设计思路基于贪心算法，首先使用排序算法将数组按照从大到小的顺序排列。然后，依次将元素放入两个集合 A1 和 A2 中。在集合元素总数为偶数时，两个子集的元素个数相等；而在集合元素总数为奇数时，我们设定 A1 的元素个数为 [n/2] + 1，其中 n/2 向下取整。这样设计的目的是使得 A1 比 A2 多一个元素，以确保在满足 |n1 - n2| 最小的同时，|S1 - S2| 最大。

算法：

1. #include <iostream>

2. #include <vector>

3.

4. using namespace std;

5.

6. // 快速排序的划分函数

7. int partition(vector<int>& arr, int low, int high) {

8. int pivot = arr[high];

9. int i = low - 1;

10.

11. for (int j = low; j < high; ++j) {

12. if (arr[j] > pivot) {

13. i++;

14. swap(arr[i], arr[j]);

15. }

16. }

17.

18. swap(arr[i + 1], arr[high]);

19. return i + 1;

20. }

21.

22. // 快速排序

23. void quickSort(vector<int>& arr, int low, int high) {

24. if (low < high) {

25. int pivotIndex = partition(arr, low, high);

26.

27. quickSort(arr, low, pivotIndex - 1);

28. quickSort(arr, pivotIndex + 1, high);

29. }

30. }

31.

32. // 划分集合的函数

33. void Parti (vector<int>& arr, int& n1, int& n2, int& S1, int& S2) {

34. quickSort(arr, 0, arr.size() - 1);

35.

36. n1 = n2 = S1 = S2 = 0;

37.

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

39. if (i % 2 == 0) {

40. n1++;

41. S1 += arr[i];

42. } else {

43. n2++;

44. S2 += arr[i];

45. }

46. }

47. }

48.

49. int main() {

50. vector<int> A;

51.

52. int n1, n2, S1, S2;

53. Parti (A, n1, n2, S1, S2);

54.

55. cout << "划分后的集合 A1，n1，S1: ";

56. for (int i = 0; i < n1; ++i) {

57. cout << A[i] << " ";

58. }

59. cout << ", " << n1 << ", " << S1 << endl;

60.

61. cout << "划分后的集合 A2，n2，S2: ";

62. for (int i = n1; i < A.size(); ++i) {

63. cout << A[i] << " ";

64. }

65. cout << ", " << n2 << ", " << S2 << endl;

66.

67. return 0;

68. }

69.

平均时间复杂度方面，排序算法的时间复杂度为 O(n log n)，而集合划分的时间复杂度为 O(n)。因此，总的平均时间复杂度为 O(n log n)。

在空间复杂度方面，快速排序的递归深度为 O(log n)，而额外使用的两个数组的空间复杂度为 O(n)。综合考虑，总的空间复杂度为 O(n)。