**1. Explain the algorithm analysis framework; how to analyze algorithms; how to design algorithms.**

**Algorithm analysis framework:**

(1) Measuring an input’s size

(2) Measuring running time

(3) Orders of growth (of the algorithm’s efficiency function)

(4) Worst-case, best-case and average-case efficiency

**Explain:** Both time and space efficiencies are measured as functions of input size. Time efficiency is measured by counting the number of basic operations executed in the algorithm. The space efficiency is measured by the number of extra memory units consumed.

The framework’s primary interest lies in the order of growth of the algorithm’s running time (space) as its input size goes infinity.

The efficiencies of some algorithms may differ significantly for inputs of the same size. For these algorithms, we need to distinguish between the worst-case, best-case and average case efficiencies.

**Algorithm Design and Analysis Process：**

(1) Understand the problem

(2) Decide on: computational means, exact vs. approximate solving, data structure, algorithm design technique

(3) Design an algorithm:

Brute force, Divide and conquer, Decrease and conquer, Transform and conquer, Space and time tradeoffs, Greedy approach, Dynamic programming, Backtracking, Branch and bound

(4) Prove correctness

(5) Analyze the algorithm:

a. Correctness

b. Time efficiency

c. Space efficiency

d. Lower bounds

f. Optimality

(6) code the algorithm

**2. Prove does not belong to .**

Assuming belong to , so any constant can make . But when , . So, the assumption is false.

**3. Find the upper bound of the number of nodes in a binary tree with a height of h.**

is the upper bound of the number of nodes in a binary tree.

**4. Explain data structures stack and queue; explain FIFO rule and FILO rule.**

**Stack and FILO:** A stack is a data structure which holds multiple elements of a single type and can be implemented with an array and an integer counter to indicate the current number of elements in the stack. Elements can be removed from a stack only in the reverse order to that in which they were inserted.

**Queue and FIFO:** A queue is a data structure which holds multiple elements of a single type and can be implemented with an array and two integer counters to indicate the current start and next insertion positions. Elements can be removed from a queue only in the order in which they were inserted.

**5. Explain data structure heap.**

A heap is a binary tree with an additional property and the value in any node is less than or equal to the value in its parent node. (except for the root node).

**6. List all main data structures and explain their definitions and operations.**

**(1) Array:** An array is a data structure consisting of a fixed number of data items of the same type

**(2) List:** A sequence of zero or more nodes each containing two kinds of information, some data and one or more links called pointers to other nodes of the linked list.

**(3) Records:** A record is a data structure consisting of a fixed number of items, unlike an array, the elements in a record may be of differing types and are named.

**(4) Graphs:** A graph G = <V, E> is defined by a pair of two sets, a finite set V of items called vertices and a set E of vertex pairs called edges.

**(5) Tree and Forest:** A tree (or free tree) is a connected acyclic graph. Forest is a graph that has no cycles but is not necessarily connected.

**7. List all main algorithm design paradigms and strategies and their corresponding problems.**

Brute force, Divide and conquer, Decrease and conquer, Transform and conquer, Space and time tradeoffs, Greedy approach, Dynamic programming, Backtracking, Branch and bound

**8. When we use a max heap to implement a priority queue, what are the time complexity of both the add and delete operations?**

The priority queue need to be sorted after add and delete operations, so the worst case of time complexity is of both the add and delete operations.

**9. Find the explicit expression of T (n), if T (n) = T (n −1)+n, T (1)=1.**

Summation(累加法):

**10. In a circular doubly linked list with 10 nodes, we will need to change 4 links if we want to delete a node other than the head node. Point out which four links.**

Suppose the node is b, the prior code of b is a, the next code of b is c. So the four links are a->next, b->prior, b->next, c->prior.

**11. Prove if and .**

if  and

So

**12. When using linked list to perform insertion sort, each time we remove an element from the input list and insert it to the correct position in the linked list. Assume that we have n numbers to be sorted, prove that the time complexity of the insertion sort algorithm is**

In the worst case of insertion sort, the first element doesn’t move, the second element needs one compare to find proper position …… No. n element needs n-1 compares to find proper position, so the total time is , .

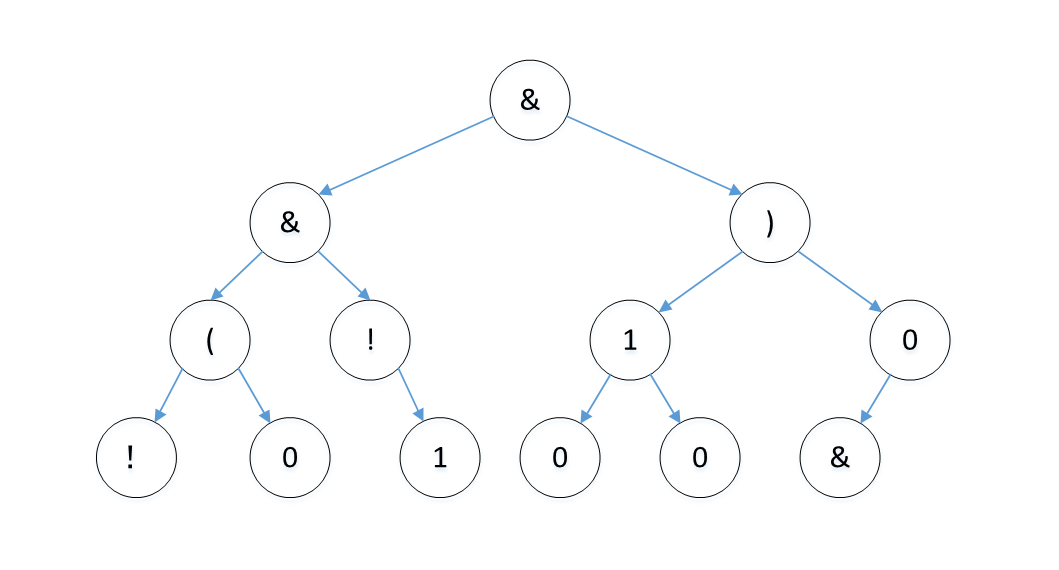
**13.** Denote the total number of moving as

**14. 1)**

**2)**

**3)**

**4)** Heapsort, shiftdown or shiftup.

**15.** 

**prefix expression:** &&(!0!1)1000&

**16~20.** CCACD