Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский политехнический университет Петра Великого»

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе №1**

**Дисциплина:** «Низкоуровневое программирование»

**Тема:** «Дерево с узлами произвольной степени»

**Вариант №1**

Выполнил студент гр. 3530901/90004 \_\_\_\_\_\_\_\_\_\_ Сидоров А.А.

*(подпись)*

Преподаватель \_\_\_\_\_\_\_\_\_\_\_ Алексюк А. О.

*(подпись)*

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Санкт-Петербург

2021

# ВВЕДЕНИЕ

Дерево с узлами свободной степени. Узел дерева может иметь большое количество детей и большое количество значений.

## Постановка задачи

* Разработать статическую библиотеку, реализующую дерево.
* Разработать демонстрационную программу – консольное приложение, обеспечивающее ввод данных из файла (файлов), их обработку и вывод в файл (файлы); имена файлов передаются в качестве параметров командной строки.

## 2. Исходные коды

С готовой программой, которая описывается в данной работе, можно ознакомиться по адресу <https://github.com/Artyom-boop/n-aryTree>

# ОПИСАНИЕ API БИБЛИОТЕКИ

Реализация дерева с узлами свободной степени представляет из себя динамическую структуру данных. Она основана на структуре node, которая состоит из:

struct node {  
 struct node\* root; // родитель узла  
 int value; // значение узла  
 struct node\* \*children; // указатель на массив «детей»  
 unsigned countChildren; // размер массива «детей»  
};

## Основные функции библиотеки

Далее будут перечислены функции, созданные для работы с заданной структурой данных, которые объявлены в “*Tree.h”.*

int getValue(struct node\* node); - возвращает значение узла  
  
void setValue(struct node\* node, int value); - изменяет значение узла  
  
unsigned getSizeChildren(struct node\* node); - возвращает размер массива «детей»  
struct node\* getChildren(struct node\* node); - возвращает указатель на массив «детей»

struct node\* getRoot(struct node\* node); - возвращает родителя узла  
  
struct node\* add(struct node\* node, int value); - создает новый узел в дереве с заданным значением  
  
void addNode(struct node\* root, struct node\* node); добавляет новый узел в дерево  
  
void removeNodeAndChildren(struct node\* children); - удаляет узел и детей  
  
void removeNode (struct node\* node); - удаляет узел и сохраняет детей в дереве  
  
void clearTrie(struct node\* root); - очищает дерево  
  
void printConsole(struct node\* node, int level); - вывод дерева в консоль  
  
void printFile(struct node \*node, int level, FILE \*out); - вывод дерева в файл

# 2. ОПИСАНИЕ ПРИМЕРНОЙ ПРОГРАММЫ

Для демонстрации типового использования структуры данных В-дерево была написана программа, которая полностью помещена в файл *“src/main.c”.*

Данная программа создает узел, добавляет в него элементы из первого входного файла и выводит в выходной файл.

Чтобы использовать программу, необходимо собрать её с помощью утилиты GNU Make и средств сборки GCC. Делается это следующим образом.

make -f Makefile – сборка библиотеки.

make -f MakefileMain – создание исполняемого файла Main.exe

В результате в папке проекта tree окажется исполняемый файл Main.

# 3. РУКОВОДСТВО ПРОГРАММИСТА

Для использования библиотеки из исходного кода необходимо клонировать публичный git репозиторий проекта в локальную область.

git clone <https://github.com/Artyom-boop/n-aryTree>

Для сборки библиотеки достаточно выполнить следующую команду в корневой папке клонированного проекта.

make

Чтобы провести тесты достаточно выполнить следующую команду в папке test клонированного проекта.

make

Если хотя бы один тест не выполнился в системе пользователя библиотеки, то правильная работа не гарантируется.

# ЗАКЛЮЧЕНИЕ

В ходе данной работы была создана статически линкуемая библиотека, реализующая функциональность дерева с узлами свободной степени. На основе этой библиотеки была сделана демонстрационная программа, которая выполняет простейшие операции со структурой данных.