Робота № 2

Визначення рекурсивних функцiй

Варіант № 12

**Мета роботи**

Набути досвiду визначення та використання функцій вищого порядку

**Завдання**

Визначте вказанi функцiї в кожному з завдань:

а) без застосування,

б) з застосуванням вбудованих функцiй вищого порядку.

1.12. Роздiлити список на двi частини при заданiй довжинi першої n, напр.

при n=3: "abcdefghik"⇒ ("abc "defghik").

2.12. Перевiрити гiпотезу Ґольдбаха у вказаному дiапазонi.

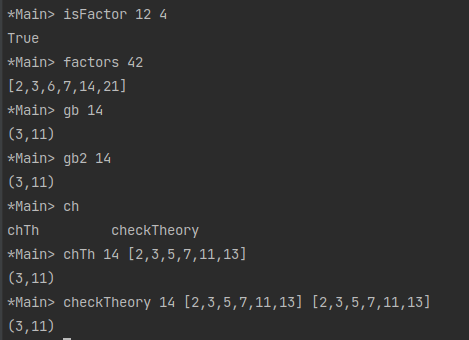
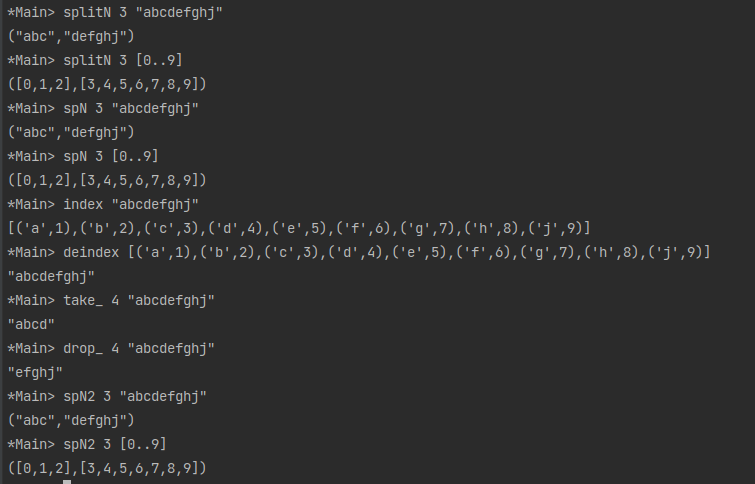
**Код програми**

module Main (main) where  
  
  
import Data.List (unfoldr)  
import Data.Maybe (listToMaybe)  
  
main :: IO ()  
main = do  
 putStrLn "hello world"  
  
  
{-  
Визначте вказанi функцiї в кожному з завдань: а) без застосування, б) з  
застосуванням вбудованих функцiй вищого порядку.  
  
  
1.12 Роздiлити список на двi частини при заданiй довжинi першої n, напр.  
при n=3: "abcdefghik"⇒ ("abc "defghik")  
  
2.12 Перевiрити гiпотезу Ґольдбаха у вказаному дiапазонi.  
  
-}  
  
  
-- 1.12 -------------------------------------------------------------------------------------------------------------  
  
  
*-- | Фунція для розбиття списку на дві частини при заданому індексу розбиття -- n*  
*-- | без використання ФВП*  
*-- >> splitN 3 "abcdefghj"*  
*-- > ("abc","defghj")*  
*--*  
*-- >> splitN 3 [0..9]*  
*-- > ([0,1,2],[3,4,5,6,7,8,9])*  
splitN :: Int -> [a] -> ([a], [a])  
splitN n list = splitAt n list  
  
*-- | Фунція для розбиття списку на дві частини при заданому індексу розбиття -- n*  
*-- | з використанням вбудовинх фцнкцій take та drop*  
*-- | без використання ФВП*  
*-- >> spN 3 "abcdefghj"*  
*-- > ("abc","defghj")*  
*--*  
*-- >> spN 3 [0..9]*  
*-- > ([0,1,2],[3,4,5,6,7,8,9])*  
spN n xs = (take n xs, drop n xs)  
  
  
*-- | Фунція для індексування списку*  
*-- | з використанням вбудовинх фцнкцій take та drop*  
*-- >> index "abcdefghj"*  
*-- > [('a',1),('b',2),('c',3),('d',4),('e',5),('f',6),('g',7),('h',8),('j',9)]*  
index :: [a] -> [(a, Int)]  
index ls = zip ls [1..length ls]  
  
