基本设计思想：贪心，先使用排序算法将数组从大到小排序，然后依次放进A1和A2当中

，当集合元素总个数为偶数时，则取两个子集元素个数相等；当集合元素总个数为奇数时，令A1个数为[n/2]+1，n/2向下取整，此时A1比A2多一个，满足|n1-n2|最小且|S1-S2|最大，

算法：

1. int n;//初始长度

2. int a[MAX\_SIZE];//初始数组

3.

4. void quick\_sort(int l, int r) {

5.

6.     if (l >= r) return;

7.

8.     int i = l, j = r, x = a[l];

9.     while (i < j)

10.     {

11.         while (a[j] <= x && i < j)//左指针右移

12.             j--;

13.         while (a[i] >= x && i < j)//右指针左移

14.             i++;

15.

16.         if (i < j) //交换使左边的元素都大于标准值，右边的元素小于标准值。

17.             swap(a[i], a[j]);

18.     }

19.     a[l] = a[i];//将标准值放于正确的位置

20.     a[i] = x;

21.     quick\_sort(l, i - 1);//继续递归左右两端数组

22.     quick\_sort(i + 1, r);

23. }

24.

25. void sel() {

26.     quick\_sort(1 , n);//使用从大到小排序整个数组。

27.

28.     int S1[MAX\_SIZE] , S2[MAX\_SIZE];

29.     int n1 = 0 , n2 = 0;

30.

31.     for (int i = 1 , j = n ; i <= j ; i++ , j--) {

32.         if (i != j) {//保证S1和S2要么个数相等，要么S1多一个

33.             S1[++n1] = a[i];

34.             S2[++n2] = a[j];

35.         }

36.         else {

37.             S1[++n1] = a[i];

38.         }

39.     }

40.     for (int i = 1 ; i <= n1 ; i++)

41.         cout<<S1[i]<<" ";

42.     for (int i = 1 ; i <= n2 ; i++)

43.         cout<<S2[i]<<" ";

44.     //此时S1，S2即为所求两个子集。

45. }

46.

平均时间复杂度: 排序算法时间复杂度为O(nlogn) ，划分集合时间复杂度为O(n) ，故最后总的平均时间复杂度为O(nlogn)。

空间复杂度：快排的递归深度为O(logn) , 额外使用的两个数组空间复杂度为O(n) ，最后总的空间复杂度为 O(n)。