

## Задача

Централен сървър съхранява база данни с обекти и техните свойства. Отвън към него постъпват последователно следните видове команди:

1. Създаване на обект от даден тип с дадено име.
2. Премахване на обект с дадено име.
3. Променяне на стойността на дадено свойство на даден (вече създаден) обект.

Примерно:

1. Създай обект от тип "Топка" с име "топка1".
2. Промени свойството на "топка1" с име "цвят" на стойност "червен".
3. Създай обект от тип "Топка" с име "топка2".
4. Промени свойството на "топка2" с име "цвят" на стойност "зелен".
5. Промени свойството на "топка1" с име "цвят" на стойност "син".
6. Създай обект от тип "Куб" с име "куб1".
7. Премахни обект с име "топка1".
8. Промени свойството на "куб1" с име "материал" на стойност "дърво".
9. Създай обект от тип "Колело" с име "колело1"
10. Създай обект от тип "Превозно\_средство" и име "камион1"
11. Промени свойство "колела" на "камион1" на стойност "колело1" // тук "колело1" е обекта създаден в стъпка 9

За улеснение, типовете на обектите са текстови низове (стрингове), имената на обектите, имената на свойствата, както и техните стойности, също така са текстови низове. Не е нужно да се знае предварително точно какви типове обекти съществуват и точно какви видове свойства (могат да) притежават. Имената на обектите са уникални. Когато бъде създаден нов обект, той все още не притежава никакви свойства. Свойствата на обектите могат да са произволни, но много често се повтарят и много често стойностите им се променят (презаписват). Също така, стойностите на свойствата могат да са референции към вече създаден обект, както в примера по-горе в последната стъпка 11. Последното налага допълнително ограничение за реда на действията за което ще стане въпрос малко по-долу.

Множество клиенти могат да се закачат към сървъра в произволен момент - или докато базата от данни е празна (сървърът още не е получил никаква команда), или в произволен момент след това. Клиентите трябва да получат цялото текущо състояние в момента, в който са се закачили към сървъра и след това да ъпдейтват динамично състоянието си според промените, извършени на сървъра. Имплементацията трябва да се стреми да съхранява и най-вече да предава минимално количество необходими данни между сървъра и клиентите. От примера се вижда, че ако даден клиент се закачи след стъпка 7, то той няма нужда да получава излишни команди отнасящи се до обект "топка1".

Командите трябва да се получават от клиентите в правилния ред, примерно команда 11 е задължително след команди 9 и 10. Ако примерно, команда 11 се изпълни преди команда 9, приемаме, че това ще наруши правилната работа на системата (в случая, за простота, свойствата са просто стрингове, но приемаме, че е важно дали са имена на вече съществуващи обекти или не).

Повтаряме, клиентите могат да са много на брой и всеки от тях може да се е закачил в произволен момент. Примерно, ако следваме горните стъпки, може клиент1 да се е закачил преди стъпка 1, клиент2 - след стъпка 6, клиент 3 - след стъпка 8 и т.н. Предаването на данните от сървъра към клиентите е най-бавната част в системата и затова количеството данни, което се предава, трябва да се минимизира.

Броят на обектите може да е голям (примерно десетки хиляди), а броят на операциите - много голям (примерно стотици хиляди). Броят на свойствата на даден обект е относително малък - най-много 20-30 и те най-често се презаписват с нови стойности.

Клиентите трябва да могат да покажат (примерно в прост текстов вид) текущото състояние на базата данни във всеки един момент.

Позволено е използването на C++11, но не е позволено използването на външни библиотеки с изключение на частта за комуникация между сървъра и клиентите. Решение на C#, Python или Node.js също е допустимо.

Решението трябва да включва сорс код и изпълним файл или файлове за Windows или Linux. Разполагаш с две седмици за подготовката му.

Успех!