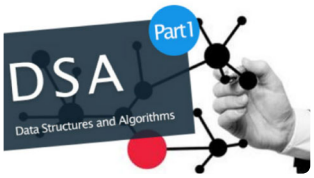


公告
评分标准
课件
测验与作业
考试
讨论区

课程分享
微信提醒课程进度
扫码下载 APP

帮助中心

第五章 二叉树后半部分（5.5~5.7）测验

返回

本次得分为：**36.00/36.00**, 本次测试的提交时间为：**2020-11-06**, 如果你认为本次测试成绩不理想，你可以选择再做一次 。

1 多选 (3分) 下列关于二叉搜索树的说法正确的有 得分/总分
Which sentences of the followings are right about binary search tree:

A. 二叉搜索树一定是满二叉树。A binary search tree must be a full binary tree.

B. 当根结点没有左儿子时，根结点一定是值最小的结点。If the root node doesn't have left child, it must be the node with the smallest value. ✔1.00/3.00

C. 二叉搜索树按照中序遍历将各结点打印出将各结点打印出来，将得到按照由小到大的排列。
If we print a binary search tree's nodes according its infix order, the sequence will be from small to large. ✔1.00/3.00

D. 如果结点x的左子树有右子树，则存在某个结点的值介于结点x的值和x左儿子的值之间，并且这个结点在$$x$$的左子树之中。
If the left child tree of a node x has a right child tree, then there exists some node whose value is between the value of node x and the value of its left child node, and this node is on the left child tree of node x. ✔1.00/3.00

2 多选 (3分) 下列关于堆的说法正确的有: 得分/总分
Which sentences of the followings are right:

A. 使用筛选法建堆要比将元素一个一个插入堆来建堆效率高。Screening method has a higher efficiency than inserting elements one by one while constructing a heap. ✔1.00/3.00

B. 堆一定是满二叉树。A heap must be a full binary tree.

C. 最小堆中，某个结点左子树中最大的结点可能比右子树中最小的结点小。In a minimum heap, the largest value on some node's left child tree could be possibly smaller than the smallest value of its right child tree. ✔1.00/3.00

D. 堆一定是完全二叉树。A heap must be a complete binary tree. ✔1.00/3.00

3 多选 (3分) 下列关于Huffman树和Huffman编码的说法正确的有: 得分/总分
Which sentences of the followings are right about Huffman tree and Huffman code:

A. Huffman树一定是完全二叉树。A Huffman tree must be a complete binary tree.

B. Huffman编码是一种前缀编码。Huffman code is a kind of prefix code. ✔1.50/3.00

C. 对于同样的一组权值两两不同的内容可以得到不同的Huffman编码方案。Different content with the same group of weights can get different Huffman codes. ✔1.50/3.00

D. Huffman编码中所有编码都是等长的。All codes in a Huffman code have the same length.

4 多选 (3分) 得分/总分

一组包含不同权的字母已经对应好Huffman编码，如果某一个字母对应编码001,下面说法正确的有

A group of letters with different weights has corresponded with Huffman codes, if a letter's corresponding code is 001, which sentences of the followings are right:

A. 建好的Huffman树至少包含4个叶结点。The Huffman tree contains at least 4 leaf nodes. ✔1.00/3.00

B. 编码0和00可能对应于其他字母。Code 0 and 00 could corresponding with other letters.

C. 以001开头的编码不可能对应其他字母。A code beginning with 001 couldn't correspond with other letters. ✔1.00/3.00

D. 以01开头和1开头的编码肯定对应某个字母。Codes beginning with 01 or 1 must correspongding with some letters. ✔1.00/3.00

5 填空 (2分) 如果按关键码值递增的顺序依次将**n**个关键码值插入到二叉搜索树中，如果对这样的二叉搜索树进行检索时，每次检索的字符都等概率的从这**n**个关键码值中选取，平均比较次数为多少?

If we insert n key values to a binary search tree successively from small to large, when we search this binary search tree, each time the search character is selected from these n key values with the same possibility, then how many times will the comparison be on average?

