中 山 大 学 软 件 学 院 **2008** 级 软 件 工 程 专 业 (2009 秋季学期)

《数据结构》期末试题答案 (B)

1. Queue is a common scene in many places such as at bus station. When a bus arrives, the first person come into the bus, and the next person in line will come up.

The Queue class imitates the behavior of lineup in real world, its partial definition follows:

```
#define    MAXQUEUE     20
class Queue {
public:
        Queue() { Rear = -1; }
        ……
        bool    Append(int data);
        bool    Serve(int &data);
        bool    Full();
        ……
private:
        int    Rear;
        int    Entry[MAXQUEUE];
}
```

(15 *points*)

(1) Write a program for method Append(int data), it appends the data to the queue.

If the operation successes, it will return true, otherwise the false will be returned.

(2) Write a program for method Serve(int &data), it saves the data of the front of the queue into reference parameter data, and then removes the front of the queue.

If the operation successes, it will return true, otherwise the false will be returned.

(3) Write a program for method Full(), it checks whether the queue is full.

If the queue is full, it will return true, otherwise the false will be returned.

参考答案 (在编程部分，任何等价的语句或编程思路都算正确，以后同此)

(1)

```
bool    Queue::Append(int data)
{
   if (Rear >= MAXQUEUE-1) return false;
   Rear++;
   Entry[Rear] = data;
   return true;
}
```

(2)

```
bool    Queue::Serve(int &data)
{
   if (Rear == -1) return false;
   data = Entry[0];
   for (int i = 0; i < Rear; i++) Entry[i] = Entry[i+1];
   Rear--;
   return true;
}
```

(3)

```
bool    Queue::Full()
{
   return (Rear==MAXQUEUE-1);
}
```

2. The binomial coefficient $C(m, n)$ may be defined by the following recurrence relation:

$$C(m, n)=\begin{cases}1 & n = 0\\ 1 & n = m\\ C(m\text{-}1, n) + C(m\text{-}1, n\text{-}1) & n < m\end{cases}$$

(1) Write the recursive function Combination_R(int m, int n) to generate $C(m, n)$ by the foregoing formula.

(2) write the non-recursive function Combination(int m, int n) to generate $C(m, n)$ by basic mathematical formula. **(15 *points*)**

参考答案

(1)

```
int combination_R(int m, int n)
{
   if (n == 0 || n == m) return 1;
   if (n == 1) return m;
   return combination_R(m-1, n-1) + combination_R(m-1, n);
}
```

(2) $C(m, n) = m!/(n!\times(m\text{-}n)!) = m(m\text{-}1)\ldots(m\text{-}n\text{+}1) / n!$

```
int combination(int m, int n)
{
   int c1, c2, i;
   c1 = 1;
```

```
    for (i = m; i > m-n; i--) c1 *= i;
    c2 = 1;
    for (i = n; i > 1; i--) c2 *= i;
    return c1/c2;
}
```

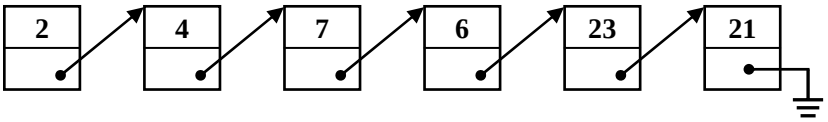
3. The Node structure and List class are defined as following: **(10 *points*)**

```
struct   Node {
        int     Key;
        struct   Node *next;
};
class List {
public:
        ……
        void   reverse();
        ……
private:
        Node   *Head;
}
```

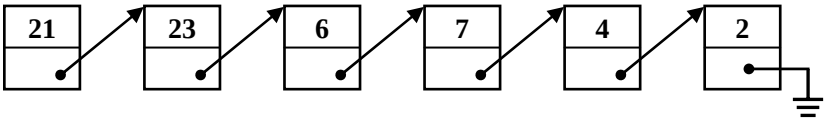
Write a program for method reverse(), it reverses the order in which the elements occur in a list. Don't apply any new nodes except some node pointers.

For example

Before operation:



After operation:



参考答案

```
void List::reverse()
{
    Node *Pt1, *Pt2;
    Pt1 = Head;
    Head = NULL;
    while (Pt1 != NULL) {
       Pt2 = Pt1;
       Pt1 = Pt1->next;
```

```
        Pt2->next = Head;
        Head = Pt2;
      }
    }
```

4. Describe Heapsort algorithm briefly. Suppose the following five elements are saved in array and the array is an initial heap, show each step for sorting these data by Heapsort algorithm.

