中山大学软件学院软件工程专业 **2017** 级 (2018 学年秋季学期)

《SE-203 数据结构》期末试题 (B 卷)

(考试形式：闭卷　　考试时间：2 小时)

《中山大学授予学士学位工作细则》第六条

考试作弊不授予学士学位

方向：_____　　**姓名：**_____　　**学号：**_____

说明：答案一律写在答题纸上，且夹在试卷里一起交还给监考老师。

I. Selection with only one choice (每小题 2 分，共 30 分)

1. In the perspective of physical structure, we can classify data structures into (　　).
   A. Dynamic structure and static structure
   B. Internal structure and external structure
   C. Contiguous structure and linked structure
   D. Compact structure and non-compact structure

2. In computer science, the time complexity is one of important properties of algorithms. If the time complexity of algorithm A for problem B is $O(1)$, it means that (　　).
   A. Algorithm A can solve problem B with 1 step
   B. Algorithm A spends 1 second to solve problem B
   C. Algorithm A can solve problem B with constant steps
   D. Algorithm A can solve problem B with nondeterminisitc steps

3. Class stack is a contiguous stack, and its elements are stored in array entry[N]. If the Top index of the stack is assigned -1 by constructor, how many elements has the stack at any time? (　　)
   A. Top　　B. Top+1　　C. Top−1　　D. N

4. Given a contiguous list with length n, the time complexity of locating a given position i ($0 \le i \le n-1$) is (　).
   A. $O(1)$　　B. $O(i)$　　C. $O(\log n)$　　D. $O(n)$

5. Suppose the sequence of data that is pushed into a stack is 1, 2, 3, 4. Which of the following sequence that is poped from the stack is impossible? (　　)
   A. 1 2 3 4　　B. 2 1 4 3　　C. 2 3 1 4　　D. 4 1 3 2

6. Suppose we use an array with length MaxQ to implement a circular queue, the rear and front point the tail and front element respectively. How many elements are there in the circular queue? (　　)
   A. rear−front　B. rear−front+1　C. rear−front+1+MaxQ　D. (rear-front+1+MaxQ)%MaxQ

7. Which of the following statements about strings is correct? ( )

A. A string is a sequence of characters

B. An empty string is composed of space characters (空格)

C. A string can only be stored in sequential structures (e.g. arrays)

D. If we want to get a real 123.5, the input 123.5 is only the real, not a string

8. Assume the infix form (中缀形式) of an arithmetic expression is A*B+C/D-E, and the postfix form (后缀形式) is ABC+*DE-/. What is the prefix form(前缀形式) of this expression? ( )

A. -A+B*C/DE B. /*A+BC-DE C. -+/CD*ABE D. -+*AB/CDE

9. Let undirected graph G=(V, E) be stored with adjacency matrix(邻接矩阵) where V is set of vertices and E is the set of edges. If $(v_i, v_j) \in E$, $v_i \neq v_j$. How many zeros are there in the adjacency matrix? ( )

A. $|V|^2-2|E|$ B. $|V|^2-|E|$ C. $2|E|$ D. $|E|$

10. The lower bound of sorting a sequence based on comparison with n elements is ( ).

A. $O(1)$ B. $O(\log n)$ C. $O(n)$ D. $O(n\log n)$

11. Which of the following constraint condition must AVL tree satisfy where T_L and T_R are the left and right subtrees respectively and $H(T)$ is the height of binary tree T?

A. $H(T_L)-H(T_R)<1$ B. $H(T_R)-H(T_L)<1$ C. $H(T_L)-H(T_R)\leq 1$ D. $|H(T_L)-H(T_R)|\leq 1$

12. Suppose that the levels of nodes in a binary tree count from 0, the levels of children of a node at i^{th} level are i+1. For example, the level of the root of the binary tree is 0, the levels of children of the root is 1. How many nodes are there at most at k^{th} level? ( )