得分/总分

(n+1)/2

✔ 2.00/2.00

6 填空 (2分) 从空二叉树开始，严格按照二叉搜索树的插入算法（不进行旋转平衡），逐个插入关键码{18,73,10,5,68,99,27,41,51,32,25}构造出一棵二叉搜索树，对该二叉搜索树按照前序遍历得到的序列为?（答案中每两个元素之间用一个空格隔开）

From a null binary tree, insert key values {18, 73, 10, 5, 68, 99, 27, 41, 51, 32, 25} successively according to the insertion algorithm of a binary search tree strictly (no rotation and balance) to construct a binary search tree. Please write down the sequence of preorder of this binary search tree. (There is one blank space between two elements)

得分/总分

18 10 5 73 68 27 25 41 32 51 99

✔ 2.00/2.00

7 填空 (2分) 从空二叉树开始，严格按照二叉搜索树的插入算法（不进行旋转平衡），逐个插入关键码{18,73,10,5,68,99,27,41,51,32,25}构造出一棵二叉搜索树，对该二叉搜索树按照**后序遍历**得到的序列为?（答案中每两个元素之间用一个空格隔开）

From a null binary tree, insert key values {18, 73, 10, 5, 68, 99, 27, 41, 51, 32, 25} successively according to the insertion algorithm of a binary search tree strictly (no rotation and balance) to construct a binary search tree. Please write down the sequence of post order of this binary search tree. (There is one blank space between two elements)

得分/总分

5 10 25 32 51 41 27 68 99 73 18

✔ 2.00/2.00

8 填空 (2分) 从空二叉树开始，严格按照二叉搜索树的插入算法（不进行旋转平衡），逐个插入关键码构造出一棵二叉树，以怎样的顺序插入关键码集合{14，32，47，6，9，12，78，63，29，81}可以使得树的深度最小?

请依次写出插入到树中的元素，每两个元素之间用一个空格隔开。

得分/总分

如果有多组满足要求的方案，请使得你的答案中先插入的元素尽可能的小。
From a null binary tree, insert key values successively according to the insertion algorithm of a binary search tree strictly (no rotation and balance) to construct a binary search tree. What is the insertion sequence that could make the tree have a smallest depth with a key value set {14，32，47，6，9，12，78，63，29，81}? Please write down the elements successively, and there is one blank space between two elements. If there are more than one answer that meet the condition, please make the element which needs to be inserted first as small as possible in your answer.

12 6 9 47 29 14 32 78 63 81

✔ 2.00/2.00

9 填空 (2分) 对于关键码序列{38，64，52，15，73，40，48，55，26，12}，用筛选法建最小值堆，若一旦发现逆序对就进行交换，共需要交换元素多少次?
For the key value sequence {38，64，52，15，73，40，48，55，26，12}, use the screening method to constuct a minimum heap, if we exchange them when we find reversed order, then how many times should we exchange them?

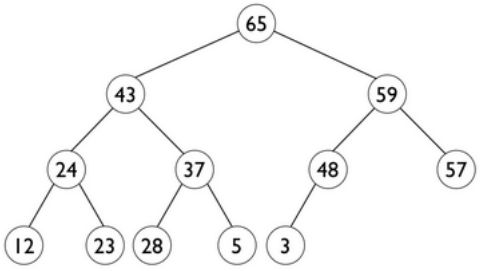
得分/总分

6

✔ 2.00/2.00

10 填空 (2分) 对于如下图所示的最大堆，删除掉最大的元素后，堆的**前序遍历**结果是
For the following maximum heap, after deleting the maximum element, the preorder traversal sequence is
请依次写出插入到树中的元素，每两个元素之间用一个空格隔开。
Please write down the elements successively, and there is one blank space between two elements.

得分/总分

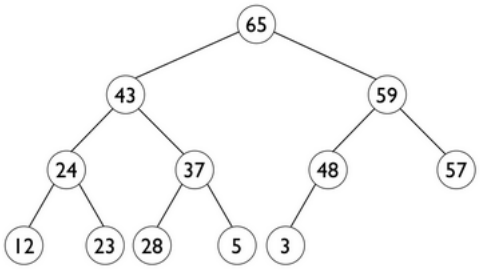


59 43 24 12 23 37 28 5 57 48 3

✔ 2.00/2.00

11 填空 (2分)
对于如下图所示的最大堆，删除掉最大的元素后，堆的**后序遍历**结果是
For the following maximum heap, after deleting the maximum element, the post order traversal sequence is

得分/总分



12 23 24 28 5 37 43 48 3 57 59

✔ 2.00/2.00

12 填空 (2分) 下表展示了在一段文本中每个字母出现的次数。
The frequencies that each letter appears in a paragraph is represented as follow.

得分/总分

a	12
e	8
i	15
o	4
u	9

对于这段文本使用Huffman编码相较使用等长编码能够节约多少比特的空间？ Comparing to use codes that have the same length, how many bits of space could be saved when we use Huffman code for the paragraph?

a	12
e	8
i	15
o	4
u	9

36

✔ 2.00/2.00

13 填空 (2分)

得分/总分

对于给定的一组权**W={1,4,9,16,25,36,49,64,81,100}**，构造一棵具有最小带权外部路径长度的三叉树，写出这棵树的带权外部路径长度。

For a given group of weights W={1, 4, 9, 16, 25, 36, 49, 64, 81, 100}, please construct a ternary tree with a minimum weighted route length and write down this weighted route length.

