**题目描述: 输入一个递增排序的数组和一个数字S，在数组中查找两个数，使得他们的和正好是S，如果有多对数字的和等于S，输出两个数的乘积最小的。**

**代码(mine):**

class Solution {

public:

    vector<int> FindNumbersWithSum(vector<int> array, int sum) {

        if (array.size() <= 1)return num;

        if ((double)sum / 2 <= (double)array[0] || (double)sum / 2 >= (double)array[array.size()-1])return num;

        int mid = 0, high = array.size(), min, max, product = array[array.size() - 1] \* array[array.size() - 1];

        bool flag\_mid = false, flag\_high = false,flag=0;

        for (int i = 0; i < array.size(); i++){

            if (array[i]>sum/2&&flag\_mid==false ){

                mid = i;

                flag\_mid = true;

            }

            if (array[i] >sum&&flag\_high==false ){

                high = i;

                flag\_high = true;

            }

        }

        for (int i=0; i <mid; i++){

            for (int j = mid; j < high;j++){

                if (array[i] + array[j] == sum&&array[i]\*array[j]<product){

                    product = array[i] \* array[j];

                    min = array[i];

                    max = array[j];

                    flag = 1;

                }

            }

        }

        if (flag){

            num.push\_back(min);

            num.push\_back(max);

        }

        return num;

    }

    vector<int>num;

};

思路:寻找到第一个大于sum/2结点mid和第一个大于sum的结点high，然后在[0,mid-1]和[mid,high-1]两段区间里分别寻找.

**A better solution:**

(1).让两个指针分别指向数组的头部和尾部，相加，如果小于sum，则增大头指针，若大于sum，减小尾指针.

(2).退出条件，相等或头部等于尾部.