A. 2^k B. k^2 C. $2k$ D. k

13. For the following sorting algorithms, which time complexity is $O(n\text{^}2)$ in the worst case? ( )

A. Head Sort B. Quick Sort C. Merge Sort D. Radix Sort

14. For the following operations, which can the queue not do it? ( )

A. Save the last element B. Remove the last element

C. Get the first element D. Remove the first element

15. Suppose that an array Data[0..N-1] is a heap. Which is the left child of Data[k] (k<N/2)? ( )

A. Data[(k+1)/2] B. Data[k+1] C. Data[2k+1] D. Data[2k+2]

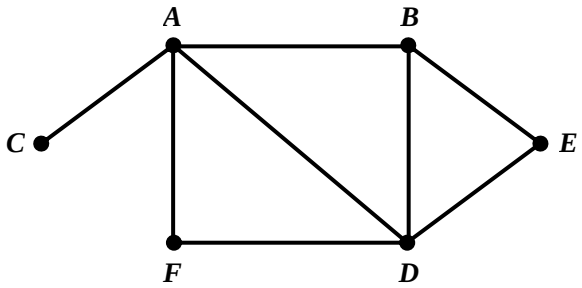
II. Questions and Answers (每小题 15 分，共 45 分)

1. In the following sequence of keys, insert the keys, in the order shown, to build them into an AVL tree, please draw the illustration figures for the whole procedure.

A, T, G, S, H, N

2. (1) Give the adjacency matrix for the following undirected graph.

(2) Suppose that the graph traversal start at vertex *A*, write the order of vertices visited and draw the traversal tree under Depth-First traversal. The nodes that is adjacency to the same node are visited in alphabetical order. (同与一个结点相邻的结点按字母序的顺序进行访问)



3. Describe QuickSort algorithm briefly, and show each step for sorting the following data into ascending sequence (升序) by QuickSort algorithm. Suppose that the first element is pivot.

Seven unsorted data: 4, 6, 3, 2, 1, 5, 7

III. Programming (共 25 分)

1. Given a class of Binary Search Tree declared as follows (10 分)

```
template <class Record>
struct Binary_node {    //some member functions are neglected, since they will not be used here.
Record data;
Binary_node< Record > *left;
Binary_node< Record > *right;
};

template <class Record>
class Search_tree: {
public:     //some member functions are neglected, since they will not be used here.
        Error_code tree_search(Record &target) const{
          Error_code result = success;
          Binary_node<Record> *found = search_for_node(root, target);
          if (found == NULL)
                result= not_present;
          else
             target = found->data;
        return result; };
```

```
private:
    Binary_node <Record> *root;
    search_for_node( Binary_node<Record> *sub_root, const Record &target) const;
}
```

Implement the function, search_for_node(Binary_node<Record> *sub_root, const Record &target) const, (1) in a recursive version, and (2) in a non-recursive version.

2. The Node structure and class Set are defined as following: (15 分)

```
struct   Node {
         int       Key;
         struct    Node *next;
         Node(int key, Node *Next=NULL) { Key = key; next = Next; };
};
class Set {
public:
         ……
         void    operator –= (const Set &Src);
         int       Counter();
         ……
private:
         Node    *Head;
}
```

Write programs for operator –= and method Counter(). Their descriptions are the following:

(1) –=: “S1 –= S2” means that S1=S1−S2 where S1 and S2 are objects of Set, ‘–’ is the difference of two sets. $x \in$ S1−S2 if and only if $x \in$ S1 and $x \notin$ S2.

(2) Counter(): It returns the number of elements in Set. It gets 0 if the set is empty.

参 考 答 案 (B)

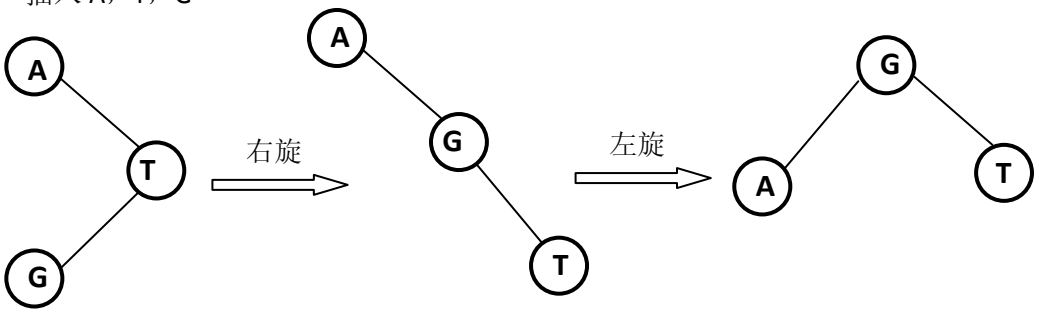
I. 单项选择 (每小题 2 分，共 30 分)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C	C	B	A	D	D	A	B	A	D	D	A	B	B	C

