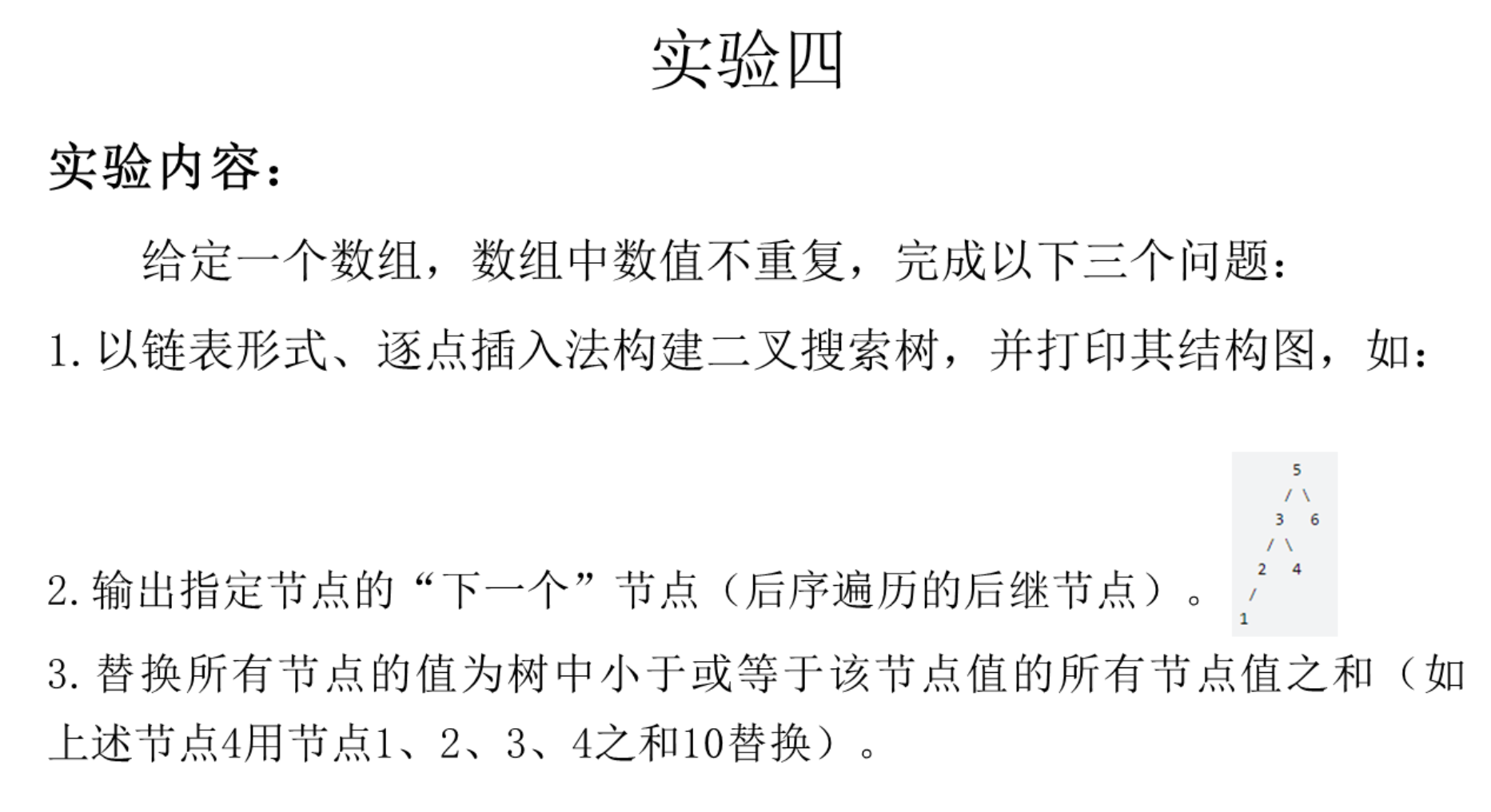
数据结构实验五

班级：20信安法 学号：2013484 姓名：张世伟

1. 实验内容



1. 设计思路
2. 打印思路

用一个二阶数组存储树

1. 输出二

寻找节点地址，后序遍历搜索树所有节点，存储在队列中。队列中的元素依次出队列，直到需要的元素。

1. 输出三

将储存输入元素储的数组元素，改为对应元素的和，然后依次插入到树中

1. 核心代码
2. 打印

void BinarySearchTree<T>::OutPutTree(BinaryTreeNode<T>\* nd, int r, int c, char\*\* screen)

{

if (nd->data < 0)

{

T tmp = -(nd->data);

int i = c;

while (tmp)

{

screen[r][i] = (char)(tmp % 10 + '0');

i--;

tmp /= 10;

if (r > row)

row = r;

}

screen[r][i] = '-';

}

else if (nd->data > 0)

{

T tmp = nd->data;

int i = c;

while (tmp)

{

screen[r][i] = (char)(tmp % 10 + '0');

i--;

tmp /= 10;

if (r > row)

row = r;

}

}

else

screen[r][c] = '0';

if (nd->lchild)

{

screen[r + 1][c - 1] = '/';

OutPutTree(nd->lchild, r + 2, c - 2, screen);

