Dwa małe roboty poruszają się po słowie złożonym ze znaków A i B. Zadanie polega na stwierdzeniu, czy słowo jest dobre czy złe.

Dwa roboty uznają słowo za dobre, jeśli liczba liter A i liczba liter B są równe w trójkowym-centrum (środkowe słowo po podziale na trzy podsłowa równej długości). Wszystkie inne słowa są złe. Dla przykładu, słowo BBABAABBBABA jest dobre, ponieważ jego trójkowe-centrum AABB zawiera dwie litery A i dwie litery B. Dla innego przykładu, słowo BAABABBBBAAB jest złe, ponieważ, jego trójkowe-centrum ABBB zawiera jedną literkę A i trzy literki B. Słowo AABBAABB jest złe, ponieważ nie posiada trójkowego centrum (długość słowa nie jest podzielna przez trzy).

W każdym momencie robot okupuje jeden znak słowa i widzi pewne rzeczy w jego otoczeniu, które zostały opisane dalej. Robot może stać na skrajnie lewej pozycji, w środku słowa, lub na skrajnie prawej pozycji (innymi słowy, może dowiedzieć się, czy są literki na lewo lub prawo od niego). Może również dowiedzieć się, czy jest sam na pozycji, czy może inny robot również znajduje się na tej pozycji razem z nim. Oczywiście, może również dowiedzieć się, jaka jest literka na pozycji, na której się znajduje (A lub B).

Poza tym, każdy robot jest w stanie zapamiętać wspomnienia ze swojej podróż. Niestety jego pamięć ma tylko cztery bity.

W głowie każdego robota jest „lista instrukcji", która opisuje jego zachowanie. Te listy dla obu robotów mogą być różne.

Każda instrukcja składa się z trzech części:

Lewa strona (warunek)→(znak *specjalny*) Prawa strona (*akcja*)

Lewa strona zawiera:

- Znak wskazujący, czy robot jest na jednym z końców słowa: **L** jeśli robot jest na skrajnie lewej pozycji, **I** jeśli robot jest na środkowej pozycji lub **R** jeśli robot jest na skrajnie prawej pozycji.
- Znak wskazujacy, czy drugi robot jest sam na tej pozycji, czy też nie. **O** jeśli robot jest sam na tej pozycji lub **T** jeśli na pozycji są dwa roboty.
- Znak, na którym stoi robot, **A** jest robot stoi na literze A lub **B** jeśli robot stoi na literze B
- 4 znaki (0 lub 1) opisujące zawartość pamięci robota. Np. 0101 jeśli pamięć robota zawiera bity 0101.

Lewa strona jest warunkiem, jej wartością jest „true" (prawda) lub „false" (fałsz). Warunek jest koniunkcją sprawdzenia obecnej pozycji robota oraz tego co przechowuje w czterech bitach pamięci.

Prawa strona wygląda następująco:

- Znak akcji robota: **L** jeśli robot wykonuje ruch w lewo, **R** jeśli robot wykonuje ruch w prawo, **S** jeśli robot pozostaje w miejscu, **Y** jeśli robot raportuje, że słowo jest dobre lub N jeśli robot raportuje, że słowo jest złe.
- Poza przypadkiem, kiedy praca robota jest zakończona (akcja Y lub N) – cztery dodatkowe znaki (0 lub 1), oznaczające zawartość pamięci np. 0110.

Każdy znak w instrukcji poza znakami specjalnymi i znakami akcji może zostać zastąpiony przez ?.

Jeśli znak ? występuje po lewej stronie instrukcji (w warunku) to znaczy, że odpowiadający znak warunku nie jest sprawdzany.

Jeśli znak występuje w części opisującej pamięć (po znaku akcji) to znaczy, że odpowiadający bit pamięci robota pozostaje niezmieniony.

Dla przykładu, instrukacja:

LT???01->R??10

mówi, że jeśli robot zajmuje skrajnie lewą pozycję słowa, inny robot zajmuje tę samą pozycję oraz dwa ostatnie bity pamięci są ustawione odpowiednio na 01 wtedy (nie ważne, czy robot stoi na A czy B) wykonuje ruch w prawo oraz ustawia dwa ostatnie bity pamięci na 1 i 0 bez zmiany dwóch pierwszych bitów.

Oba roboty zaczynają na skrajnie lewej pozycji słowa. Początkowo, wszystkie bity w pamięci robota są ustawione na 0.

Każdej milisekundy, każdy robot rozgląda się, przeszukuje pamięć, listę instrukcji oraz decyduje, czy wykonać ruch w lewo, ruch w prawo, zostać w miejscu lub wyrazić swoją opinię o słowie. Dodatkowo, robot może zmienić coś w swojej pamięci. W momencie, jeśli robot wyrazi swoją opinię na temat słowa (uzna, że słowo jest dobre lub złe), zadanie jest uznane za skończone i robot się wyłącza. Jeśli dwa roboty wyrażą opinię o słowie, wtedy liczy się opinia pierwszego z nich.

W szczegółach. Robot przegląda kolejne instrukcje i próbuje **dokładnie dopasowywać** instrukcję ze swojej listy (tzn. wziąć instrukcję, która nie ma po lewej stronie znaku zapytania). Jeśli taką znajdzie, to fajnie, i ją wykonuje. W przeciwnym przypadku, robot jeszcze raz przegląda listę instrukcji i wykonuje pierwszą, która jest dopasowana (tym razem jest to instrukcja ze znakiem ?). Jeśli nadal takiej instrukcji nie ma, robot wyłącza się bez rozpoznania rodzaju słowa (zadanie uznajemy za niewykonane).

