ONE. Single-Choice

1．An algorithm(算法) is referred to (　　).

A．a calculating method

B．a sorting method

C．a sequential set of instructions to solve a problem

D．a searching method

2．The Linked List is designed for conveniently (　　) data item.

A．getting　　B．inserting

C．finding　　D．locating

3．In the four choices, (　　) is not the properties of algorithm.

A．Acceptability(有穷性)　　B．Determinism(确定性)

C．Feasibility(可行性)　　D．Cyclicity(周期性)

4．Assume a sequence list as 1 ,2 ,3 ,4 ,5 ,6 is pushed in a stack(栈), an impossible output sequence list is (　　).

A．2 ,4 ,3 ,5 ,1 ,6　　B．3 ,2 ,5 ,6 ,4 ,1

C．1 ,5 ,4 ,6 ,2 ,3　　D．4 ,5 ,3 ,6 ,2 ,1

5．A queue(队列) is a structure that follows the principle of (　　).

A．First-In/First-Out　　B．First-In/Last-Out

C．Last-In/First-Out　　D．Random In and Out

6．Removing the data item at index i in an array with n items, (　　) items need to be shifted left one position.

A．n-i　　B．n-i+1

C．i　　D．n-i-1

7．There is an algorithm with inserting an item to a ordered SeqList(顺序链表) and still keeping the SeqList ordered. The computational efficiency of this inserting algorithm is (　　).

A．*O*(log2n)　　B．*O*(1)

C．*O*(n)　　D．*O*(n2)

8．The addresses which store Linked List (　　).

A．must be sequential　　B．must be partly sequential

C．must be no sequential　　D．can be sequential or discontinuous

9．According to the definition of Binary Tree, there will be (　　) different Binary Trees with 3 nodes.

A．6　　B．5

C．4　　D．3

10．A Binary Tree will have (　　) nodes on its level i at most.

A．2i　　B．2^i

C．2^{i+1}　　D．2^{i-1}

11．In the following sorting algorithm, (　　) is an unstable(不稳定) algorithm.

A．the insertion sort(插入排序)　　B．the bubble sort(气泡法排序)

C．quicksort (快速排序)　　D．mergesort(归并排序)

12．Assume that there is an ordered list consisting of 100 data items, using binary search(二分法查找) to find a special item, the maximum comparisons is (　　).

A．25　　B．1

C．10　　D．7

13．The result from scanning a Binary Search Tree (二叉排序树) in inorder traversal is in (　　) order.

A．descending or ascending　　B．descending

C．ascending　　D．out of order

14．To connect n vertices in an undirected graph, it needs (　　) edges at least.

A．n　　B．n-1

C．n+1　　D．1

15．In an undirected graph with n vertexes, the maximum edges is (　　).

A．n(n+1)/2　　B．n(n-1)/2

C．n(n-1)　　D．n^2

16．The output from scanning a minimum heap(小顶堆) with level traversal algorithm(　　).

A．must be an ascending sequence.

B．must be descending sequence.

C．must have a minimum item at the head position.

D．must have a minimum item at the rear position.

17．When a recursive algorithm (递归算法) is transformed into a no recursive algorithm, a structure (　　) is generally used.

A．SeqList　　B．Stack

C．Queue　　D．Binary Tree

18．In the following data structure, (　　) is non-linear structure.

A．Binary Tree　　B．Stack

C．Queue　　D．SeqList

19．A circular queue(循环队列) is empty if (　　).

A．(rear+1)% Maxsize == front　　B．front == rear

C．rear+1 == front　　D．(rear-1)% Maxsize == front

20．The difference between static sorting table(静态查找表) and dynamic sorting table (动态查找表) is (　　).

A．the difference in logical structure

B．the difference in storage structure

C．the difference of data type

D．insertion and deletion only can be done in dynamic sorting table

TWO．Blank filling questions

1．A connected graph has 【1】 component(s).

2．In a complete binary tree,

the sequence number of node i's parent (if exist) is 【2】

the sequence number of node i's left child (if exist) is 【3】

the sequence number of node i's right child (if exist) is 【4】

3．【5】 is the fastest known sorting algorithm in practice.

4．A full binary tree of a given height h(h>=1) has 【6】nodes.

5．An undirected graph G has N vertices. The number of edges of a MST (最小生成树)of this graph is【7】.

6．Commonly used graph search methods are 【8】 and 【9】.

7．Complete the common queue operations.

```
#include    "stdio.h"
#include    "stdlib.h"
#include    "malloc.h"
typedef    int    Status ;
#define    MAXSIZE    100          //capacity of the queue
typedef    struct
{    //point to an array which stores elements of the queue
     int    * base ;
     int    front ;              //front index
     int    rear ;               //rear index
} SqQueue ;
Status    InitQueue_Sq ( SqQueue    &Q )
{
     Q.base = ( int * )malloc ( MAXSIZE * sizeof ( int ) ) ;
     if ( !Q.base )         exit ( 0 ) ;
     Q.front = Q.rear = 0 ;
      return    OK ;
```

```
} //InitQueue_Sq
Status EnQueue_Sq ( SqQueue    &Q , int    e )
{    if (_____【10】_____) //Is the queue full?
          return    ERROR ;
     Q.base[ Q.rear ] = e ;
     _____【11】_____;
     return    OK ;
} //EnQueue_Sq
Status DeQueue_Sq ( SqQueue    &Q, int &e )
{
     if ( Q.rear == Q.front )
          return    ERROR ;
     e = Q.base[Q.front] ;
     _____【12】_____;
     return    OK ;
} //DeQueue_Sq
```

THREE. Answer True or False for each of the following assertions.

