

JS-子集合-某组织举行会议

题目描述：

某组织举行会议，来了多个代表团同时到达，接待处只有一辆汽车，可以同时接待多个代表团，为了提高车辆利用率，请帮接待员计算可以坐满车的接待方案，输出方案数量。

约束：

- 1、一个团只能上一辆车，并且代表团人数（代表团数量小于 30，每个代表团人数小于 30）小于汽车容量（汽车容量小于 100）
- 2、需要将车辆坐满

输入描述：

第一行 代表团人数，英文逗号隔开，代表团数量小于 30，每个代表团人数小于 30

第二行 汽车载客量，汽车容量小于 100

输出描述：

坐满汽车的方案数量

如果无解输出 0

补充说明：

各代表团人数 5,4,2,3,2,4,9

汽车载客量 10

输出 4

解释 以下几种方式都可以坐满车，所以，优先接待输出为 4

[2, 3, 5]

[2, 4, 4]

[2, 3, 5]

示例 1

输入：

5,4,2,3,2,4,9

10

输出：

4

说明：

解释 以下几种方式都可以坐满车，所以，优先接待输出为 4

[2, 3, 5]

[2, 4, 4]

[2, 3, 5]

[2, 4, 4] [2, 4, 4]

```
const rl = require("readline").createInterface({ input: process.stdin });
```

```
var iter = rl[Symbol.asyncIterator]();
```

```
const readline = async () => (await iter.next()).value;
```

```
void async function () {
```

```
    // Write your code here
```

```
    let nums = (await readline()).split(",").map(num => +num);
```

```
    let count = Number(await readline());
```

```
    let res = [];
```

```
nums.sort((a, b) => a - b);
const backTrack = (index = 0, arr = []) => {
  let sum = arr.reduce((prev, curr) => prev + curr, 0);
  if(sum >= count) {
    sum == count && res.push([...arr]);
    return;
  }

  for(let i = index; i < nums.length; i++) {
    arr.push(nums[i]);
    backTrack(i+1, arr);
    arr.pop();
  }
}

backTrack();

console.log(res.length);
}()
```