705

✔ 2.00/2.00

14 填空 (2分) 请阅读下面一段代码

得分/总分

Please read the following code

C++:

```
while (!aStack.empty() || pointer) {?
    while (pointer != NULL) {
        // 1号访问点 //No. 1 visiting point
        element.pointer = pointer; element.tag = Left;
        aStack.push(element);
        pointer = pointer->leftchild();
    }
    element = aStack.top(); aStack.pop();
    pointer = element.pointer;
    if (element.tag == Left) {
        // 2号访问点 //No. 2 visiting point
        element.tag = Right;
        aStack.push(element);
        pointer = pointer->rightchild();
    } else {
        // 3号访问点 //No. 3 visiting point
        pointer = NULL;
    }
}
```

Python:

```python
while not aStack.isEmpty() or pointer:
    while pointer != None:
        # 1号访问点 # No.1 visiting point
        element.pointer = pointer
        element.tag = Left
        aStack.push(element)
        pointer = pointer.leftchild
    element = aStack.pop()
    pointer = element.pointer
    if element.tag == Left:
        # 2号访问点 # No.2 visiting point
        element.tag = Right
        aStack.push(element)
        pointer = pointer.rightchild
    else:
        # 3号访问点 # No.3 visiting point
        pointer = None
```

若此段代码的作用是用来进行**前序遍历**，那么应该在几号访问点进行访问？（只需要填写数字）

if this code is used to do a preorder traversal, which visiting point should be visited? (You only need to write down the number)

1

✔ 2.00/2.00

15 填空 (2分) 请阅读下面一段代码 得分/总分

Please read the following code

C++:

```cpp
while (!aStack.empty() || pointer) {?
    while (pointer != NULL) {
        // 1号访问点 //No. 1 visiting point
        element.pointer = pointer; element.tag = Left;
        aStack.push(element);
        pointer = pointer->leftchild();
    }
    element = aStack.top(); aStack.pop();
    pointer = element.pointer;
    if (element.tag == Left) {
        // 2号访问点 //No. 2 visiting point
        element.tag = Right;
        aStack.push(element);
        pointer = pointer->rightchild();
    } else {
        // 3号访问点 //No. 3 visiting point
        pointer = NULL;
    }
}
```

Python:

```python
while not aStack.isEmpty() or pointer:
    while pointer != None:
        # 1号访问点 # No.1 visiting point
        element.pointer = pointer
        element.tag = Left
        aStack.push(element)
        pointer = pointer.leftchild
    element = aStack.pop()
    pointer = element.pointer
    if element.tag == Left:
        # 2号访问点 # No.2 visiting point
        element.tag = Right
        aStack.push(element)
        pointer = pointer.rightchild
    else:
        # 3号访问点 # No.3 visiting point
        pointer = None
```

若此段代码的作用是用来进行**中序遍历**，那么应该在几号访问点进行访问？（只需要填写数字）

if this code is used to do an infix order traversal, which visiting point should be visited? (You only need to write down the number)

2

✔ 2.00/2.00

16 填空 (2分) 请阅读下面一段代码 得分/总分

Please read the following code

C++:

```
while (!aStack.empty() || pointer) {?
    while (pointer != NULL) {
        // 1号访问点 //No. 1 visiting point
        element.pointer = pointer; element.tag = Left;
        aStack.push(element);
        pointer = pointer->leftchild();
    }
    element = aStack.top(); aStack.pop();
    pointer = element.pointer;
    if (element.tag == Left) {
        // 2号访问点 //No. 2 visiting point
        element.tag = Right;
        aStack.push(element);
        pointer = pointer->rightchild();
    } else {
        // 3号访问点 //No. 3 visiting point
        pointer = NULL;
    }
}
```

Python:

```
while not aStack.isEmpty() or pointer:
    while pointer != None:
        # 1号访问点 # No.1 visiting point
        element.pointer = pointer
        element.tag = Left
        aStack.push(element)
        pointer = pointer.leftchild
    element = aStack.pop()
    pointer = element.pointer
    if element.tag == Left:
        # 2号访问点 # No.2 visiting point
        element.tag = Right
        aStack.push(element)
        pointer = pointer.rightchild
    else:
        # 3号访问点 # No.3 visiting point
        pointer = None
```

若此段代码的作用是用来进行**后序遍历**，那么应该在几号访问点进行访问？（只需要填写数字）

if this code is used to do an post order traversal, which visiting point should be visited? (You only need to write down the number)

3

✔ 2.00/2.00