Twoim zadaniem jest stworzenie dwóch list intrukcji, po jednej dla każdego robota. Lista instrukcji powinna pozwalać poprawnie stwierdzać, czy dane słowo długości **co najmniej dwa**, złożone z liter A i B jest dobre, czy złe.

To jest zadanie typu **wygeneruj wyjście**. Twoim zadaniem jest przygotować i wysłać jeden plik tekstowy robots.txt zawierający (w tej kolejności) następujące linie:

- Linia z napisem Robot 1
- Lista instrukcji opisujących instrukcje pierwszego robota (jedna instrukcja w linii)
- Linia z napisem Robot 2
- Lista instrukcji opisujących instrukcje drugiego robota (jedna instrukcja w linii)

Plik robots.txt może zawierać komentarze, które są pomijane przez robota. Linia z komentarzem musi zaczynać się znakiem % (procent).

Ograniczenia

Liczba ruchów, które mogą wykonać roboty przed ogłoszeniem odpowiedzi (czy słowo jest dobre, czy złe) nie powinna przekroczyć 1000 * długość słowa.

Każdy test zawiera 16 lub 20 słów, złożonych z liter A lub B. Każde słowo jest długości przynajmniej 12 i nie ma więcej niż 3600 znaków.

Jeden test zawiera tylko słowa długości 12.

Dwa inne testy zawierają tylko słowa długości nie większej niż 36.

Ocenianie

Test zostanie zaakceptowany, jeśli każde słowo zostanie poprawnie rozpoznane (uznane za dobre lub złe) oraz roboty zrobią to w nie więcej nić 1000 * długość słowa ruchach. W przeciwnym przypadku program zostanie oceniony na 0 punktów.

Przykład

Dla wyjaśnienia, rozważmy prostsze zadanie. Niech robot uważa za poprawne tylko słowa długości 3, których trzeci znak to A, tzn. tylko słowa AAA, ABA, BAA i BBA uznaje jako dobre. Można to uzyskać wykonując bardzo prostą strategię. Jeden robot jest wystarczający do tego zadania, więc ten drugi może stać w miejscu. Ten pierwszy „pracujący" może wykonać dwa kroki w prawo i sprawdzić, czy znajduje się na skrajnie prawej pozycji i czy stoi na literze A.

robots.txt	**Wyjaśnienie**
Robot 1	Lista instrukcji dla robota 1.
???????->S????	Zawsze zostań na swoim miejscu.
Robot 2	Lista instrukcji dla robota 2.
L??0000->R0001	Jeśli robot znajduje się na skrajnie lewej pozycji z pamięcią 0000, idź w prawo i zapisz do pamięci 0001.
I??0001->R0010	Jeśli robot znajduje się na pozycji środkowej z pamięcia 0001, idź w prawo i zapisz do pamięci 0010.

R??0001->N	Jeśli robot znajduje się na skrajnie prawej pozycji z pamięcią 0001 – wtedy słowo zawiera tylko dwa znaki, więc odpowiedź „coarse” (złe słowo).
R?A0010->Y	Jeśli robot znajduje się na pozycji skrajnie prawej na literce A z pamięcią 0010, odpowiedź „fine” (dobre słowo).
R?B0010->N	Jeśli robot znajduje się na pozycji skrajnie prawej na litrze B z pamięcią 0010 – odpowiedź „coarse” (słowo jest złe).
I??0010->N	Jeśli robot jeszcze nie osiągnął skrajnie prawej pozycji, to znaczy, że słowo jest za długie, więc odpowiadamy “coarse” (słowo jest złe).

Interpreter

Aby przetestować lokalnie plik robots.txt można skorzystać z dostarczonego interpretera robots.cpp. Pobierz plik do folderu, w którym znajduje się plik robots.txt. Aby użyć interpretera, podaj mu na wejściu string, który wygenerowałeś. Możesz uruchomić to w trybie konsolowym, jeśli używasz skompilowanej wersji, która znajduje się w tym samym katalogu. Możesz również użyć trace mode interpretera, jeśli chcesz wykonywać wygenerowany kod krok po kroku, śledząc stan robota i czekając na ENTER z wejścia.

Uruchomienie bez parametrów (bez śledzenia krok po kroku), interpreter akceptuje słowa ze standardowego wejścia zgodnie z instrukcjami z pliku. Jeśli jeden parametr jest podany:

- jeśli jest to słowo złożone z liter A i B, wtedy jest ono sprawdzane bez śledzenia krok po kroku
- jeśli jest to słowo, które nie jest poprawne, wtedy interpreter czeka na inne wejście (być może tym razem poprawne) i przełącza na tryb śledzenia krok po kroku

Jeśli są dwa parametry, pierwszy z nich to string do sprawdzenia, a obecność drugiego uruchania tryb śledzenia krok po kroku.

Dla przykłady (tryb konsolowy):

Running robots ABAB	sprawdza słowo ABAB bez śledzenia krok po kroku
Running robots ABAB 1	sprawdza słowo ABAB ze śledzeniem krok po kroku
Running robots 1	czekanie na wejściu, które jest zostanie zinterpretowane ze śledzeniem krok po kroku

Możesz używać i zmieniać interpreter jak Ci wygodnie.