   Five unsorted data: 5, 4, 2, 3, 1 **(10 *points*)**

参考答案

堆排序算法的主要思想：

(1) 用数组存储待排序的数据，利用堆结构的特点，根据排序顺序(“从小到大”，还是“从大到小”)对待排数据进行堆化操作；

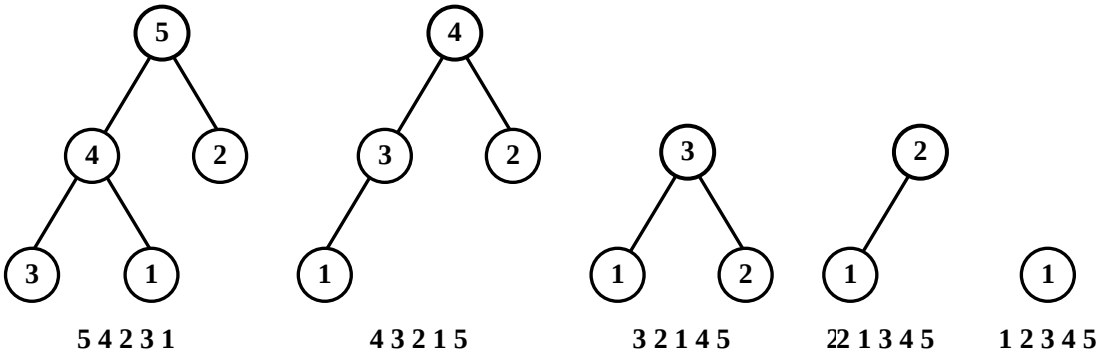
(2) 对堆顶数据和堆中最后数据进行交换，并减少堆的大小；

(3) 调整新的排序数据，使之满足相应堆的要求；

(4) 重复步骤(2)和(3)，直到只有一个待排数据为止；

(5) 这时存储在数组中的数据就是按要求排好序的。

对 5 个待排数据：5, 4, 2, 3, 1，其用堆排序的排序过程如下：



所以，这时的排序结果为：1, 2, 3, 4, 5

5. The following table is a hash table with 13 elements. It uses quadratic probing for resolving collisions. The hash function $h(key)$ and quadratic probing function h_i are defined as following:

0	1	2	3	4	5	6	7	8	9	10	11	12
	82	30	56			47		62	50	90		

$h(key) = (key + 11) \% 13$

$h_i = (h(key) + 13 + d_i) \% 13$, $d_i = 1, -1, 4, -4, \ldots, i^2, -i^2, \ldots$

(1) Insert key 69 into the hash table and list each formula and computing result for insert.

(2) Search key 88 and list each formula and computing result for search. **(15 *points*)**

参考答案

(1)

$h(69) = (69 + 11) \% 13 = 2$ // 冲突

$h_1 = (2 + 13 + 1) \% 13 = 3$ // 冲突

$h_2 = (2 + 13 - 1) \% 13 = 1$ // 冲突

$h_3 = (2 + 13 + 4) \% 13 = 6$ // 冲突

$h_4 = (2 + 13 – 4) \% 13 = 11$ // 不冲突

哈希(hash)表中第 11 个项为空，所以，数据 69 可存入其中。哈希表的插入数据操作结束。

(2)

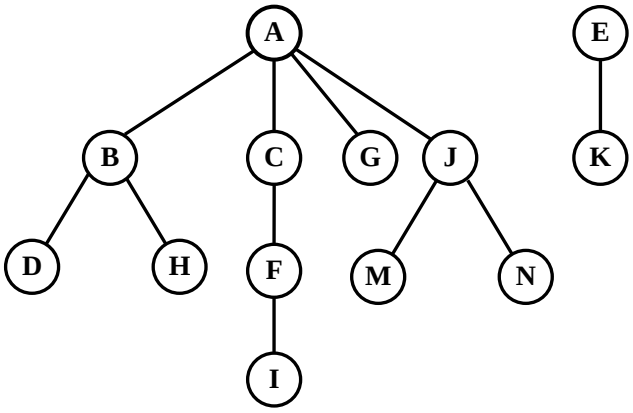
$h(88) = (88 + 11) \% 13 = 8$ // 哈希表的第 8 项数据不是 88