( ) 1. The Minimum Spanning Tree of the graph is always unique.

( ) 2. In tree structure, there is exactly one path from the root to each node.

( ) 3. For every node in an AVL tree (平衡二叉树), the height of the left and right sub-trees can differ by at most 1.

( ) 4. Max Heap(大顶堆) can locate the current maximum in O(1) time.

( ) 5. For any non-empty tree, if n is the number of nodes and e is the numbers of edges then n=e+1.

( ) 6. If a binary search tree has l_i leaves at hight i(i>=1), then $l_i / 2^i <= 1$

( ) 7. The topological sorted order (拓扑排序) of nodes in a graph (if it exists) is always unique.

( ) 8. A stack is a linear list in which all additions are restricted to one end, called the top. A stack is also called a FIFO list.

( ) 9. The array { 23, 17, 14, 6, 13, 10, 1, 5, 7, 12 } is a heap (堆).

( ) 10. We can use Time Complexity and Space Complexity to evaluate the efficiency of an algorithm.

FOUR. Assume that we are using the hashing function hash(key) = key mod 11 and the following sequence of keys create a hash table: 12, 44, 13, 88, 23, 94, 11, 39, 20, 16, 5. Show

the resulting hash table: Use linear probing (线性探测) with an increment of 1.

FIVE. You are given an undirected graph（fig.1）.

a. Give adjacency matrix(邻接矩阵) representation of this graph.

b. Construct a minimum spanning tree, using the Kruskal algorithm(克鲁斯卡尔算法). Draw the progress of creation.

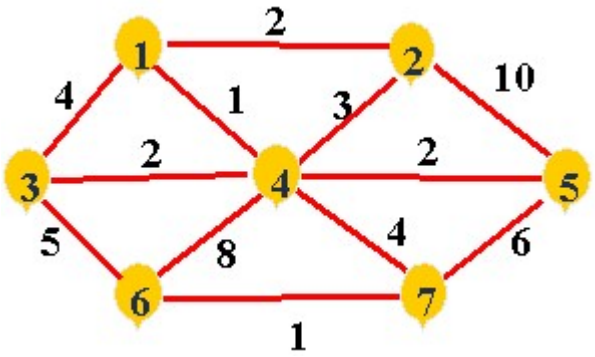

fig.1

SIX. Suppose the inorder traversal sequences of a binary tree is HDBIEAFJCGK，the postorder traversal sequences is HDIEBJFKGCA. Please construct the binary tree，show the final result and write out the preorder traversal sequence of the binary tree.

SEVEN. First step, show the result of inserting 40, 22, 53, 99, 37, 13, 61, 80, 70, 24 into an initially empty binary search tree. Then show the result of deleting the root.

EIGHT. The frequency of the symbols { a, b, c, d, e, f } is { 0.07, 0.09, 0.12, 0.22, 0.23, 0.27 }. Construct the Huffman tree and give the Huffman code of the symbols { a, b, c, d, e, f }.

NINE. Get the program running results.

1．Operation of stack

```
//Suppose Stack and corresponding functions already be corrected defined
 void main( )
 {
      Stack    S ;
      char    x, y ;
      InitStack( S ) ;
      x='c' ;              y='t' ;
      Push(S, x) ;       Push(S,'u') ;       Push(S, y) ;
```

```
        Pop(S, x) ;      Push(S,'r') ;      Push(S, x) ;
        Push(S, y) ;     Pop(S, y) ;        Push(S,'s') ;
        while( !StackEmpty( S ) )   {   Pop( S, y ) ;      printf(y) ;   }
        printf( x ) ;
    }
```

2．Operation of a binary tree.

```
#include   <stdio.h>
#define   LeftChild( i )   ( 2 * ( i ) + 1 )
void fun1( int   A[ ], int   i, int   n )
{   int   Child ;
    int   Tmp ;
    for( Tmp = A[ i ]; LeftChild( i ) < n; i = Child )
    {   Child = LeftChild( i ) ;
        if( Child != n-1 && A[Child+1] > A[Child] )     Child++ ;
        if( Tmp < A[Child]) A[i] = A[Child];
        else   break ;
    }
    A[ i ] = Tmp ;
}
void fun( int A[ ], int n )
{   int   i, temp ;
    for( i = n / 2; i >= 0; i-- )
        fun1( A, i, n ) ;
    for( i = n - 1; i > 0; i-- )
    {   temp = A[0], A[0] = A[i], A[i] = temp ;
        fun1( A, 0, i ) ;
    }
}
#define N 15
void main( )
{   int   a[N]={ 5, 9, 3, 2, 99, 8, 7, 1, 77, 54, 23, 12, 88, -6, -10 } ;
    int   i ;
    fun( a, N ) ;
    for( i=0; i<N; i++ )
        printf( "%d ", a[ i ] ) ;
}
```

TEN. Programming (All methods have been declared in textbook can be used directly, or you can rewrite them if they are not same in your answer)

Assume there are two ascending ordered lists L1 and L2, please merge L1 and L2 into a new list L3. There will be no duplicate(重复的) items in L3. Then please reverse the L3 into a descending ordered list.