题目描述

任务编排服务负责对任务进行组合调度。

参与编排的任务有两种类型，其中一种执行时长为taskA，另一种执行时长为taskB。

任务一旦开始执行不能被打断，且任务可连续执行。

服务每次可以编排num个任务。

请编写一个方法，生成每次编排后的任务所有可能的总执行时长。

输入描述

第1行输入分别为第1种任务执行时长taskA，

第2种任务执行时长taskB，

这次要编排的任务个数num，以逗号分隔。

注：每种任务的数量都大于本次可以编排的任务数量

- 0 < taskA
- 0 < taskB
- 0 <= num <= 100000

输出描述

数组形式返回所有总执行时时长，需要按从小到大排列。

用例

输入	1,2,3
输出	[3, 4, 5, 6]
说明	无

题目解析

本题看上去是求解 全排列，但是实际上无论如何排列，其实就只是两种类型任务的排列，而且本题要求任务编排的执行总时长，这其实就是就每种排列的和，因此其实我们不需要求解排列，只需要求解该排列对应的组合即可

比如用例中，有三个任务，那么有如下组合：

- 三个1
- 两个1，一个2
- 一个1，两个2
- 三个2

无论每个组合，能编排成几个排列，其实执行总时长，即排列的和都是一样的

- 三个1：和为3
- 两个1，一个2：和为4
- 一个1，两个2：和为5
- 三个2：和为6

因此，本题其实就是求解两种类型任务各自可能的数量。

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,
});
```

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,
});

rl.on("line", (line) => {
  const [taskA, taskB, num] = line.split(",").map(Number);
  console.log(getResult(taskA, taskB, num));
});

function getResult(taskA, taskB, num) {
  if (taskA === taskB) {
    return [taskA * num];
  }

  const ans = new Set();
  for (let i = 0; i <= num; i++) {
    ans.add(taskA * i + taskB * (num - i));
  }

  return [...ans].sort((a, b) => a - b);
}
```

Java算法源码

```java
import java.util.*;

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

        String str = sc.next();
        Integer[] arr = Arrays.stream(str.split(",")).map(Integer::parseInt).toArray(Integer[]::new);

        System.out.println(getResult(arr[0], arr[1], arr[2]));
    }

    public static String getResult(int taskA, int taskB, int num) {
        if(taskA == taskB) {
            return Arrays.toString(new int[]{taskA * num});
        }

        TreeSet<Integer> ans = new TreeSet<>();
        for(int i=0; i<=num; i++) {
            ans.add(taskA * i + taskB * (num - i));
        }

        return ans.toString();
    }
}
```

Python算法源码

```python
# 输入获取
taskA, taskB, num = map(int, input().split(","))


# 算法入口
def getResult(taskA, taskB, num):
    if taskA == taskB:
        return [taskA * num]

    ans = set()
    for i in range(num + 1):
        ans.add(taskA * i + taskB * (num - i))

    sorted(ans)

    return list(ans)


# 算法调用
print(getResult(taskA, taskB, num))
```