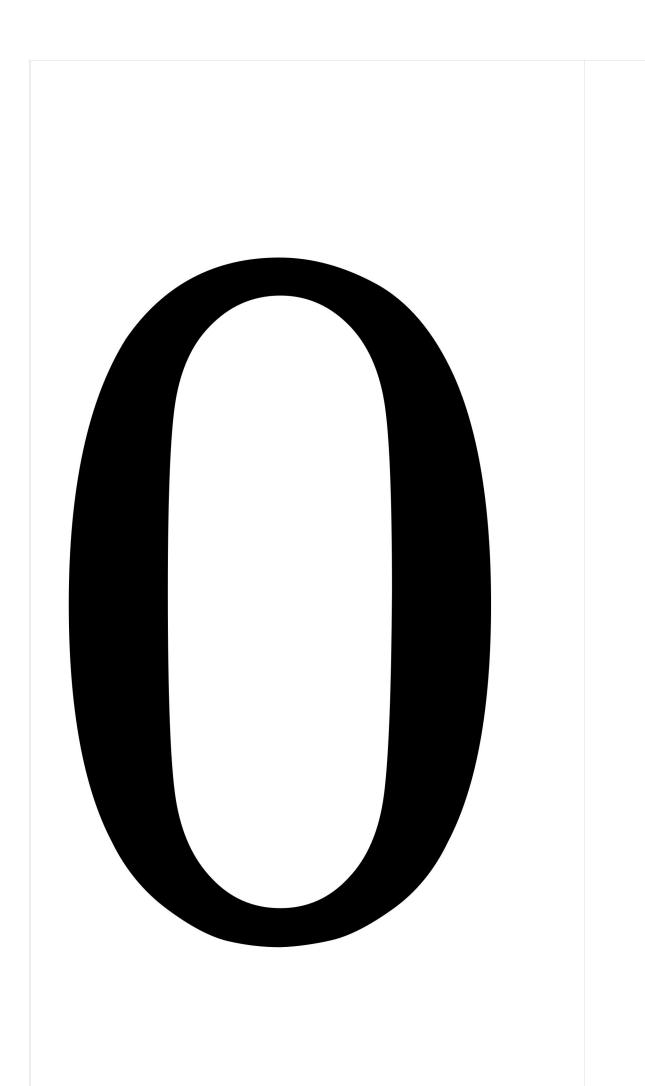
题目描述:

对报文进行重传和重排序是常用的可靠性机制,重传缓冲区内有一定数量的子报文,每个子报文在原始报文中的顺序已知,现在需要恢复出原始报文。

输入描述:

输入第一行为N,表示子报文的个数,



输入第二行为 N 个子报文,以空格分开,子报文格式为字符串报文内容+后缀顺序索引,字 符串报文内容由[a-z,A-Z]组成,后缀为整形值,表示顺序。顺序值唯一,不重复。 输出描述: 输出恢复出的原始报文。按照每个子报文的顺序的升序排序恢复出原始报文,顺序后缀需要 从恢复出的报文中删除掉。 示例 1 输入: rolling3 stone4 like1 a2 输出: like a rolling stone 说明: 4 个子报文的内容分别为 'rolling', 'stone', 'like', 'a', 顺序值分别为 3, 4, 1, 2, 按照 顺序值升序并删除掉顺序后缀,得到恢复的原始报文: like a rolling stone 示例 2 输入: gifts6 and7 Exchanging1 all2 precious5 things8 kinds3 of4 输出: Exchanging all kinds of precious gifts and things process.stdin.resume(); process.stdin.setEncoding("utf-8"); let input = ""; process.stdin.on("data", (data) => { input += data; return;

```
});
function sp(str) {
     var x = "";
     varid = 0;
     for (var i = 0; i < str.length; i++) {
          if ("0" <= str[i] && str[i] <= "9") {
                id = i;
                break;
          }
          x += str[i];
     }
     return [x, Number(str.slice(id))];
}
process.stdin.on("end", () => {
     const lines = input.trim().split("\n");
     // write your code here
     var n = Number(lines[0]);
     var a = lines[1].split(" ").map(sp);
     a.sort((x, y) => x[1] - y[1]);
     var res = "";
     for (var i = 0; i < a.length; i++) {
          res += a[i][0];
          if (i != a.length - 1) res += " ";
     }
     console.log(res);
});
```