

ГУАП

КАФЕДРА № 34

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ  
ПРЕПОДАВАТЕЛЬ

Старший преподаватель  
должность, уч. степень, звание

подпись, дата

К.А. Жиданов  
инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

по курсу: ЯЗЫКИ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛА

СТУДЕНТКА ГР. № 3145

подпись, дата

Пьяно Ю.Д.  
инициалы, фамилия

Санкт-Петербург 2022

## Вариант №4

### Дерево

#### Цель работы

Реализовать АДТ (абстрактный тип данных) в виде пользовательского типа данных и набора функций, реализующих заданные операции. Помимо стандартных интерфейсов (чтение/добавление/поиск/удаление), требуется реализовать чтение/выгрузку данных из файла.

#### Ход работы

1. Создаем заголовочный файл для программы

```
#ifndef __TREE__
#define __TREE__
#include <stdio.h>
#include <stdlib.h>
#define N 6

typedef struct Tnode {
    int key;
    struct Tnode* desc[N];
} TNode;

//Tree create
TNode* node_create(int);
TNode* tree_add(TNode*, int);

int deserialize(TNode*, FILE*);

//Add an element in tree

#endif
```

2. Создаем Си-файл и реализуем в нем следующие функции:

##### 2.1. Создаём элемент ноды

```
TNode* node_create(int key) {

    TNode* node = (TNode*)malloc(sizeof(TNode));
    node->key = key;
    for (int i = 0; i < N; i++) {
        node->desc[i] = NULL;
    }
    return node;
};
```

## 2.2.Добавление элемента

```
TNode* tree_add(TNode* root, int key) {  
  
    TNode* node = (TNode*)malloc(sizeof(TNode));  
    node->key = key;  
    int i = 0;  
    while (root->desc[i] != NULL && i < N) {  
        i++;  
    }  
    if (root->desc[i] == NULL) {  
        root->desc[i] = node;  
        return root;  
    }  
    else if ( root->desc[i] != NULL && i > N) {  
        return NULL;  
    }  
}
```

## 2.3.Выгрузка дерева из файла

```
int deserialize(TNode* root, FILE* fp) {  
    int val;  
    if (!scanf(fp, " %d", &val) || val == MARKER) {  
        return 1;  
    }  
    root = node_create(val);  
    for (int i = 0; i < N; i++) {  
        if (deserialize(root->desc[i], fp))  
            break;  
    }  
    return 0;  
}
```

## 3. Создаем основной файл проекта с тестовыми значениями

```
#include "tree.h"  
int test() {  
    TNode* x = NULL;  
  
    if (x != NULL) {  
        return 1;  
    }  
  
    x = node_create(2);  
    if (x == NULL) {  
        return 2;  
    }  
  
    tree_add(x, 7);  
    tree_add(x, 9);  
  
    if (x->desc[0] == NULL) {  
        return 3;  
    }  
  
    if (x->desc[1] == NULL) {  
        return 4;  
    }  
  
    if (x->desc[2] != NULL) {  
        return 4;  
    }  
}
```

```
        return 0;
    }

    int main() {
        return test();
    }
```