**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФРАСТРУКТУРИ ТА ТЕХНОЛОГІЙ**

ІНСТИТУТ УПРАВЛІННЯ, ТЕХНОЛОГІЙ ТА ПРАВА

**ФАКУЛЬТЕТ УПРАВЛІННЯ І ТЕХНОЛОГІЙ**

Кафедра інформаційних технологій

# Лабораторна робота №4

з дисципліни «Алгоритми і структури данних»

Виконав:

ст. гр. КН-19122

Мамаєв Антон Сергійович

Київ – 2021

**Лабораторна робота №4**

Умова задачі







Лістинг програми

#include <stdio.h>

#include <iostream>

#include <windows.h>

#include <fstream>

#include <iomanip>

using namespace std;

int tabs = 0; //Для создания отступов

int f = 0, f2 = 0, f3 = 0, f4 = 0;

struct Branch {

int Data;

Branch\* LeftBranch;

Branch\* RightBranch;

};

void Add(int aData, Branch\*& aBranch) {

if (!aBranch) {

aBranch = new Branch;

aBranch->Data = aData;

aBranch->LeftBranch = 0;

aBranch->RightBranch = 0;

if (f2 == 0)

f = aBranch->Data;

f2 = 1;

return;

}

else {

if (aBranch->Data > aData) {

Add(aData, aBranch->LeftBranch);

}

else {

Add(aData, aBranch->RightBranch);

}

}

}

void print(Branch\* aBranch)

{

if (!aBranch) return; //Если ветки не существует - выходим. Выводить нечего

tabs += 5; //Иначе увеличим счетчик рекурсивно вызванных процедур

//Который будет считать нам отступы для красивого вывода

print(aBranch->RightBranch); //Выведем ветку и ее подветки слева

for (int i = 0; i < tabs; i++) cout << " "; //Потом отступы

cout << aBranch->Data << endl; //Данные этой ветки

print(aBranch->LeftBranch);//И ветки, что справа

tabs -= 5; //После уменьшим кол-во отступов

return;

}

void pr\_obh(Branch\*& aBranch)

{

if (NULL == aBranch) return; //Если дерева нет, выходим

pr\_obh(aBranch->LeftBranch); //Обошли левое поддерево

pr\_obh(aBranch->RightBranch); //Обошли правое поддерево

cout << aBranch->Data << endl; //Посетили узел

}

void pr\_obh2(Branch\*& aBranch, int el)

{

if (NULL == aBranch) return; //Если дерева нет, выходим

pr\_obh2(aBranch->LeftBranch, el); //Обошли левое поддерево

pr\_obh2(aBranch->RightBranch, el); //Обошли правое поддерево

if (f3 == 0)

cout << aBranch->Data << endl; //Посетили узел

if (aBranch->Data == el) {

f3 = 1;

}

}

void FreeTree(Branch\* aBranch)

{

if (!aBranch) return;

FreeTree(aBranch->LeftBranch);

FreeTree(aBranch->RightBranch);

delete aBranch;

return;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int vel;

Branch\* Root = 0;

int element, dia = 1; srand(time(NULL));

while (dia != 0) {

cout << "\nОберіть дію: 1 - створити нове випадкове дерево, 2 - зворотній обхід дерева, 3 - чи належить елемент до дерева, 0 - завершити роботу\n";

cin >> dia;

switch (dia)

{

case 1:

Root = 0;

cout << "Введіть к-сть елементів для дерева: ";

cin >> vel;

cout << endl;

for (int i = 0; i < vel; i++)

{

Add(rand() % 25, Root);

}

cout << "Виведення бінарного дерева: " << endl;

print(Root);

cout << endl;

break;

case 2:

cout << "Зворотній обхід бінарного дерева: " << endl;

pr\_obh(Root);

cout << endl;

break;

case 3:

f3 = 0;

cout << "Введіть елемент для пошуку: ";

cin >> element;

cout << "Пройдені вузли при пошуку:\n";

pr\_obh2(Root, element);

if (f3 == 0) cout << "Серед пройдених вузлів не знайдено вказаного числа\n";

break;

}

}

}

Результат виконання

