

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №6 з дисципліни
«Основи програмування – 2. Методології програмування»

«Дерева»

Варіант 22

Виконав студент ІП-13, Музичук Віталій Андрійович
(шифр, прізвище, ім'я, по батькові)

Перевірила Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 4

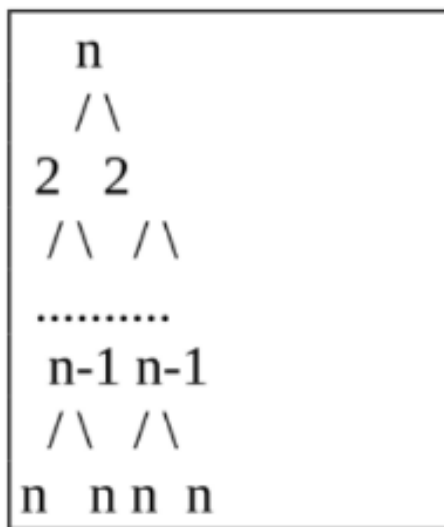
Перевантаження операторів

Мета – вивчити особливості організації і обробки дерев.

Варіант 22

Завдання:

Побудувати дерево наступного виду:



, де n - додатне ціле число

1. Виконання завдання на мові C++:

// Lab_6.cpp

```
#include "tree.h"

int main() {
    int value, how_print; bool flag = true;
    while (flag) {
        try {
            cout << "Enter a number: ";
            string input;
            cin >> input;
            value = stoi(input);
            if (value <= 0) throw "You enter wrong number of nodes";

            cout << "How you want to print tree: 0 - vertically, 1 - horizontally?" << endl;
```

```

        cin >> input;
        how_print = stoi(input);

        if (how_print != 0 && how_print != 1) throw "You have
entered wrong number for print method";
        flag = false;

    }
    catch (const invalid_argument& ex) {
        cout << "The entered numbers is incorrect\nTry again" <<
endl;
    }
    catch (const char* arr) {
        cout << arr << endl << "Try again" << endl;
    }
}

BinaryTree tree;
tree.createTree(value);

if (how_print == 0)
    tree.printVertical();
else
    tree.printHorizontal();

return 0;
}

```

//tree.cpp

```
#include "tree.h"
```

```
Node::Node(int number) {  
    this->number = number;  
    this->left = NULL;  
    this->right = NULL;  
}
```

```
int Node::num_recursion = 0;
```

```
void BinaryTree::createTree(int value) {  
    root = new Node(1);  
    max_num = value;  
    if (value > 1)  
    {  
        insertNode(root, 2);  
    }  
}
```

```
void BinaryTree::insertNode(Node* node, int num) {  
    if (!node->left && !node->right) {  
        node->left = new Node(num);  
        node->right = new Node(num);  
    }  
    if (num < max_num)  
        insertNode(node->left, num + 1);  
    if (num < max_num)  
        insertNode(node->right, num + 1);  
}
```

```
}
```

```
void BinaryTree::printVertical() {  
    if (root == NULL)  
        return;  
    vector<Node*> stack;  
    stack.push_back(root);  
    int tabs = (pow(2, max_num) - 1) / 2;  
    int level = 0;  
    while(!stack.empty())  
    {  
        for (int k = 0; k < pow(2, level); k++)  
        {  
            for (int i = 0; i < tabs; i++) cout << " ";  
            Node* node = stack.front();  
            cout << node->number;  
            stack.erase(stack.begin());  
            for (int i = 0; i < tabs + 1; i++) cout << " ";  
  
            if (node->left)  
                stack.push_back(node->left);  
  
            if (node->right)  
                stack.push_back(node->right);  
        }  
        level++;  
        tabs /= 2;  
        cout << endl << endl;  
    }  
}
```

```
}
```

```
void BinaryTree::printHorizontal() {  
    root->print();  
}
```

```
void Node::print() {  
    if (right) {  
        num_recursion += 5;  
        right->print();  
    }  
    for (int i = 0; i < num_recursion; i++) cout << " ";  
    cout << this->number << endl;  
    if (left) {  
        num_recursion += 5;  
        left->print();  
    }  
    num_recursion -= 5;  
}
```


// tree.h

```
#pragma once  
#include <iostream>  
#include <string>  
#include <vector>  
  
using namespace std;
```

```
class Node {
    int number;
    Node* left, * right;
    static int num_recursion;
public:
    Node(int);
    void print();
    friend class BinaryTree;
};

class BinaryTree {
    Node* root;
    int max_num;
public:
    BinaryTree() : root(NULL), max_num(0) {}
    void createTree(int value);
    void insertNode(Node* node, int number);
    void printVertical();
    void printHorizontal();
};
```

Тестування програми:

 Microsoft Visual Studio Debug Console

```
Enter a number: 5
How you want to print tree: 0 - vertically, 1 - horizontally?
0
      1
    2  2
  3  3  3  3
4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
```



```
Enter a number: 7
```

How you want to print tree: 0 - vertically, 1 - horizontally?

Висновок: Під час виконання лабораторної роботи я вивчив особливості створення дерев на прикладі мови C++. Результатом виконання лабораторної роботи є програма, основним завданням якої є створення дерева та виведення його на екран. Після тестування програм можна зробити висновок, що вони справляються із поставленою задачею.