# Лабораторная работа 4, задание 6.1 (по варианту)

## Условие

a\*

r\*

l

c

e\*

r

i

e\*

d\*

e\*

e

z

t\*

f

s

Рис. 1: trie-дерево

Пример trie-дерева приведен на рис. 1. Булевское значение, равное True отмечает конец слова, читаемого, начиная с корня дерева. На рисунке узлы с такими значениями помечены символом \*. Таким образом, в дереве представлены слова fa, false, far, fare, fact, fried, frieze. Определите следующие функции:

1)exists, которая проверяет, что заданное слово содержится в trie-дереве.

## Код

{-

ПИ19-3 Деменчук

Задание 6.1 (по варианту 5)

Пример trie-дерева приведен на рис. 1. Булевское значение, равное True отмечает конец слова, читаемого, начиная с корня дерева.

На рисунке узлы с такими значениями помечены символом \*.

Таким образом, в дереве представлены слова fa, false, far, fare, fact, fried, frieze. Определите следующие функции:

1)exists, которая проверяет, что заданное слово содержится в trie-дереве.

-}

-- Созданный кастомный тип дерева

data MyTree = Unit Char Bool [MyTree] deriving (Eq, Show)

-- Метод для проверки на наличие элемента в дерева

exist :: MyTree -> String -> Bool

exist t s = s `elem` (treeToList t "")

--Конвертер дерева в список

treeToList :: MyTree -> String -> [String]

treeToList (Unit c True \_) acc = [acc ++ [c]]

treeToList (Unit c False ts) acc = concatMap (\ u -> treeToList u (acc++[c])) ts

--Main-метод

main = do

--Построение тестового дерева

print(exist (Unit 'f' False [(Unit 'a' False [(Unit 'l' True []),(Unit 's' True [])])]) "fal")

print(exist (Unit 'f' False [(Unit 'a' False [(Unit 'l' True []),(Unit 's' True [])])]) "t")

## Скриншот

