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. Модульне програмування»

«Дерева»

Варіант 29

Виконав студент ІП-15 Рибалка Ілля Сергійович
(шифр, прізвище, ім'я, по батькові)

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

Київ 2022

Лабораторна робота 5

Дерева

Мета - вивчити особливості організації і обробки дерев.

Індивідуальне завдання

Варіант 29

29. Побудувати дерево, елементами якого є символи. Визначити і вивести на друк усі термінальні вершини (листя) цього дерева.

Розв'язання

Код

main.cpp

```cpp
#include "Tree.h"



int main()

{

   std::string str;

   std::cout << "Введіть набір символів - ";

   std::getline(std::cin, str);

   while(str.size()<1)

   {

       std::cout << "Введіть набір символів - ";

       std::getline(std::cin, str);

   }

   Tree tree(str);

   std::cout << "Сформоване дерево на основі введених символів:" << std::endl;

   tree.print();
```

```
    std::cout << std::endl;

    std::cout << "Термінальні вершини цього дерева:" << std::endl;

    tree.printleafs();

    std::cout << std::endl;

    return 0;

}
```

Tree.h

```
#pragma once

#include <iostream>

#include <iomanip>

#include <string>



struct Branch

{

    char Data;

    Branch *Left;

    Branch *Right;

    Branch(char (ch)): Data(ch), Left(NULL), Right(NULL) {};

};



class Tree

{

     Branch *root;

    Branch *add(char, Branch *); // Додавання елементу до дерева

    void printleafs(Branch *); // Знаходження і виведення термінальних вершин
```

```
    void print(Branch *, int); // Виведення дерева

    void remove(Branch *); // Видалення дерева

public:

    Tree(std::string);

    void printleafs();

    void print();

    ~Tree();

};
```

Tree.cpp

```
#include "Tree.h"



Tree::Tree(std::string data)

{

    this->root = new Branch(data[0]);

    if (data.size()>1) for (int i = 1; i < data.size(); i++) this->root = add(data[i],
this->root);

}



Branch *Tree::add(char tData, Branch *node)

{

    if (node)

    {

        if (node->Data>=tData)

        {

            if (!node->Left) node->Left = new Branch(tData);

            else add(tData, node->Left);
```

```
        }
        else
        {
            if (!node->Right) node->Right = new Branch(tData);
            else add(tData, node->Right);
        }
    }
    else node = new Branch(tData);
    return node;
}

void Tree::print()
{
    print(this->root, 0);
}

void Tree::printleafs()
{
    printleafs(this->root);
}

void Tree::print(Branch *node, int n)
{
    if (node)
    {
```

```
        n++;

        print(node->Left, n);

        for (int i = 0; i < n; i++) std::cout << "   ";

        std::cout << node->Data << std::endl;

        print(node->Right, n);

        n--;

    }

}

void Tree::printleafs(Branch *node)

{

    if(node)

    {

        printleafs(node->Left);

        if (!node->Left && !node->Right)

            std::cout << node->Data << " ";

        printleafs(node->Right);

    }

}



void Tree::remove(Branch *node)

{

    if (node)

    {

        remove(node->Left);

        remove(node->Right);
```

```
        delete node;

    }

}



Tree::~Tree()

{

    remove(this->root);

}
```

Тестування

```
Введіть набір символів – qwerty
Сформоване дерево на основі введених символів:
     e
   q
       r
         t
     w
       y

Термінальні вершини цього дерева:
e t y
MacBook-Pro-Mac:Lab6 mac$ cd "/Volumes/files/S2
Введіть набір символів – 12345
Сформоване дерево на основі введених символів:
   1
     2
       3
         4
           5

Термінальні вершини цього дерева:
5
```

Висновок

Я вивчив особливості організації і обробки дерев. В ході роботи над лабораторною роботою було створено програму для обробки бінарного дерева, що складається з символів і виводу його листків.