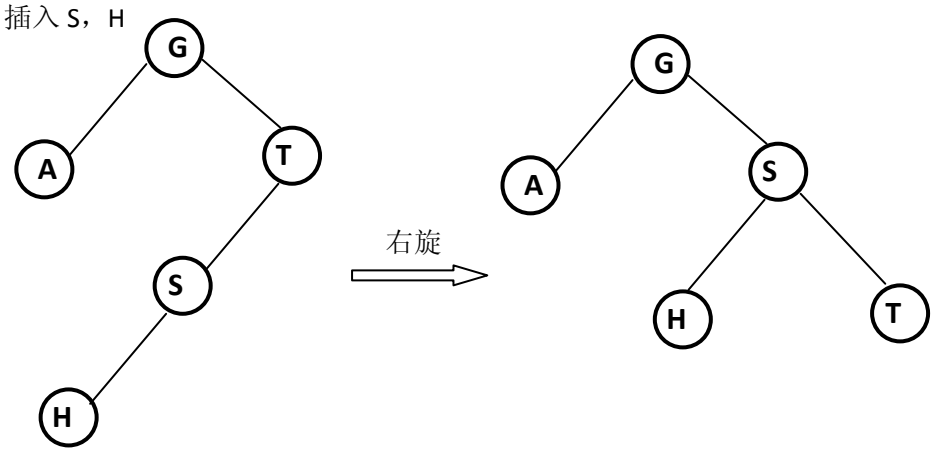
II. 解答题 (每小题 15 分，共 45 分)

