МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Алгоритмы и структуры данных»
Тема: Рекурсивная обработка иерархических списков

Студент гр. 9303 _____ Ахримов А.М.

Преподаватель _____ Филатов А.Ю.

Санкт-Петербург
2020

Цель работы.

Познакомиться с одной из часто используемых на практике нелинейных конструкций, способами её организации и рекурсивной обработки. Получить навыки решения задач обработки иерархических списков, как с использованием базовых функций их рекурсивной обработки, так и без использования рекурсии.

Задание.

3) Подсчитать общую длину всех плеч заданного бинарного коромысла bk. Для этого ввести рекурсивную функцию

short Length (const БинКор bk).

Основные теоретические положения.

В практических приложениях возникает необходимость работы с более сложными, чем линейные списки, нелинейными конструкциями. Рассмотрим одну из них, называемую иерархическим списком элементов базового типа El или S-выражением.

Определим соответствующий тип данных S_expr (El) рекурсивно, используя определение линейного списка (типа L_list):

< S_expr (El) > ::= < Atomic (El) > | < L_list (S_expr (El)) >,

< Atomic (E) > ::= < El >.

< L_list(El) > ::= < Null_list > | < Non_null_list(El) >

< Null_lis t> ::= Nil

< Non_null_list(El) > ::= < Pair(El) >

< Pair(El) > ::= (< Head_l(El) > . < Tail_l(El) >)

< Head_l(El) > ::= < El >

< Tail_l(El) > ::= < L_list(El) >

Выполнение работы.

Для определения бинарного коромысла заведём следующие структуры:

- binTree – бинарное коромысло, у которого есть два плеча типа Arm*.
- Arm – плечо коромысла с полями length(длина плеча), tag(указывающее на то, есть ли продолжение у этого плеча) и Node.
- Node – структура, в заивсимости от tag, хранящая либо вес гирьки weight, либо binTree*.

Функция `short Length(const binTree* tree)` считает общую длину всех плеч бинарного коромысла. Данная функция вызывает две другие рекурсивные функции `traversal_arm` и `traversal_tree`, которые проходятся по всем плечам коромысла в глубину.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	((2((3 10) (1 0))) (4 -1))	tree depth level = 1 lenght = 2 current k = 2 tree depth level = 2 lenght = 3 current k = 5 tree depth level = 2 lenght = 1 current k = 6 tree depth level = 1 lenght = 4 current k = 10 Length = 10
2.	((4((51 9) (10((8 1) (78 2))))) (7((12 41) (20 35))))	tree depth level = 1 lenght = 4 current k = 4 tree depth level = 2 lenght = 51 current k = 55 tree depth level = 2 lenght = 10 current k = 65 tree depth level = 3 lenght = 8 current k = 73 tree depth level = 3 lenght = 78 current k = 151 tree depth level = 1 lenght = 7 current k = 158 tree depth level = 2 lenght = 12 current k = 170 tree depth level = 2 lenght = 20 current k = 190 Length = 190

Выводы.

В ходе выполнения данной работы была изучена такая нелинейная конструкция, как иерархические списки. Были приобретены навыки по их обработки с помощью рекурсивных функций.