Zadanie X
Kolejka PRL-owska

Nieodłączną częścią codzienności okresu PRL-u była kolejka. W kolejce czekano na wszystko: na mięso, cukier, sprzęt AGD, meble, samochód, podłączenie telefonu, papier toaletowy. Czas oczekiwania w kolejce trwał od kilku godzin do kilku lat...

W przypadku gdy przewidywany czas oczekiwania w kolejce był długi sporządzano tzw. listę kolejkową, a nad utrzymaniem porządku kolejki czuwał komitet kolejkowy. Każda osoba z listy kolejkowej miała obowiązek stawić się o określonej godzinie w kolejce i podczas odczytywania listy zgłosić swoją obecność. W przypadku kolejki kilkudniowej było to na ogół 3-4 razy na dobę. Niestawienie się skutkowało skreśleniem z listy. Kolejka PRL-owska nie była prawdziwą, sprawiedliwą kolejką. Zdarzało się, że ktoś uprzywilejowany dopisywał się do kolejki na początek, czasem ktoś rezygnował, nierzadko ktoś wpychał się w środek...

Napisz program, który będzie symulował działanie listy kolejkowej. Operacje wykonywane na liście:

- `NEW name age` — nowa osoba o imieniu `name` oraz wieku `age` chce się dopisać do listy kolejkowej; dodawana jest na koniec.
- `VIP name age` — do kolejki chce dołączyć osoba o imieniu `name` oraz wieku `age` w bardzo podeszłym wieku albo w ciąży. Zgodnie z zasadami kolejek PRL-owskich taka osoba obsługiwana jest poza kolejnością. Należy dodać ją na początek listy kolejkowej.
- `INSERT name age number` — Pewna osoba trzymała miejsce w kolejce dla kolegi bratanka koleżanki z pracy. Kolega ten, o imieniu `name` wciska się do kolejki na pozycje `number`. Pozycje w liście kolejkowej liczone są od 1. Jeśli `number` jest podany niepoprawnie, należy wypisać `ERROR`.
- `BUYING` — osoba z początku listy dokonuje wyczekanego zakupu; jest usuwana z listy. Jeśli lista jest pusta należy wypisać `ERROR`.
- `RESIGNATION` — Osoba z końca listy traci nadzieje na jakikolwiek zakup, rezygnuje z oczekiwania. Jeśli lista jest pusta należy wypisać `ERROR`.
- `DELETE name age` — Osoba o imieniu `name` nie zgłosiła swojej obecności. Zostaje skreślona z listy. Jeśli osoby `name` nie ma na liście należy wypisać `ERROR`. Jeśli na liście jest więcej osób o imieniu `name` i wieku `age` należy usunąć osobę występującą najwcześniej.
- `PRINT` — wypisanie wszystkich osób z listy kolejkowej. Jeśli lista jest pusta należy wypisać `EMPTY`.
- `REVERSE` — Hm.. Przyjechała ciężarówka z towarem. Produkty zostały wypakowane do sklepu obok, akurat tam gdzie stała ostatnia osoba z kolejki... Ostatni będą

pierwszymi.. Osoby dokonują zakupów w odwrotnej kolejności. Jeśli lista jest pusta należy wypisać `EMPTY`.

- `CLEAN` — Brakło produktów. Nie ma już na co czekać. Lista przestaje istnieć.

Zadanie należy rozwiązać poprzez zaimplementowanie klasy `list`, realizującej **wskaźnikową listę jednokierunkową**. Elementami listy są obiekty klasy `person` zawierającej prywatne pola: `name`, `age` oraz `next` wskazujące kolejny element listy. Zalecana jest implementacja listy z głowa. Klasa `list` powinna być zaprzyjaźniona z klasą `person`, by mieć dostęp do jej pól prywatnych.

Wejście

Pierwsza linia wejścia zawiera liczbę całkowitą z ($1 \leqslant z \leqslant 2 \cdot 10^9$) – liczbę zestawów danych, których opisy występują kolejno po sobie. Opis jednego zestawu jest następujący:

Pierwsza linia zawiera liczbę naturalną n ($1 \leqslant n \leqslant 10^4$) będącą liczbą operacji wykonywanych na liście. W kolejnych n liniach znajdują się operacje. Wszystkie nazwy występujące w operacjach składają się z dużych i małych liter alfabetu angielskiego i nie przekraczają 8 znaków.

Wyjście

Dla każdego zestawu danych wykonaj kolejno operacje na liście.

Dostępna pamięć: 2MB
Wymagany język: C++

Przykład

Dla danych wejściowych:

```
1
15
INSERT Karol 40 1
VIP Gertruda 25
VIP Pawel 50
NEW Pawel 30
INSERT Rafal 45 10
PRINT
REVERSE
DELETE Karol 40
INSERT Jozek 33 1
DELETE Gertruda 25
DELETE Kasia 25
RESIGNATION
PRINT
CLEAN
PRINT
```

Poprawną odpowiedzią jest:

```
ERROR
Pawel 50
Gertruda 25
Karol 40
Pawel 30
ERROR
Jozek 33
Pawel 30
EMPTY
```