$h_1 = (8 + 13 + 1) \% 13 = 9$ // 哈希表的第 9 项数据不是 88

$h_2 = (8 + 13 - 1) \% 13 = 7$

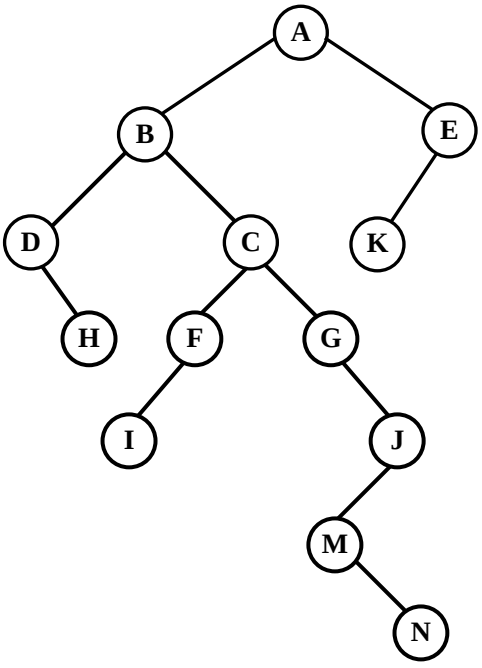
因为哈希表第 7 项为空，所以，待查的 88 一定不在哈希表。查找过程结束。

6. According to the natural correspondence between binary tree and orchard (ordered forest), convert the following ordered forest into a binary tree.
Write the node order under preorder traversal for the binary tree. **(10 *points*)**



参考答案

本题有序森林转换所对应的二叉树如右图所示。

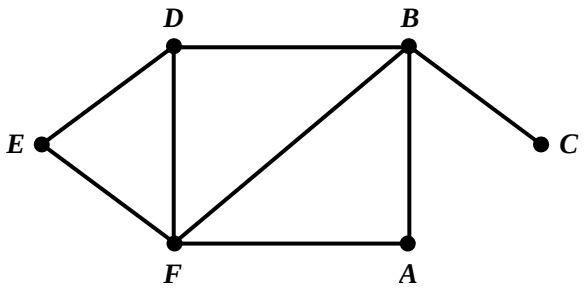


转换后得到的二叉树的先序遍历结点序列为：

A B D H C F I G J M N E K

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

(2) Suppose that the graph traversal start at vertex *E*, **write** the order of vertices visited and **draw** the traversal tree under depth-first traversal. **(15 *points*)**



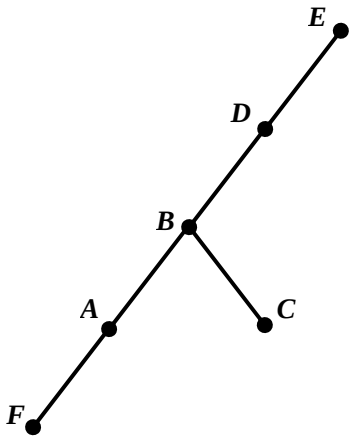
参考答案

邻接矩阵：

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

从结点 *E* 出发，用深度优先遍历图时，其结点输出序列：E, D, B, A, F, C。

深度优先遍历时的遍历树如右下图所示。



8. The BiNode structure and binary_search_tree class are defined as following: **(10 *points*)**

```
struct   BiNode {
         int     Key;
         struct BiNode *LChild, *RChild;
};
class BSTree {
public:
         ……
```

```
        bool    Insert(const int &key);
        ……
    private:
        BiNode    *Root;
    }
```

The binary search tree is a binary tree that is either empty or in which the data entry of every node has a key and has two properties:

(1). The key of the left child of a node (if it exists) is less than the key of its parent node.

(2). The key of the right child of a node (if it exists) is greater than the key of its parent node.

Write a program for the member function Insert(const int &key) of class BSTree. The function inserts the key into the binary search tree.

If the key is inserted successfully, it will return true, otherwise the false will be returned.

Notice: Any private member function can be defined if you think it is necessary.

参考答案

```
    class BSTree {
    public:
        ……
        bool    Insert(const int &key);
        ……
    private:
        bool    Insert(BiNode* &root, const int &key);    // 填加的私有成员函数
        BiNode    *Root;
    }
bool    BSTree::Insert(const int &key)
{
    return Insert(Root, key);
}
bool    BSTree::Insert(BiNode* &root, const int key)
{
    if (root == NULL) {
        BiNode *Pt = new BiNode;
        if (Pt == NULL) return false;
        Pt->Key = key;
        Pt->LChild = Pt->RChild = NULL;
        root = Pt;
        return true;
```

```
    }
    if (key == root->Key) return false;
    if (key < root->Key)) return Insert(root->LChild, key);
    return Insert(root->RChild, key);
}
```