2022113573 张宇杰

建立动态二叉树，并显示

```cpp
#include "DynamicBinaryTree.hpp"
#include "StaticBinaryTree.hpp"

#include <sstream>
#include <iostream>
#include <string>

using namespace std;
using namespace BinaryTree;

int main(int argc, const char *argv[])
{
    system("chcp 65001");
    stringstream ss("ABDH##I##E##CF#J##G##");

    DynamicBinaryTree<char> tree;

    tree.create(ss, '\0', '#');

    tree.show(cout);

    return 0;
}
```

```
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
                                                    g++ main.cpp -o test
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src> ./test
Active code page: 65001
    ┌─G
┌─C
│   │   ┌─J
│   └─F
A
│   ┌─E
└─B
    │   ┌─I
    └─D
        └─H
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
```

将拓展先序遍历序列输出至文件 fstream 即为保存

将输入流 stringstream 改为 fstream 即为从文件中加载

建立静态二叉树，并显示

```cpp
#include "DynamicBinaryTree.hpp"
#include "StaticBinaryTree.hpp"

#include <sstream>
#include <iostream>
#include <string>

using namespace std;
using namespace BinaryTree;

int main(int argc, const char *argv[])
{
    system("chcp 65001");
    stringstream ss("ABDH##I##E##CF#J##G##");

    StaticBinaryTree<char> tree;

    tree.create(ss, '\0', '#');

    tree.show(cout);
    tree.show_space(15);

    return 0;
}
```

```
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
                                                    g++ main.cpp -o test
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src> ./test
Active code page: 65001
    ┌─G
  ┌─C
  │ │   ┌─J
  │ └─F
A
  │   ┌─E
  └─B
    │   ┌─I
    └─D
      └─H
Space of <c>
Size: 512
Index || Data   || LChild || RChild
    0 || ♦      || 11     || 7476880
    1 || A      || 2      || 7
    2 || B      || 3      || 6
    3 || D      || 4      || 5
    4 || H      || -1     || -1
    5 || I      || -1     || -1
    6 || E      || -1     || -1
    7 || C      || 8      || 10
    8 || F      || -1     || 9
    9 || J      || -1     || -1
   10 || G      || -1     || -1
   11 ||        || 12     || 0
   12 ||        || 13     || 0
   13 ||        || 14     || 0
   14 ||        || 15     || 0
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
```

采用二叉树的上述二叉链表存储结构，编写程序实现二叉树的先序、中序和后序遍历的递归和非递归算法以及层序遍历算法，并以适当的形式显示和保存二叉树及其相应的遍历序列

```
#include "DynamicBinaryTree.hpp"
#include "StaticBinaryTree.hpp"

#include <sstream>
#include <iostream>
#include <string>

using namespace std;
using namespace BinaryTree;

int main(int argc, const char *argv[])
{
    system("chcp 65001");
    stringstream ss("ABDH##I##E##CF#J##G##");

    DynamicBinaryTree<char> tree;

    tree.create(ss, '\0', '#');

    tree.show(cout);
    tree.for_each(PRE_ORDER, [](char ch){cout << ch << ' ';}, [](){cout << "# ";});
    cout << endl;
    tree.for_each_no_rec(PRE_ORDER, [](char ch){cout << ch << ' ';}, [](){cout << "# ";});
    cout << '\n' << endl;
    tree.for_each(IN_ORDER, [](char ch){cout << ch << ' ';}, [](){cout << "# ";});
    cout << endl;
    tree.for_each_no_rec(IN_ORDER, [](char ch){cout << ch << ' ';}, [](){cout << "# ";});
    cout << '\n' << endl;
    tree.for_each(POST_ORDER, [](char ch){cout << ch << ' ';}, [](){cout << "# ";});
    cout << endl;
    tree.for_each_no_rec(POST_ORDER, [](char ch){cout << ch << ' ';}, [](){cout << "# ";});
    cout << '\n' << endl;
```

```
33.         tree.for_each(LEVER_ORDER, [](char ch){cout << ch << ' ';}, [
              ](){cout << "# ";});
34.         cout << endl;
35.
36.         return 0;
37.     }
```

```
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src> g++ main.cpp -o test
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src> ./test
Active code page: 65001
    ┌─G
  ┌─C
  │ │   ┌─J
  │ └─F
A
  │   ┌─E
  └─B
    │   ┌─I
    └─D
      └─H
A B D H # # I # # E # # C F # J # # G # #
A B D H # # I # # E # # C F # J # # G # #

# H # D # I # B # E # A # F # J # C # G #
# H # D # I # B # E # A # F # J # C # G #

# # H # # I D # # E B # # # J F # # G C A
# # H # # I D # # E B # # # J F # # G C A

A B C D E F G H I # # # J # # # # # # # #
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
```

将拓展先序遍历序列输出至文件 fstream 即为保存

将输入流 stringstream 改为 fstream 即为从文件中加载

注：for_each 中的形参使用的是 lambda 表达式

设计并实现判断任意一棵二叉树是否为完全二叉树的算法

```cpp
#include "DynamicBinaryTree.hpp"
#include "StaticBinaryTree.hpp"

#include <sstream>
#include <iostream>
#include <string>

using namespace std;
using namespace BinaryTree;

int main(int argc, const char *argv[])
{
    system("chcp 65001");
    stringstream ss("ABDH##I##E##CF#J##G##");

    DynamicBinaryTree<char> tree1, tree2;
    tree1.create(ss, '\0', '#');
    ss.str("ABDH##I##E##CF##G##");
    tree2.create(ss, '\0', '#');

    tree1.show(cout);
    cout << "tree1.complete() = " << tree1.complete() << endl;
    tree2.show(cout);
    cout << "tree2.complete() = " << tree2.complete() << endl;

    return 0;
}
```

```
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src> g++ main.cpp -o test
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src> ./test
Active code page: 65001
    ┌─G
  ┌─C
  │ │   ┌─J
  │ └─F
A
  │   ┌─E
  └─B
    │   ┌─I
    └─D
      └─H
tree1.complete() = 0
    ┌─G
  ┌─C
  │ └─F
A
  │   ┌─E
  └─B
    │   ┌─I
    └─D
      └─H
tree2.complete() = 1
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
```

设计并实现计算任意一棵二叉树的宽度的非递归算法

```cpp
#include "DynamicBinaryTree.hpp"
#include "StaticBinaryTree.hpp"

#include <sstream>
#include <iostream>
#include <string>

using namespace std;
using namespace BinaryTree;

int main(int argc, const char *argv[])
{
    system("chcp 65001");
    stringstream ss("ABDH##I##E##CF#J###");

    DynamicBinaryTree<char> tree1, tree2;
    tree1.create(ss, '\0', '#');
    ss.str("ABDH##I##E##CF##G##");
    tree2.create(ss, '\0', '#');

    tree1.show(cout);
    cout << "tree1.width() = " << tree1.width() << endl;
    tree2.show(cout);
    cout << "tree2.width() = " << tree2.width() << endl;

    return 0;
}
```

```
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
                                                    g++ main.cpp -o test
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src> ./test
Active code page: 65001
  ┌─C
  │ │ ┌─J
  │ └─F
A
  │ ┌─E
  └─B
    │ ┌─I
    └─D
      └─H
tree1.width() = 3
    ┌─G
  ┌─C
  │ └─F
A
  │ ┌─E
  └─B
    │ ┌─I
    └─D
      └─H
tree2.width() = 4
(base) PS D:\File\大二秋\DSA\作业3\binary-tree\src>
```