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(B卷)

**(考试形式：闭卷   考试时间：2 小时)**

警示

《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：　学院各方向　　　　　　姓名：　　　　　　　　学号：　　　　　　

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.

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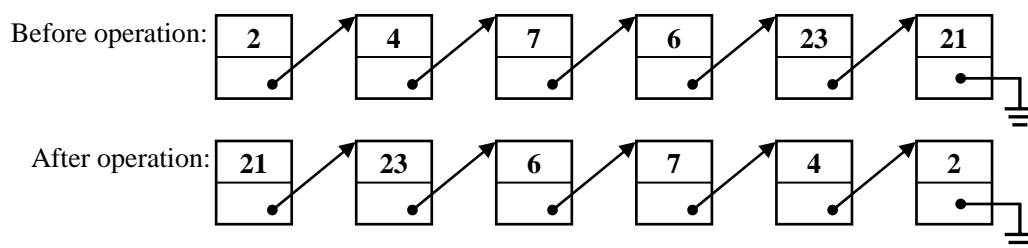
3. The Node structure and List class are defined as following: **(10 *points*)**

```
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:



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*)**

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**)

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**)
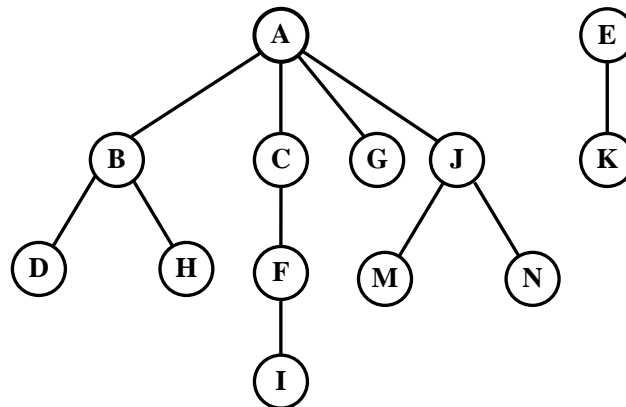


7. (1) Give the adjacency matrix for the following undirected graph.

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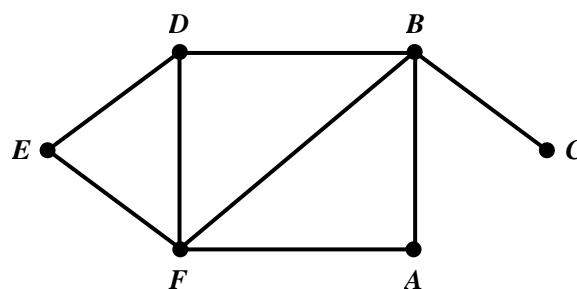
8. The BiNode structure and binary_search_tree class are defined as following:     (**10** *points*)

```
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.

*<u>Notice</u>: Any private member function can be defined if you think it is necessary.*