运维工程师🔍采集到某产品线网运行一天产生的日志n条，现需根据日志时间先后顺序对日志进行排序，日志时间格式为H:M:S:N。

- H表示小时(0~23)
- M表示分钟(0~59)
- S表示秒(0~59)
- N表示毫秒(0~999)

时间可能并没有补全，也就是说，01:01:01:001也可能表示为1:1:1:1。

输入描述

第一行输入一个整数n表示日志条数，1<=n<=100000，接下来n行输入n个时间。

输出描述

按时间升序排序之后的时间，如果有两个时间表示的时间相同，则保持输入顺序。

用例

输入	2 01:41:8:9 1:1:09:211
输出	1:1:09:211 01:41:8:9
说明	无

输入	3 23:41:08:023 1:1:09:211 08:01:22:0
输出	1:1:09:211 08:01:22:0 23:41:08:023
说明	无

输入	2 22:41:08:023 22:41:08:23
输出	22:41:08:023 22:41:08:23
说明	两个时间表示的时间相同，保持输入顺序

题目解析

排序日志时间用到的知识是：

- 将字符串时间中各要素提取出来（正则捕获组），然后计算出总毫秒数
- 根据Array.prototype.sort自定义排序策略，来算出的总毫秒数进行排序

Java算法源码

```java
import java.util.Arrays;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

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

    int n = Integer.parseInt(sc.nextLine());

    String[] logs = new String[n];
    for (int i = 0; i < n; i++) logs[i] = sc.nextLine();

    getResult(logs);
  }

  public static void getResult(String[] logs) {
    Arrays.sort(logs, (a, b) -> (int) (convet(a) - convet(b)));

    for (String log : logs) {
      System.out.println(log);
    }
  }

  public static long convet(String log) {
    String reg = "(\\d+):(\\d+):(\\d+).(\\d+)";

    Matcher matcher = Pattern.compile(reg).matcher(log);

    if (matcher.find()) {
      long H = Long.parseLong(matcher.group(1)) * 60 * 60 * 1000;
      long M = Long.parseLong(matcher.group(2)) * 60 * 1000;
      long S = Long.parseLong(matcher.group(3)) * 1000;
      long N = Long.parseLong(matcher.group(4));
      return H + M + S + N;
    } else {
      return 0;
    }
  }
}
```

JS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 = [];
let n;
rl.on("line", (line) => {
  lines.push(line);
  if (lines.length == 1) {
    n = lines[0] - 0;
  }

  if (n && lines.length == n + 1) {
    lines.shift();
    console.log(getResult(lines));
    lines.length = 0;
  }
});

/* 日志排序 */
function getResult(logs) {
  logs.sort((log1, log2) => {
    const a = convert(log1);
    const b = convert(log2);
    return a - b;
  });
  return logs.join("\n");
}

/* 利用正则捕获组，将时间每部分提取出来 */
function convert(log) {
  const [_, H, M, S, N] = /(\d+):(\d+):(\d+).(\d+)/.exec(log);
  return (
    parseInt(H) * 60 * 60 * 1000 +
    parseInt(M) * 60 * 1000 +
    parseInt(S) * 1000 +
    parseInt(N)
  );
}
```

Python算法源码

```python
# 输入获取
import re

n = int(input())
logs = [input() for _ in range(n)]


def convert(log):
    reg = re.compile(r"(\d+):(\d+):(\d+).(\d+)")

    matcher = reg.match(log)

    H = int(matcher.group(1)) * 60 * 60 * 1000
    M = int(matcher.group(2)) * 60 * 1000
    S = int(matcher.group(3)) * 1000
    N = int(matcher.group(4))

    return H + M + S + N


# 算法入口
def getResult():
    logs.sort(key=lambda x: convert(x))
    return "\n".join(logs)


# 算法调用
print(getResult())
```