**Отчет по лабораторной работе № 23 по курсу “Практикум на ЭВМ”**

Студент группы M8O-106Б-20 Почечура Артемий

Контакт e-mail: carbo59@yandex.ru

Преподаватель каф.806: Дубинин Алексей Владимирович

Входной контроль знаний с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан\_\_\_\_\_\_\_ <<\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_>> 2020г., итоговая оценка\_\_\_\_\_

Подпись преподавателя\_\_\_\_\_\_\_\_\_\_\_\_\_

**1. Тема:** Динамические структуры данных. Обработка деревьев.

**2. Цель работы:** Составить программу на языке Си для построения и обработки дерева общего

вида или бинарного дерева. Основные функции работы деревом реализовать в виде универсальных процедур или функций. (добавление узла, удаление узла, текстовая визуализация дерева, вычисление значения некоторой функции от дерева).

**3. Задание (вариант №):** вариант 8: определить число вершин дерева, степень которых совпадает со значением элемента.

**4. Оборудование ПЭВМ студента:** ЭВМ MSI GL62M 7RDX, процессор Intel core I7 c ОП 8192 Мб

**5. Программное обеспечение ЭВМ студента:**

Операционная система семейства Unix, наименование Ubuntu, версия 20.04

Интерпретатор команд GNU Bash версия 4.4.20

Редактор текстов nano

**Утилиты операционной системы: -**

**6. Идея, метод, алгоритм:**

1) Добавлять элемент в дерево, передавая в функцию путь, состоящий из букв b(brother), c(child) или просто точки (обозначение корня). Будем проходиться по буквам пути и идти по дереву вниз или вбок в зависимости от буквы на конкретной позиции. Пройдя весь указанный путь, ставим переданный элемент;

2) Удалять элемент будем примерно тем же способом, что и добавлять: передавать путь того элемента, который нужно удалить. Кроме удаления самого элемента, нужно рекурсивно удалить всех его детей, их детей и т.д.;

3) Чтобы реализовать задачу, которая соответствует моему варианту, нужно посчитать всех детей вершины и сравнить это значение со значением, которое лежит в самой вершине. Если они совпадают, то в счетчик нужно прибавить единицу. Проделать это нужно для каждой вершины дерева;

4) Печать дерева можно осуществить с помощью двух очередей (обход в ширину). В одну очередь кладём все вершины нулевого уровня, затем во вторую кладём детей каждой вершины нулевого подуровня и меняем очереди местами. Данное действие будет происходить в цикле, условие которого будет “пока первая очередь не пуста”. Вывод будет осуществлён по уровням.

Также можно печатать с помощью обхода в глубину. Для этого нам потребуется стек, в который мы будем класть элементы, пока проходим по детям какого-либо элемента. Когда все его дети пройдены, начинаем обход по братьям. В корне стека в этот момент уже будет лежать брат предыдущего элемента. Вывод будет осуществлён в виде полноценного дерева.

**7. Сценарий выполнения работы:**

1) Реализация ввода (считываем символы, пока не встретиться EOF);

2) Реализация функции ‘h’ (help):

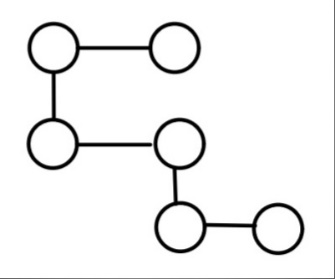
Тесты:

h

“Вывод справки”

3) Реализация функции ‘a’ (add element):

Тесты:



a one cccc

“Ошибка”

a odin cb

“Ошибка”

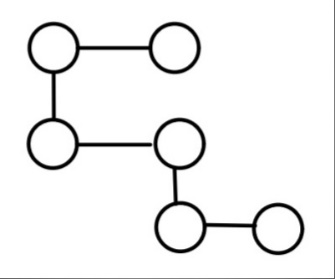
a two .c

“Ошибка”

a three ccc

“Успешное добавление”

4) Реализация функции ‘d’ (delete):



Тесты:

d cbc

“Успешное удаление”

d cbc

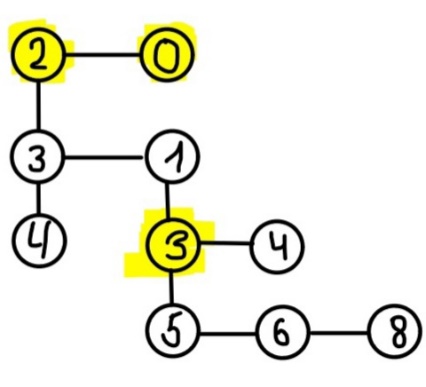
“Успешное удаление”

d cbc

“Ошибка”

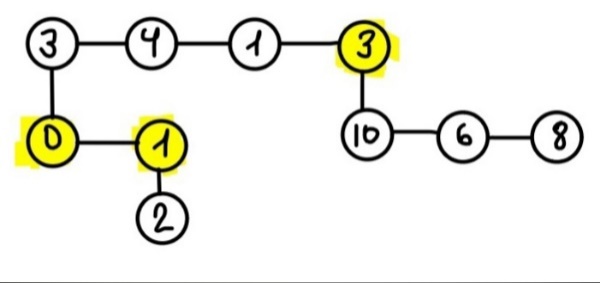
5) Реализация функции ‘t’ (task):

Тесты:



t

3

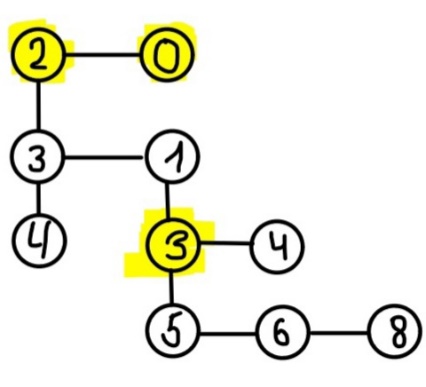


t

3

6) Реализация функции ‘p’ (print):

Тесты:



p dfs

two

three

four

one

three

five

six

eight

four

zero

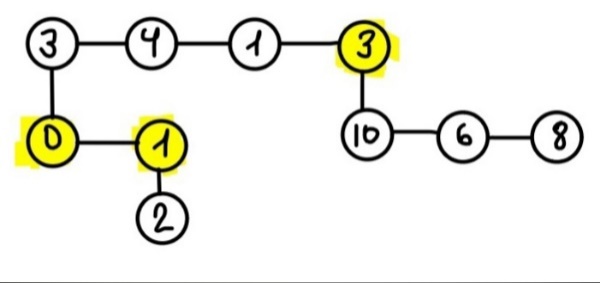
p bfs

two zero

three one

four three four

five six eight



p dfs

three

zero

one

two

four

one

three

ten

six

eight

p bfs

three four one three

zero one ten six eight

two

7) Реализация корректора ввода:

Тесты:

hhhhhhhhh

“Ошибка”

a one . g

“Ошибка”

d one c

“Ошибка”

p dfs bfs

“Ошибка”

t .

“Ошибка”

**8. Распечатка протокола:-**

**9. Дневник отладки: -**

**10. Замечание автора по существу работы: -**

**11. Выводы:** в ходе работы применил на практике знания по работе с очередями, с помощью которых можно было осуществить эффективный вывод дерева. Потренировался работать с указателями, активно использовал команду “valgrind” для проверки эффективности работы программы по памяти (в моей программе это было важно, т.к. помимо дерева выделение памяти происходило и для очереди, которая реализована на связных списках).

Подпись студента\_\_\_\_\_\_\_\_\_\_\_\_\_\_