

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. Метидології програмування»  
«Дерева»

Варіант 8

Виконав студент     ПІ-15 Дацьо Іван Іванович  
(шифр, прізвище, ім'я, по батькові)

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

## Лабораторна робота 6

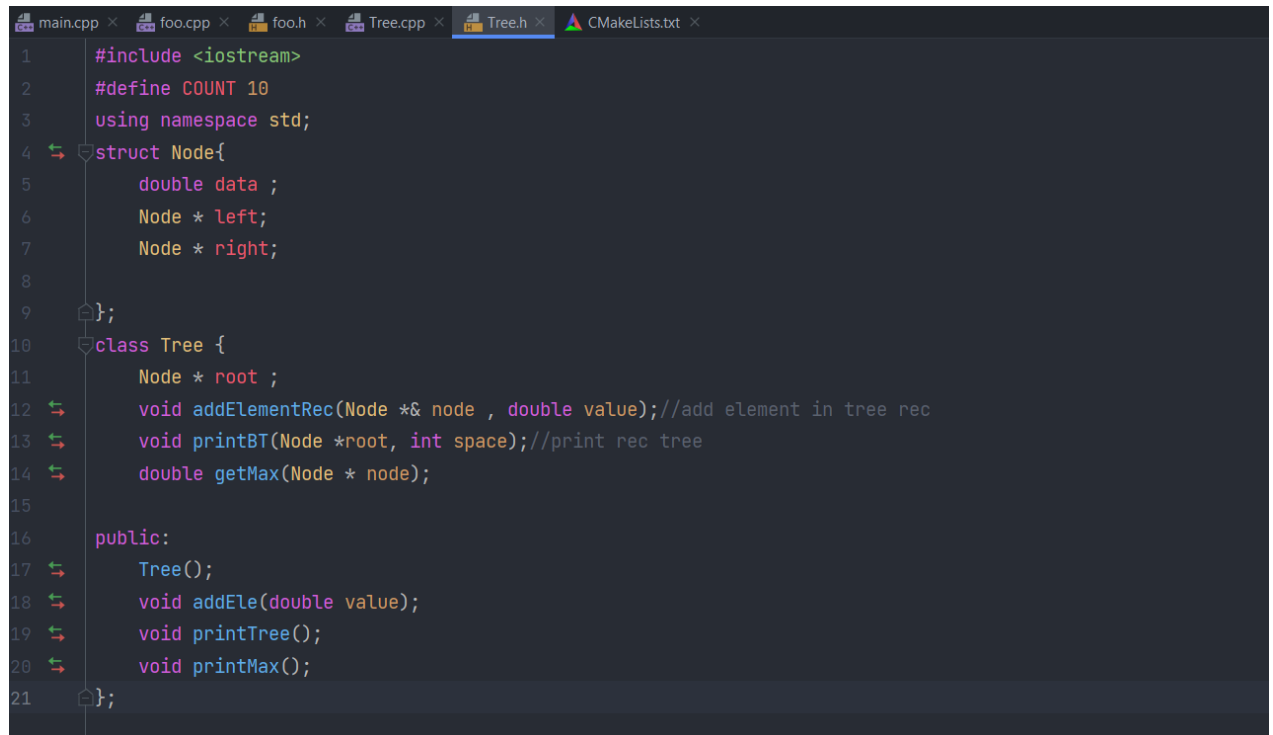
### Дерева

#### Варіант 8

8. Побудувати бінарне дерево, елементами якого є дійсні числа. Знайти значення найбільшого елемента цього дерева та надрукувати його.

#### Код програми

C++



```
1  #include <iostream>
2  #define COUNT 10
3  using namespace std;
4  struct Node{
5      double data ;
6      Node * left;
7      Node * right;
8
9  };
10 class Tree {
11     Node * root ;
12     void addElementRec(Node *& node , double value); //add element in tree rec
13     void printBT(Node *root, int space); //print rec tree
14     double getMax(Node * node);
15
16 public:
17     Tree();
18     void addEle(double value);
19     void printTree();
20     void printMax();
21 }
```

```

main.cpp x foo.cpp x foo.h x Tree.cpp x Tree.h x CMakeLists.txt x
1  #include "Tree.h"
2  Tree::Tree() {
3      root = nullptr;
4  }
5  void Tree::addElementRec(Node *& node, double value) {
6      if (!node) {
7          node = new Node;
8          node->left = nullptr;
9          node->right = nullptr;
10         node->data = value;
11         return;
12     }
13
14     if (value >= node->data) {
15         addElementRec(& node->right, value);
16     }
17     else {
18         addElementRec(& node->left, value);
19     }
20 }
21 void Tree::addEle(double value) {
22     this->addElementRec(& root, value);

