题目描述

部门组织绿岛骑行团建活动。租用公共双人自行车，每辆自行车最多坐两人，最大载重M。
给出部门每个人的体重，请问最多需要租用多少双人自行车。

输入描述

第一行两个数字m、n，分别代表自行车限重，部门总人数。

第二行，n个数字，代表每个人的体重，体重都小于等于自行车限重m。

- 0<m<=200
- 0<n<=1000000

输出描述

最小需要的双人自行车数量。

用例

输入	3 4 3 2 2 1
输出	3
说明	无

题目解析

本题需要最少的车辆，即尽可能组合出重量小于等于m的两人组。

首先，我们可以将所有人按体重升序，然后将最大体重和m比较，若最大体重大于等于m，则这个人只能一人占一辆车，车数量count++，然后将最大体重弹出，继续将剩下体重中最大的和m比较，逻辑同上，直到最大体重小于m时，停止弹出。

在剩余体重中，我们利用 双指针，i指针指向最小体重，j指针指向最大体重，然后组合它们，即arr[i]+arr[j]，和m比较，若小于等于m，则说明这两个人可以共享一辆车，车数量count++，然后i++，j--。如果arr[i]+arr[j]>m，则说明两个人无法共享一辆车，我们只能优先将这里车分配给较大体重的人，此时车数量count++，然后j--。

按上面逻辑移动双指针，最后可能会出现两种情况：

- i > j　　此情况下所有人均分配到了车，因此可以直接输出count作为题解
- i === j　此情况下还有一个人未分配到车，因此需要count++，为这个人单独分配一辆车

JavaScript算法源码

```
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

const lines = [];
rl.on("line", (line) => {
  lines.push(line);

  if (lines.length === 2) {
    const [m, n] = lines[0].split(" ").map(Number);
    const arr = lines[1].split(" ").map(Number);

    console.log(getResult(arr, m, n));

    lines.length = 0;
  }
});

function getResult(arr, m, n) {
  arr.sort((a, b) => a - b);

```

JavaScript算法源码

```javascript
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

const lines = [];
rl.on("line", (line) => {
  lines.push(line);

  if (lines.length === 2) {
    const [m, n] = lines[0].split(" ").map(Number);
    const arr = lines[1].split(" ").map(Number);

    console.log(getResult(arr, m, n));

    lines.length = 0;
  }
});

function getResult(arr, m, n) {
  arr.sort((a, b) => a - b);

  let count = 0;

  // while (arr.at(-1) >= m) {
  //   count++;
  //   arr.pop();
  // }

  let i = 0;
  let j = arr.length - 1;

  while (i < j) {
    if (arr[i] + arr[j] <= m) i++;
    j--;
    count++;
  }

  if (i === j) count++;

  return count;
}
```

Java算法源码

```java
import java.util.Arrays;
import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    int m = sc.nextInt();
    int n = sc.nextInt();

    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
      arr[i] = sc.nextInt();
    }

    System.out.println(getResult(arr, m));
  }

  public static int getResult(int[] arr, int m) {
    Arrays.sort(arr);

    int count = 0;

    int i = 0;
    int j = arr.length - 1;

    while (i < j) {
      if (arr[i] + arr[j] <= m) i++;
      j--;
      count++;
    }

    if (i == j) count++;

    return count;
  }
}
```

Python算法源码

Python算法源码

```python
# 输入获取
m, n = map(int, input().split())
arr = list(map(int, input().split()))


# 算法入口
def getResult(arr, m, n):
    arr.sort()

    count = 0

    i = 0
    j = n - 1

    while i < j:
        if arr[i] + arr[j] <= m:
            i += 1
        j -= 1
        count += 1

    if i == j:
        count += 1

    return count


# 算法调用
print(getResult(arr, m, n))
```