OKI++

*Uwaga: Celem tego zadania jest byś napisał(a) własną **klasę** która realizuje **język** OKI++ - przykład: https://youtu.be/WMQtAACifHc?t=543*

Python, C++, … mają mnóstwo wad. Nienaturalne komendy, dziwna składnia. No i sa nudne. Dlatego powstaje oddolna inicjatywa nowego języka programowania - OKI++. Twoim zadaniem jest napisać interpreter, który wykona program zapisany w języku OKI++.

Program w języku OKI++ to ciąg słów rozdzielonych białymi znakami. Wejście czytamy aż do napotkania słowa **nara**, które nie należy do programu.

Interpreter przechowuje zmienne całkowite oznaczone nazwami. Zmienne nie muszą być deklarowane przed użyciem — ich domyślna wartość to 0.

Polecenia języka

Składnia	Działanie
`oki doki`	Natychmiast kończy wykonywanie programu.
`krzycz " słowo1 słowo2 ... słowok "`	Wypisuje słowa zapisane między cudzysłowami, oddzielone spacjami, a potem przechodzi do nowej linii.
`chomikuj nazwa wartość`	Przypisuje zmiennej nazwę podaną wartość. Wartość może być liczbą całkowitą albo nazwą innej zmiennej.
`oby lewa operator prawa polecenie...`	Sprawdza warunek. Jeśli jest prawdziwy, następne polecenie wykonuje się normalnie. Jeśli jest fałszywy, następne polecenie zostaje pominięte wraz ze wszystkimi swoimi tokenami. Operatory porównania to dokładnie: **< > =**

Uwaga. W testach pomijane przez **oby** polecenie nigdy nie jest poleceniem **oby**.

Wejście

Wejście zawiera poprawny program w języku OKI++. Program składa się ze słów rozdzielonych białymi znakami i kończy się słowem **nara**.

- Nazwy zmiennych są ciągami małych liter łacińskich o długości co najwyżej 20.
- Stałe liczbowe mieszczą się w przedziale od -10^6 do 10^6.
- Liczba słów programu przed słowem nara nie przekracza 10^5.
- Program jest zawsze składniowo poprawny.
- Każdy program zawiera polecenie **oki doki** albo kończy się naturalnie ostatnim poleceniem po którym następuje komenda **nara**.

Wyjście

Wypisz dokładnie te wiersze, które powstają podczas wykonywania poleceń **krzycz**. Każde takie polecenie generuje osobny wiersz.

Napisz własną klasę która realizuje intepretację języka C++: https://youtu.be/WMQtAACifHc?t=543

Przykłady

Przykład 1

Wejście	`chomikuj wynik 7`
	`oby wynik > 5`
	`krzycz " Olimpiada czeka "`
	`krzycz " Ucz sie dalej "`
	`oki doki`
	`nara`

Wyjście	`Olimpiada czeka` `Ucz sie dalej`

Wyjaśnienie: Ponieważ wynik = 7, warunek wynik > 5 jest prawdziwy. Oba polecenia krzycz zostają wykonane.

Przykład 2

Wejście	`chomikuj punkty 3`
	`oby punkty > 5`
	`krzycz " Wygrales "`
	`krzycz " Sprobuj ponownie "`
	`oki doki`
	`nara`

Wyjście	`Sprobuj ponownie`

Wyjaśnienie: Warunek jest fałszywy, więc pierwsze polecenie krzycz zostaje pominięte.

Przykład 3

Wejście	`chomikuj a 10`
	`chomikuj b a`
	`oby b = 10`
	`krzycz " Zmienne dzialaja "`
	`oby a < 0`
	`krzycz " To sie nie wypisze "`
	`krzycz " A to tak "`
	`oki doki`
	`nara`

Wyjście	`Zmienne dzialaja` `A to tak`

Wyjaśnienie: Zmienna b dostaje wartość zmiennej a. Następnie drugi warunek jest fałszywy, więc tylko jeden z kolejnych komunikatów zostaje pominięty.

Podzadania

Grupa	Punkty	Dodatkowe ograniczenia
1	20	Brak poleceń oby.
2	20	W każdym poleceniu oby obie porównywane wartości są liczbami.
3	20	Liczba słów programu nie przekracza 1000.
4	40	Pełne ograniczenia.

Język OKI++ jest prosty, ale wymaga uważnego czytania słów i poprawnego pomijania całych poleceń po nieudanym warunku.