МИНИСТЕРСТВО НАУКИ
И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

НГТУ
НЭТИ | **Факультет прикладной математики и информатики**

Кафедра прикладной математики

Практическое задание № 3+

по дисциплине «Структуры данных и алгоритмы»

БИНАРНЫЕ ДЕРЕВЬЯ

Бригада 2 ЯГОДКИН АЛЕКСАНДР

Группа ПМ-25

Вариант 4

Преподаватель ТРАКИМУС ЮРИЙ ВИКТОРОВИЧ

Новосибирск, 2023

1 Задание

Используя рекурсию построить копию заданного двоичного дерева T (тип элемента дерева задать самостоятельно). В файле дерево представлено в виде списочной записи.

2 Анализ задания

Входные данные: В файл in.txt вводятся дерево в виде списочной записи.

Выходные данные: В файл out.txt выводятся результаты программы: копия заданного бинарного дерева. “Не удалось открыть файл out.txt”; “не удалось открыть файл in.txt”;

Решение задачи: Дерево считывается из файла "in.txt", строится функцией BuildTree() и сохраняется в переменной t. Функция copytree() создает копию дерева, используя рекурсивный подход. Скопированное дерево записывается в файл "out.txt" функцией output().

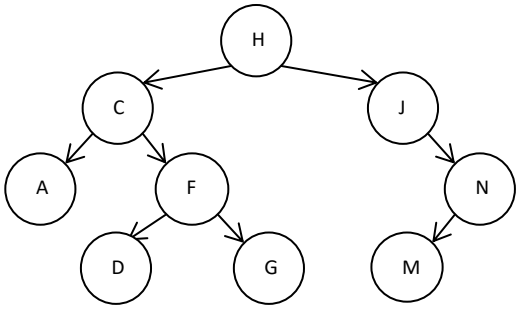
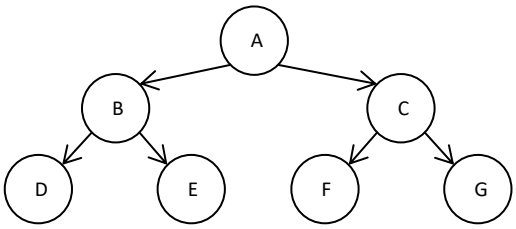
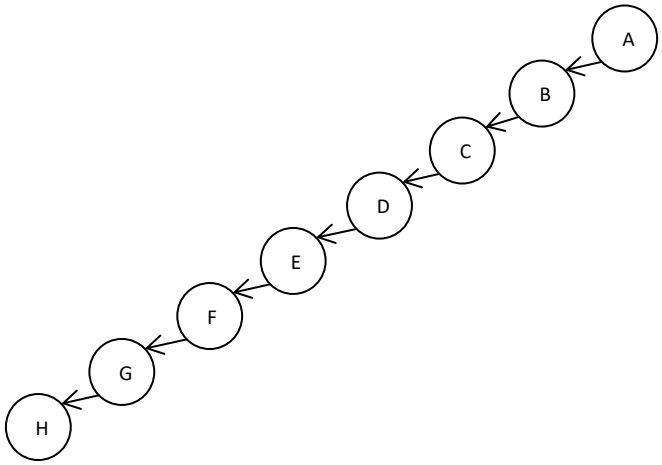
3 Программа

```
#include <stdio.h>
#include <windows.h>
#include <locale.h>

struct tree
{
   char elem;
   tree *left, *right;
   tree(char _elem = 0, tree *_right = NULL, tree *_left = NULL) :
      elem(_elem), left(_left), right(_right) { };
};

tree *BuildTree(FILE *f)
{
   char c = 0;
   fscanf_s(f, "%c", &c, sizeof(c));
   switch (c)
   {
      case '(':
               fscanf_s(f, "%c", &c, sizeof(c));
               return new tree(c, BuildTree(f), BuildTree(f));
      case ')':
      case ',': return BuildTree(f);
      case '0': return NULL;
      default: ;
   }
}

bool input(tree *&t)
{
   FILE *f = NULL;
   fopen_s(&f, "in.txt", "r");
   if (f)
   {
      t = BuildTree(f);
```

```
        fclose(f);
        return true;
    }
    else
        return false;
}

tree *copytree(tree *K)
{
    return K ? new tree(K->elem, copytree(K->right),
        copytree(K->left)) : NULL;
}

void printTree(tree *t, FILE *f)
{
    fprintf_s(f, "%c", t ? t->elem : '0');
    if (t)
    {
        fprintf_s(f, " (");
        printTree(t->left, f);
        fprintf_s(f, ") (");
        printTree(t->right, f);
        fprintf_s(f, ")");
    }
}

void output(tree *t)
{
    FILE *f = NULL;
    fopen_s(&f, "out.txt", "w");
    if (f)
    {
        fprintf_s(f, "Скопированное дерево:\n");
        printTree(t, f);
        fclose(f);
    }
    else
        printf_s("Не удалось открыть файл out.txt.");
}

int main()
{
    setlocale(0, "");
    tree *K = NULL;
    if (!input(K))
    {
        printf("Не удалось открыть файл in.txt.\n");
        return 1;
    }
    tree *t2 = copytree(K);
    output(t2);
```

```
    return 0;
}
```

4 Набор тестов

№	Входные данные	Назначение
1	(H, (C, (A, 0, 0), (F, (D, 0, 0), (G, 0, 0))), (J, 0, (N, (M, 0, 0), 0)))	Бинарное дерево поиска. 
2	(A, (B, (D, 0, 0), (E, 0, 0)), (C, (F, 0, 0), (G, 0, 0)))	Бинарное дерево, где у каждой ветви два потомка, кроме последнего уровня. 
3	(A, (B, (C, (D, (E, (F, (G, (H, 0, 0), 0), 0), 0), 0), 0), 0), 0), 0)	Вырожденное бинарное дерево, у которого только левые ветви. 

5 Результаты работы программы

№	Ввод/Вывод программы
1	скопированное дерево: (H (C (A (0) (0)) (F (D (0) (0)) (G (0) (0)))) (J (0) (N (M (0) (0)) (0)))
2	скопированное дерево: (A (B (D (0) (0)) (E (0) (0))) (C (F (0) (0)) (G (0) (0)))
3	скопированное дерево: (A (B (C (D (E (F (G (H (0) (0)) (0)) (0)) (0)) (0)) (0)) (0)) (0)