1. **数组和链表的区别:**

答：数组不允许 动态定义数组大小，使用前必须定义其大小；链表采用动态分配内存的方式，需要时分配内存空间，不需要时释放空间，不会造成内存的浪费。

从逻辑结构上来看：数组的大小一旦定义就不能改变，不能适应数据动态增减的情况；链表进行动态存储分配，可以适应数据动态的增减情况，可以方便插入删除数据

从存储角度来看：数组从栈中分配空间，对程序员方便快速，单自由度小；链表是从堆中分配空间，自由度大，但申请管理比较麻烦。

访问：数组在内存中是连续的，可用下标索引进行访问，链表是链式存储结构，访问元素时只能通过线性顺序由前向后顺序访问

1. **栈和队列的区别：**

答：

a，规则：栈 后进先出，队列： 先进先出（fifo）；

b，插入删除操作限定不同：栈：只能在一端插入删除；队列：在一端插入，在另一端删除。

c，遍历数据的速度不同：栈：只能从栈顶取数据，最先进入栈底的需要遍历整个栈才能取出来，而且在遍历数据的同时，需要为数据开辟临时空间，保持数据在遍历前的一致性。队列：基于地址指针进行遍历，而且可以从头部或者尾部进行遍历，但不能同时遍历，遍历速度快。

1. **单链表的反转：**

用三个结点保存链表的结构，然后将该前两个结点反转，三个结点以此向链表的下一个位置移动一个结点。

1. Node \* ReverseList(Node \*head)
2. {
3. Node \*p1,\*p2,\*p3;
4. **if**(head==NULL||\*head==NULL)
5. **return** head;
6. p1=head;
7. p2=p1->next;
8. **while**(p2)             //注意条件
9. {
10. p3=p2->next;       //要改变p2->next的指针，所以必须先保留p2->next
11. p2->next=p1;
12. p1=p2;            //循环往后
13. p2=p3;
14. }
15. head->next=NULL;   //原先的head已经变成tail，别忘了置空，只有到这步才能置空
16. \*head=p1;
17. **return** head;
18. }
19. **二元查找树：**

二元查找树： 它首先要是一棵[二元树](http://baike.baidu.com/view/19209.htm)，在这基础上它或者是一棵空树；或者是具有下列性质的二元树： （1）若左子树不空，则左子树上所有结点的值均小于它的父结点的值； （2）若右子树不空，则右子树上所有结点的值均大于等于它的[根结点](http://baike.baidu.com/subview/549553/549553.htm)的值； （3）左、右子树也分别为二元查找树

1. **二元树的遍历：**