1. AVL

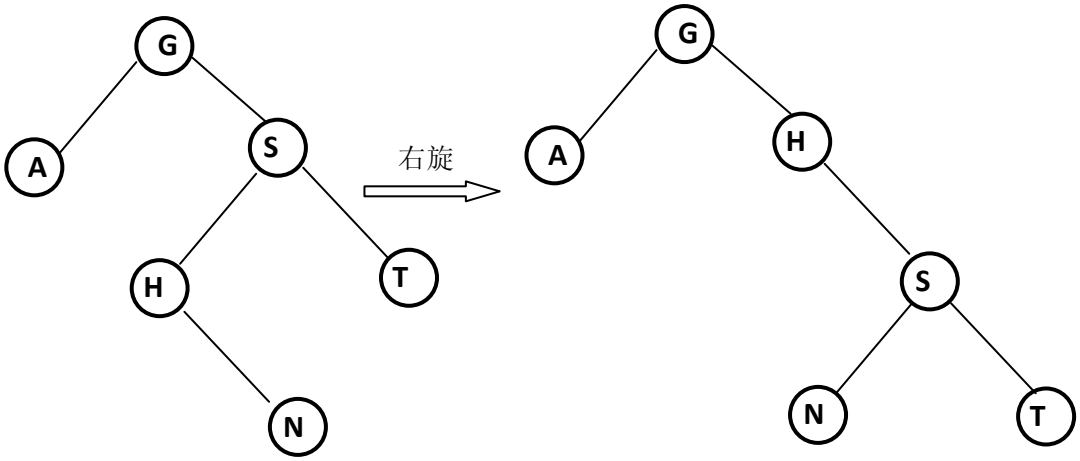
插入 A，T，G

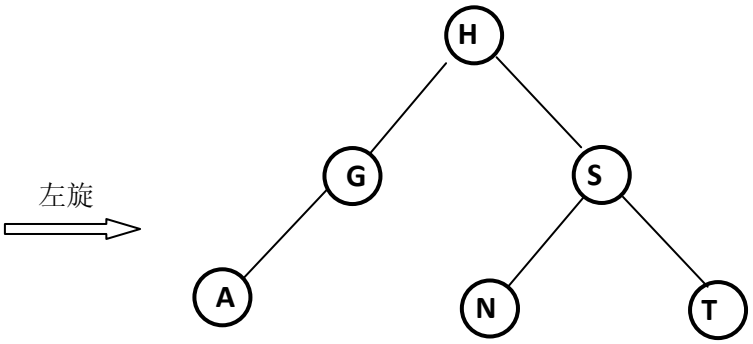


插入 S，H



插入 N

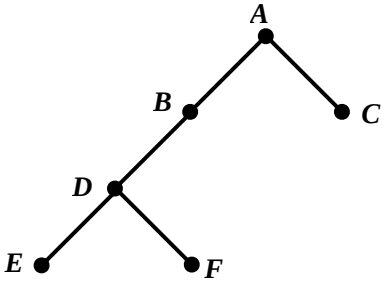




2. 无向图的邻接矩阵如下：

	A	B	C	D	E	F
A	0	1	1	1	0	1
B	1	0	0	1	1	0
C	1	0	0	0	0	0
D	1	1	0	0	1	1
E	0	1	0	1	0	0
F	1	0	0	1	0	0

深度优先搜索树如下：



3. 快速排序算法是一种“分而治之”的排序思想。假设待排序的数据存储在 Data[low..high]之中，其排序思想如下：

(1). 如果待排数据的区间是错误的，则排序结束；

(2). 如果 high–low+1<2，则排序结束；

(3). 根据支点的选择策略确定一个支点；

(4). 利用支点把待排数据分成“比支点小的部分”、支点(m 是支点的位置)和“比支点大的部分”；

(5) 递归地对待排数据 Data[low..m–1]和 Data[m+1,high]进行快速排序；

(6) 已排好序的数据 Data[low..m–1]、Data[m+1,high]和支点 Data[m]组合在一起，得到到一个已排好序的数据 Data[low..high]。

对题中给定数据的排序步骤如下：

第一次支点：4

第一次划分: 1, 3, 2, 4, 6, 5, 7

第二次支点: 1

第二次划分: 1, 3, 2

第三次支点: 3

第三次划分: 1, 2, 3

第四次支点: 6

第四次划分: 5, 6, 7

最终结果: 1, 2, 3, 4, 5, 6, 7

如果上述步骤表述基本正确，就算解答正确。

III. 编程题 (共 25 分)

1. 参考程序

(1) 递归版本

```
template <class Record>
Binary_node<Record> *Search_tree<Record>::
                    search_for_node( Binary_node<Record> *sub_root,
                                                    const Record & target) const
{
        if (sub_root == NULL || sub_root->data == target)
                return sub_root;
        else if (sub_root->data < target)
                return search_for_node(sub_root-> right, target);
        else return search_for_node(sub_root->left, target);
}
```

(2) 非递归函数

```
template<class Record>
Binary_node<Record> *Search_tree<Record> search_for_node(Binary_node<Record>
*sub_root, const Record &target)
{
while(sub_root != NULL && sub_root->data != target)
{
   if(sub_root->data < target)
         sub_root = sub_root->right;
   else
         sub_root = sub_root->left;
}

   return sub_root;
}
```

2. 参考程序

(1)

```
void Set::operator -= (const Set &Src)
{
   Node *head, *Pt, *Pt1, *Pt2;
   head = NULL;
   for (Pt = Head; Pt != NULL; Pt = Pt->next) {
        for (Pt2 = Src.Head; Pt2 != NULL; Pt2 = Pt2->next)
             if (Pt->Key == Pt2->Key) break;
        if (Pt2 != NULL) continue;
        Pt2 = new Node(Pt->Key);
        if (Pt2 != NULL) {
           if (head == NULL) head = Pt2;
           else Pt1->next = Pt2;
           Pt1 = Pt2;
        }
   }
   while (Head != NULL) { Pt = Head->next;      delete Head;    Head = Pt; }
   Head = head;
}
```

(2)

```
int Set:: Counter ()
{
   int count = 0;
   for (Node *Pt = Head; Pt != NULL; Pt = Pt->next)
      count++;
   return count;
}
```