}

if (nd->rchild)

{

screen[r + 1][c + 1] = '\\';

OutPutTree(nd->rchild, r + 2, c + 2, screen);

}

return;

}

1. 问题1

template<class T>

void BinarySearchTree<T>::PostOrder(BinaryTreeNode<T>\* nd, queue<T>& Queue)

{

if (nd == nullptr)

return;

PostOrder(nd->lchild, Queue);

PostOrder(nd->rchild, Queue);

Queue.push(nd->data);

return;

}

template<class T>

void BinarySearchTree<T>::OutPutNext(BinaryTreeNode<T>\* target)

{

queue<T>Queue;

PostOrder(root, Queue);//把树中所有元素存到队列中

while (!Queue.empty() && Queue.front() != target->data)

{

Queue.pop();

}

Queue.pop();

if(Queue.front())

cout << Queue.front() << endl;

return;

}

1. 输出2

for (int i = 1; i <= num; i++)

{

int sum = 0;

for (int j = 1; j <= num; j++)

{

if (input[j] <= input[i])

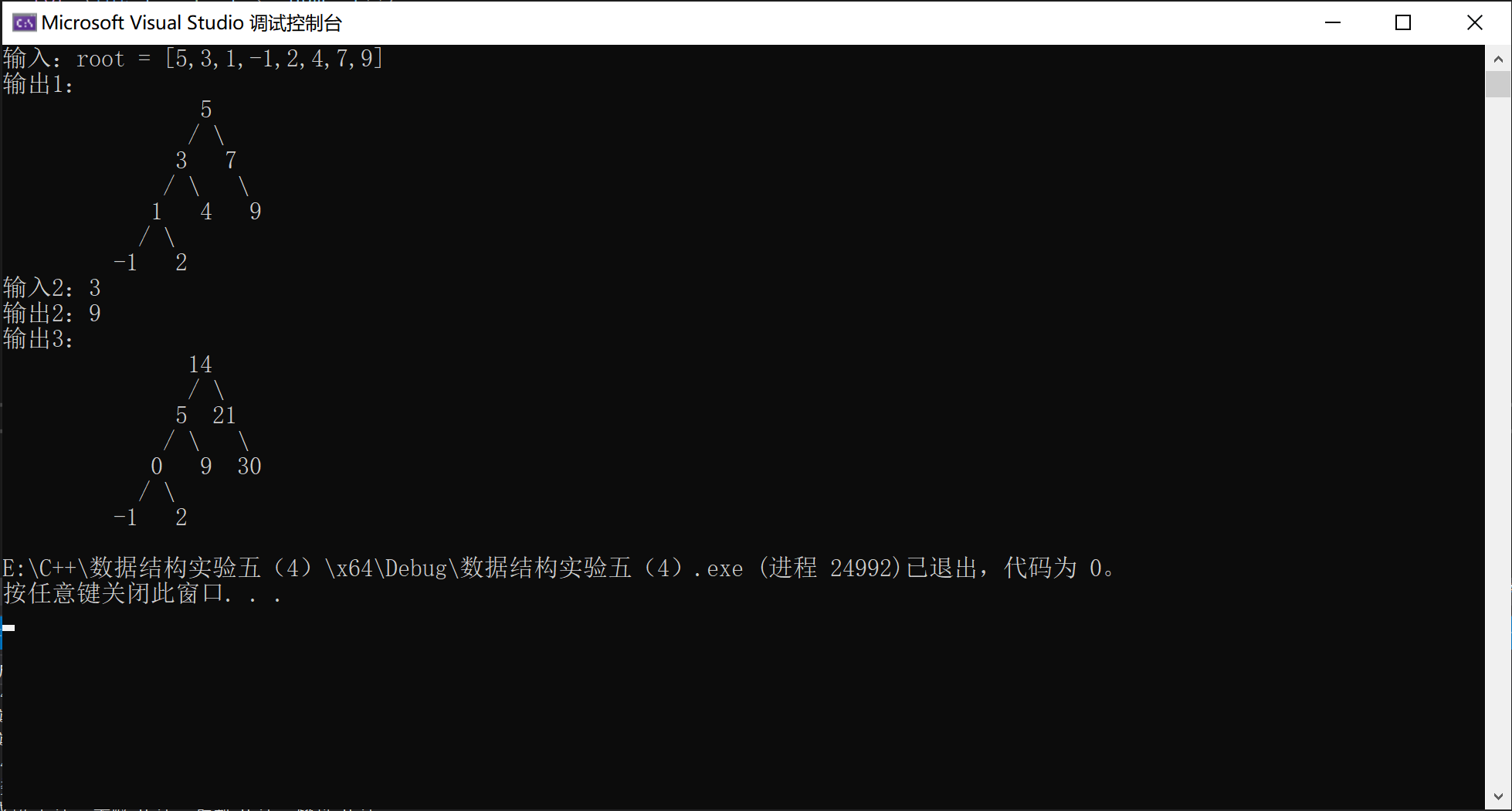
sum += input[j];

}

Sum.Insert(sum);

}

1. 实验结果



1. 复杂度分析
2. 输出1
3. 输出2
4. 输出3