*-- | Фунція для деіндексування списку*  
*-- | з використанням вбудовинх фцнкцій take та drop*  
*-- >> deindex [('a',1),('b',2),('c',3),('d',4),('e',5),('f',6),('g',7),('h',8),('j',9)]*  
*-- > "abcdefghj"*  
deindex :: [(a, Int)] -> [a]  
deindex [] = []  
deindex ((i,x):xs) = i : deindex xs  
  
  
*-- | Фунція для створення списку з перших n елементів заданого списку*  
*-- >> take\_ 4 "abcdefghj"*  
*-- > "abcd"*  
take\_ :: Int -> [a] -> [a]  
take\_ k \_  
 | k <= 0 = []  
take\_ \_ [] = []  
take\_ n list = deindex (takeWhile (\ (x,y) -> y <= n) (index list))  
  
  
*-- | Фунція для створення списку з усіх після n перших елементів заданого списку*  
*-- >> drop\_ 4 "abcdefghj"*  
*-- > "efghj"*  
drop\_ :: Int -> [a] -> [a]  
drop\_ k \_  
 | k <= 0 = []  
drop\_ \_ [] = []  
drop\_ n list = deindex (dropWhile (\ (x,y) -> y <= n) (index list))  
  
  
*-- | Фунція для розбиття списку на дві частини при заданому індексу розбиття -- n*  
*-- | з використання ФВП*  
*-- >> spN2 3 "abcdefghj"*  
*-- > ("abc","defghj")*  
*--*  
*-- >> spN2 3 [0..9]*  
*-- > ([0,1,2],[3,4,5,6,7,8,9])*  
spN2 n xs = (take\_ n xs, drop\_ n xs)

-- 2.12 -------------------------------------------------------------------------------------------------------------

*-- | Фунція для отримання дільників числа*  
*-- | з використання ФВП filter*  
*-- >> factors 42*  
*-- > [2,3,6,7,14,21]*  
factors a = filter (isFactor a) [2..a-1]  
  
  
*-- | Функція для перевірки на дільник цілого числа*  
*-- >> isPr 11*  
*-- > True*  
isFactor a b = a `mod` b == 0  
  
  
*-- | Функція для перевірки на просте число*  
*-- >> isPr 11*  
*-- > True*  
isPr 1 = False  
isPr a = null $ factors a  
  
  
*-- | Фунція для перевірки гіпотези Ґольдбаха*  
*-- | з використання ФВП filter та map*  
*-- >> gb2 14*  
*-- > (3,11)*  
gb2 :: Int -> (Int, Int)  
gb2 a = head $ filter (\(x,y) -> isPr x && isPr y) $  
 map (\c -> (c, a - c)) [3,5..a `div` 2]  
  
  
  
  
*-- | Фунція для перевірки гіпотези Ґольдбаха*  
*-- | без використання ФВП*  
*-- >> checkTheory 14 [2,3,5,7,11,13] [2,3,5,7,11,13]*  
*-- > (3,11)*  
checkTheory :: Int -> [Int] -> [Int] -> (Int, Int)  
checkTheory n ls (x:xs)  
 | any\_ ls n x /= (0, 0) = any\_ ls n x  
 | otherwise = checkTheory n ls xs  
 where  
 any\_ [] n k = (0, 0)  
 any\_ (x:xs) n k  
 | x + k == n = (k, x)  
 | otherwise = any\_ xs n k  
  
  
*-- | Фунція для перевірки коректності введеного числа,*  
*-- | Якщо число задовольняе умови теорії, то повертае функціії перевірки checkTheory*  
*-- >> chTh 14 [2,3,5,7,11,13]*  
*-- > (3,11)*  
chTh :: Int -> [Int] -> (Int, Int)  
chTh n ls  
 | n < 2 = (0, 0)  
 | odd n = (0, 0)  
 | otherwise = checkTheory n ls ls  
  
*-- | Фунція для перевірки гіпотези Ґольдбаха*  
*-- | без використання ФВП*  
*-- >> gb 14*  
*-- > (3,11)*  
gb :: Int -> (Int, Int)  
gb n = chTh n $ genPrimes n  
 where  
 genPrimes n = [i | i <- [2..n], isPr i]

**Приклад роботи**



**Висновок**: під час виконання цієї лабораторної работи було набуто знання про ФВП та досвід роботи з ними. В результаті виконання роботи було завершено Роботу №2 Комп’ютерного парктикуму.