```

```

main.cpp x foo.cpp x foo.h x Tree.cpp x Tree.h x CMakeLists.txt x
22     this->addElementRec(& root, value);
23 }
24 void Tree::printBT(Node *root, int space)
25 {
26
27     if (root == NULL) // base case
28         return;
29
30     space += COUNT; // + distance level
31
32     printBT(root->right, space); // print right
33
34     cout<<endl;
35     for (int i = COUNT; i < space; i++)
36         cout<<" ";
37     cout<<root->data<<"\n";
38
39     printBT(root->left, space); // print left
40
41 }
42 void Tree::printTree() {
43     this->printBT(root, space: 0);

```

```

main.cpp × foo.cpp × foo.h × Tree.cpp × Tree.h × CMakeLists.txt ×
40     printBT( root: root->left, space); // print left
41 }
42 void Tree::printTree() {
43     this->printBT(root , space: 0);
44 }
45
46 double Tree::getMax(Node* node) {
47     if (node->right != NULL) {
48         return getMax( node: node->right);
49     }
50     else {
51         return node->data;
52     }
53 }
54 void Tree::printMax() {
55     cout <<getMax( node: root);
56 }
57 }
58

```

```

main.cpp × foo.cpp × foo.h × Tree.cpp × Tree.h × CMakeLists.txt ×
1     #pragma once
2     #include "Tree.h"
3     #include <string>
4     using namespace std;
5     void addElements(Tree & tree , int size);
6     double valNumber();
7     bool isFloatNumber(string s);
8     int inputNumber();
9     bool isNumber(string s);
10

```

```

main.cpp x foo.cpp x foo.h x Tree.cpp x Tree.h x CMakeLists.txt x
1  #include "foo.h"
2  void addElements(Tree & tree , int size){
3      cout << "Enter elements : "<< endl;
4      double element;
5      for (int i = 0 ; i < size; i++){
6          element = valNumber();
7          tree.addEle( value: element);
8      }
9  }
10 double valNumber(){// Enter number of elements
11     string n;
12     cout << "Elem : ";
13     cin >> n;
14
15     while (!isFloatNumber( s: n) ){
16         cout << "Enter correct number : "<< endl;
17         cin >> n;
18     }
19     return stod( str: n);
20 }
21 bool isFloatNumber(string s)// check if it is float number
22 {

```

```

main.cpp x foo.cpp x foo.h x Tree.cpp x Tree.h x CMakeLists.txt x
22 {
23     int count = 0;
24     for (char ch : s) {
25         if (ch == '.'){
26             count ++;
27             if(count > 1){
28                 return false;
29             }
30         }
31         else if (!isdigit( C: ch) && ch != '-') return false;
32     }
33
34     return true;
35 }
36 int inputNumber(){
37     string n;
38     cout << "Enter number of elements : "<< endl;
39     cin >> n;
40     while (!isNumber( s: n)){
41         cout << "Enter correct number of elements : ";
42         cin >> n;
43     }

```

```
main.cpp x foo.cpp x foo.h x Tree.cpp x Tree.h x CMakeLists.txt x
34     return true;
35 }
36 int inputNumber(){
37     string n;
38     cout << "Enter number of elements : "<< endl;
39     cin >> n;
40     while (!isNumber(s: n)){
41         cout << "Enter correct number of elements : ";
42         cin >> n;
43     }
44     return stoi(str: n);
45 }
46 bool isNumber(string s) // check if it is int number
47 {
48     for (char ch : s) {
49         if (!isdigit(ch)) return false;
50     }
51
52     return true;
53 }
54
55
```

```
main.cpp x foo.cpp x foo.h x Tree.cpp x Tree.h x CMakeLists.txt x
1  #include <iostream>
2  #include "foo.h"
3  int main (){
4      Tree tree;
5      int size = inputNumber();
6      addElemnts(& tree, size);
7      cout << "Tree: " << endl;
8      tree.printTree();
9      cout << "Max Element : " << endl;
10     tree.printMax();
11 }
12
13
```

## Тестування:

Enter number of elements :

6

Enter elements :

Elem :4

Elem :7

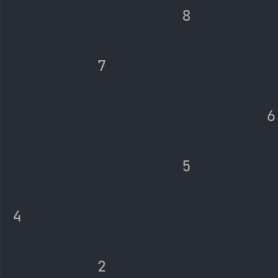
Elem :8

Elem :2

Elem :5

Elem :6

Tree:



Max Element :

8

## Висновки:

Було вивчено особливості організації та обробки дерев та застосовано ці навички на практиці.