语言知识点记录

Java数组切片 java.util.Arrays.copyOf(arr,k);可以当做数组切片

Split方法里面所有保留字需要加转义符号”\\”

剑指offer 22， 链表倒数第k个节点

用快慢指针：

不需要遍历两遍就能找到。

链表删除节点，如果给了head，就查到这个节点 把上个节点next改到下一个。 如果没给head，就依次往后覆盖，并把倒数第二个的next 改成null。

归并排序：

class Solution {

    public int[] smallestK(int[] arr, int k) {

        if(k==0){return new int[0];}

        if(k==arr.length){return arr;}

        // if(k<=arr.length/2){

        //     retrun find\_min(arr,k);

        // }

        // else{

        //     return find\_max(arr,arr.length-k);

        // }

        sort(arr,0,arr.length-1);

        return java.util.Arrays.copyOf(arr,k);

    }

    public static void sort(int[] arr,int start, int end){

        int mid=(start+end)/2;

        if(start<end){

            sort(arr,start,mid);

            sort(arr,mid+1,end);

            merge(arr,start,end,mid);

        }

    }

    public static void merge(int[] arr,int start, int end, int mid){

        int [] tmp=new int[end-start+1];

        int first\_idx=start;

        int sec\_idx=mid+1;

        int count=0;

        while(first\_idx <=mid && sec\_idx<=end){

            if(arr[first\_idx]<arr[sec\_idx]){

                tmp[count]=arr[first\_idx];

                first\_idx+=1;

                count+=1;

            }

            else{

                tmp[count]=arr[sec\_idx];

                sec\_idx+=1;

                count+=1;

            }

        }

        while(first\_idx<=mid){

            tmp[count]=arr[first\_idx];

            count+=1;

            first\_idx+=1;

        }

        while(sec\_idx<=end){

            tmp[count]=arr[sec\_idx];

            count+=1;

            sec\_idx+=1;

        }

        int idx=0;

        for(int i=start;i<=end;i++){

            arr[i]=tmp[idx];

            idx+=1;

        }

    }

环形链表判断也可以通过快慢指针！ 如果快慢指针能相遇则说明有环！