251 - P3 <https://www.dropbox.com/request/oD0UXBfp6CfNIAISwQH>

Rezolvarea subiectelor. Va rugam sa cititi cu atentie 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 | Expr :+: Expr | 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 clasei `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 = ((Val 1) :+: (Var "x")) :*: (Val 1)
```

```
ex2 = ex1 :+: (Val 3)
```

```
ex3 = ((Val 0) :*: (Val 2)) :+: (Val 3)
```

```
ex4 = 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 vocală, se adaugă un 'p' și vocala din nou, e.g., 'a' -> "apa"
- dacă caracterul nu este o vocală, rămâne nemodificat, e.g., 'p' -> "p"

Sa se rezolve problema în două moduri (o soluție fără monade și o soluție cu monade).

Exemple:

"Mi-e foame" -> "Mipi-epe fopoapamepe"

"Ana" -> "Apanapa"

P3 [1 pct]

Se da tipul de date

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

Sa se scrie instanța completă a clasei `Monad` pentru tipul `ReaderM`, astfel încât să păstreze proprietatea de monadă. Nu este nevoie să faceți instanțe și pentru clasele `Applicative` și `Functor`.

Puteti testa soluția pe următoarea expresie:

```
testReaderM :: ReaderM String String
```

```
testReaderM = ma >>= k
```

```
  where
```

```
    ma =
```

```
      ReaderM
```

```
        (\ str -> if length str > 10 then Just (length str) else Nothing)
```

```
    k val =
```

```
      ReaderM
```

```
        (\ str -> if val `mod` 2 == 0 then Just "par" else Nothing)
```