МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НАУКИ И ТЕХНОЛОГИЙ ИМЕНИ АКАДЕМИКА М.Ф. РЕШЕТНЕВА”

Институт информатики и телекоммуникаций

Кафедра информатики и вычислительной техники

**Отчёт по Лабораторной работе № 5**По дисциплине “Функциональное программирование”

**Вариант № 12**

**Выполнил:** студент 1 курса  
Группы: БПИ22-02  
Трифонов Кирилл Вячеславович

**Преподаватель:**   
доцент каф. ИВТ, к.т.н

Богданов Константин Валериевич

**Красноярск 2023**

Задания на лабораторную работу 5.

2. Определите тип  
данных, представляющий бинарные деревья поиска. В отличие от  
деревьев, представленных в методических указаниях, в деревьях  
поиска данные могут находиться не только в листьях, но и в промежуточных узлах дерева. Будем использовать деревья для представления ассоциативного массива, сопоставляющие значения ключей  
(представляемых как строки) целым числам. Для каждого узла с  
некоторым ключом в левом поддереве должны содержаться элементы с меньшими значениями ключа, а в правом — с большими. При  
поиске соответствия между строкой и числом необходимо учитывать эту информацию, поскольку она позволяет более эффективно  
извлекать информацию из дерева.

import Data.Maybe

*--Для работы maybe Just и isJust*

Предварительно импортируем пакет Maybe из библиотек GHC

*--Описание типа*

data *BinS* = Tip | Bin (*String*, *Int*) *BinS* *BinS* deriving *Show*

1) add, добавляющую в дерево заданную пару ключа и значения.

*--функция задания пары ключа и его значения*

add :: (*String*, *Int*) -> *BinS* -> *BinS*

add (k, v) Tip = Bin (k, v) Tip Tip

add (k, v)(Bin (k1, v1) l r)

            | k > k1 = Bin (k1, v1) l (add (k, v) r)

            | otherwise = Bin (k1, v1) (add (k, v) l) r

Если подаётся пустая ветка и пара, то возвращает ветки с парой. Если подаётся заполненная ветка, то она обновляется рекурсивно предварительно определив порядок исходя из ключа.

2) find, возвращающую число, соответствующее заданной строке.

*--функция поиска номера строки*

find :: *String* -> *BinS* -> *Maybe* *Int*

find k Tip = Nothing

find k (Bin (k1, v1) l r)

            | k == k1 = Just v1

            | otherwise = find k (if k > k1 then r else l)

Если подаётся пустая ветка, то возвращает Nothing из библиотеки Maybe, если заполненная ветка, то рекурсивно проверяет всё поисковое дерево и возвращает значение по ключу.

3) exists, проверяющую, что элемент с заданным ключом содержится в дереве.

*--проверка на существование*

exist :: *String* -> *BinS* -> *Bool*

exist k = isJust . find k

Оператор isJust возвращает True, если ключ k существует в дереве, иначе false.

4) toList, преобразующая заданное дерево поиска в список,

упорядоченный по значениям ключей.

*--преобразование в 1 уровневый список*

toList :: *BinS* -> [*Int*]

toList Tip = []

toList( Bin (\_, v1) l r) = concat [toList l, [v1], toList r]

С помощью оператора concat рекурсивно распаковываются списки.