Институт приборостроения, автоматизации и информационных технологий

Кафедра информационных систем и цифровых технологий

Дисциплина «Функциональное и логическое программирование»

Отчет к лабораторной работе № 6

«Рекурсивные операции со списками на Haskell»

Выполнил:

Василения Иван Валерьевич

Мельниченко Артём Олегович

Принял:

Гордиенко А.П.

Орёл, 2024г

module Main where

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

suff [] = [[]]

suff (x:xs)

= (x:xs): suff xs

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

dellast [\_] = []

dellast (x:xs) = x: dellast xs

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

pref [] = [[]]

pref xs = xs : (pref (dellast xs))

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

un [] ys = ys

un xs [] = xs

un (x:xs) ys

| x `elem` ys = un xs ys

| otherwise = x : un xs ys

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

seg [] = [[]]

seg (x:xs) = un (pref (x:xs)) (seg xs)

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

permutations [] = [[]]

permutations xs = concatMap (\x -> map (x:) (permutations (delete x xs))) xs

where

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

delete \_ [] = []

delete x (y:ys) | x == y = ys

| otherwise = y : delete x ys

main :: IO ()

main = do

let list = [1, 2, 3]

completeSuff = suff list

completePref = pref list

completeSeg = seg list

completePer = permutations list

print completeSuff

print completePref

print completeSeg

print completePer