[J] Firewall Ruleset Minimization

Time limit: 1 second
Memory limit: 65535 kBytes

Description

You are responsible for maintaining the firewall of a high-security data center. The firewall operates based on a ruleset - a prioritized list of rules that decide whether to allow or deny traffic based on IP address ranges and port ranges.

Each rule specifies an action (`ALLOW` or `DENY`), a range of IPv4 addresses, and a range of network ports. The firewall evaluates incoming traffic against the rules from top to bottom, applying **the first matching rule**.

Over time, the ruleset has grown large and contains redundant or overlapping rules. Your task is to remove **redundant rules** without changing the firewall's behavior.

We say that a rule R_j is redundant if there exists an *earlier* single rule R_i (with $i < j$) such that every (IP, port) pair matched by R_j is also matched by R_i.Removing R_j does not change which rule the firewall selects for any (IP, port) pair in this case.

Note: if a rule is covered only by the union of several earlier rules (no single earlier rule fully covers it), it is **not** considered redundant and must not be removed.

Futher remarks

- The firewall evaluates rules from top to bottom. The first rule that matches a given (IP, port) pair determines the action.
- IP ranges are inclusive.
- IPv4 clarification: an address of the form `A.B.C.D` represents a 32-bit unsigned integer with `A` as the most-significant 8 bits and `D` as the least-significant 8 bits. Thus, ranges like `A.B.C.D` to `E.F.G.H` should be interpreted as the closed interval of the corresponding 32-bit integers.

Input

The first line contains a single integer N, the number of firewall rules.

Each of the next N lines describes one rule: `action ip_start ip_end port_start port_end`. The action can either be `ALLOW` or `DENY` (in uppercase). IPv4 addresses are in dot-decimal notation, defining a closed inclusive range of IPs. Ports are integers defining an inclusive port range.

Output

The output consists of the list of rules after removing redundant rules. Each rule is formatted as in the input.

Constraints

- $1 \leq N \leq 10\,000$
- IP addresses are valid IPv4 in dot-decimal notation: `A.B.C.D`, $0 \leq$ `A,B,C,D` ≤ 255.
- Port ranges are inclusive and within 0 to 65535.

Example

Input 1

```
4
ALLOW 192.168.1.0 192.168.1.255 20 25
ALLOW 192.168.1.0 192.168.1.255 26 30
DENY 10.0.0.0 10.0.0.255 0 1023
DENY 10.0.0.0 10.0.0.255 1024 65535
```

Output 1

```
ALLOW 192.168.1.0 192.168.1.255 20 25
ALLOW 192.168.1.0 192.168.1.255 26 30
DENY 10.0.0.0 10.0.0.255 0 1023
DENY 10.0.0.0 10.0.0.255 1024 65535
```

There are no rules that are shadowed by other rules in this example, so all rules are kept.

Input 2

```
3
ALLOW 10.0.0.0 10.0.0.255 80 90
DENY 10.0.0.0 10.0.0.255 80 80
ALLOW 10.0.0.0 10.0.0.255 443 443
```

Output 2

```
ALLOW 10.0.0.0 10.0.0.255 80 80
ALLOW 10.0.0.0 10.0.0.255 443 443
```

The `DENY` rule is removed because the preceding `ALLOW` rule already handles all traffic it would match. Firewalls apply the first matching rule, so the `DENY` is unreachable.

Input 3

```
2
ALLOW 0.0.10.0 0.0.12.0 1000 2000
ALLOW 0.0.11.0 0.0.11.255 1500 1500
```

Output 3

```
ALLOW 0.0.10.0 0.0.12.0 1000 2000
```

The second rule is redundant because every (IP, port) pair it matches is already matched by the earlier `ALLOW` rule.