საგნის დასახელება: ფუნქციონალური პროგრამირება

სპეციალობა: კომპიუტერული მეცნიერება

ლექტორი: ასოც.პროფ. ნათელ არჩვაძე

ქულათა ჯამი: 20

ქულათა გადანაწილების ინსტრუქცია: 1-4 საკითხი -2ქ., 5-8 საკითხი -3ქ. =19

1. განსაზღვრეთ ფუნქცია, რომელიც შესასვლელზე ღებულობს მთელ რიცხვს n-ს და აბრუნებს 1-დან n -მდე ნატურალური რიცხვის კვადრატების სიას;

let nSquare 0 = []; nSquare n = n\*n:nSquare(n-1); nSquareR n = reverse(nSquare n) +2

1. სტანდარტული prelude ფაილის გამოუყენებლად რეკურსიის საშუალებით განსაზღვრეთ ფუნქცია, რომელიც შექმნის n ერთნაირი ელემენტების სიას: replicate::Int->a->[a].

let replicate 0 a = [a]; replicate n a = a:replicate (n-1) a +2

1. განსაზღვრეთ ფუნქცია, რომელიც შესასვლელზე ღებულობს მთელ რიცხვებს n-ს და m-ს და აბრუნებს n-დან m -მდე კენტი სამის ჯერადი რიცხვების სიას.

let odd3 a b | a == b = [] | mod a 3==0 && odd a= a:odd3(a+1) b| otherwise = odd3 (a+1) b +2

1. ფუნქცია ამოაგდებს მოცემული სიიდან n - ურ წევრს. შეცვალეთ ფუნქციის განმარტება - გაითვალისწინეთ შემთხვევა, როცა n-ის მნიშვნელობა სიის სიგრძეზე მეტია.

let pop n [] = []; pop 0 (x:xs) = xs; pop n (x:xs) = x:pop (n-1) xs +1

1. ორი სიის ელემენტების აჯამვის ფუნქცია. აბრუნებს სიას, რომელიც შედგება პარამეტრი სიების ელემენტების ჯამისგან. გაითვალისწინეთ, რომ გადაცემული სიები შეიძლება იყოს სხვადასხვა სიგრძის. მაგალითად, listsum [1,2,3][4,5]=[5,7,3].

let sumListElements [] [] = []; sumListElements [] ls = ls; +3 sumListElements ls [] = ls; sumListElements (l:ls) (n:ns) = l+n:sumListElements ls ns

sumListElements :: Num a => [a] -> [a] -> [a]

sumListElements [4, 5, 6] [4, 5]

[8,10,6]

it :: [Integer]

1. ფუნქცია substitute :: Char -> Char -> String -> String, რომელიც ცვლის მოცემულ სიმბოლოს მეორე სიმბოლოთი. მაგალითად, substitute ’e’ ’i’ "eigenvalue" აბრუნებს "iiginvalui".

let substitute m n [] = []; substitute m n (a:as) | m == a = n:substitute m n as | otherwise = a :substitute m n as +3

substitute :: Eq a => a -> a -> [a] -> [a]

substitute 'a' 'm' "ankle"

"mnkle"

it :: [Char]

1. განსაზღვრეთ ფუნქცია makePositive, რომელიც უცვლის ნიშანს რიცხვების სიის ყველა ელემენტს. მაგალითად, makePositive [-1, 10, 5, -10, -20] გვაძლევს [1,-10,-5,10,20]. +3

let makePositive [] = []; makePositive (n:ns) = -n:makePositive ns

makePositive :: Num a => [a] -> [a]

makePositive [-1,5]

[1,-5]

it :: [Integer]

1. განსაზღვრეთ ფუნქცია, რომელიც მოცემული არგუმენტიდან (სიიდან) ამოაგდებს მხოლოდ იმ a რიცხვებს, რომლებიც აკმაყოფილებენ პირობას: 150<a<400.

let takeWithoutRange [] = []; takeWithoutRange (n:ns) = if n>150 && n<400 then takeWithoutRange ns else n:takeWithoutRange ns +3

takeWithoutRange :: (Ord a, Num a) => [a] -> [a]

takeWithoutRange [15, 156, 400, 500]

[15,400,500]

it :: [Integer]