Chaotic Evil

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Рассмотрим язык, чуть более похожий на Си, чем в прошлый раз. В нём есть следующие простые типы:

Имя типа	Размер
`bool`	1 байт
`char` или `signed char`	1 байт
`unsigned char`	1 байт
`short`, `signed short`, `short int` или `signed short int`	2 байта
`unsigned short` или `unsigned short int`	2 байта
`int`, `signed` или `signed int`	4 байта
`unsigned` или `unsigned int`	4 байта
`long`, `signed long`, `long int` или `signed long int`	8 байт
`unsigned long` или `unsigned long int`	8 байт
`long long`, `signed long long`, `long long int` или `signed long long int`	8 байт
`unsigned long long` или `unsigned long long int`	8 байт

В языке есть оператор `sizeof`, который позволяет узнать размер любого типа в байтах. К примеру, `sizeof(int)` равен четырём.

В языке есть оператор `alignof`, который позволяет узнать **выравнивание** любого типа в байтах. Адрес переменной какого-то типа `T` в памяти должен делиться на `alignof(T)`. `sizeof(T)` всегда делится на `alignof(T)`. `alignof(T)` всегда является неотрицательной целой степенью двойки. Для простых типов, `alignof(T) == sizeof(T)`.

В языке есть массивы фиксированной длины, состоящие из элементов одного типа. Массив из `n` элементов, каждый типа `T` обозначается как `T[n]`. `sizeof(T[n])` равен `sizeof(T) * n`. К примеру, `sizeof(short[13])` равен 26, так как размер типа `short` — два байта, а в массиве 13 элементов. `alignof(T[n])` равен `alignof(T)`. Поддержки многомерных массивов в языке нет.

В языке есть структуры — композитные типы, позволяющие объединять фиксированное количество переменных (полей) разных типов в одну. Пусть в структуре $n > 0$ полей $f_1, \ldots, f_n$ типов $T_1, \ldots, T_n$. Пусть эта структура лежит в памяти по адресу a. Тогда должны выполняться следующие дополнительные условия:

- Адрес f_1 равен a.
- Для $k = 2, \ldots, n$, Адрес f_k больше адреса f_{k-1}.
- Поля не могут пересекаться
- Как и для самой структуры, так и для всех её полей должны выполнятся стандартные правила выравнивания.
- Выравнивание структуры — максимум из выравниваний её полей.
- Размер структуры не меньше суммы размеров её полей.
- Размер структуры — минимальный из размеров, удовлетворяющий всем условиям.

Вам предлагается написать программу, вычисляющую `sizeof` и `alignof` для произвольных типов.

Формат входных данных

Во вводе записаны команды, `typedef`, `sizeof` или `alignof`.

Команда `typedef` объявляет новый тип. Например, `typedef eightbytes unsigned char[8]` объявляет новый тип `eightbytes`, который представляет собой массив из восьми `unsigned char`. `typedef` может также объявлять структуры. Смотрите примеры. Гарантируется, что имя нового типа — непустая строка из латинских букв длиной не более 32 символов, кроме `bool`, `char`, `signed`, `unsigned`, `short`, `int`, `long`, `struct`, `typedef`, `sizeof`, `alignof`

Гарантируется, что объявления новых типов имеют уникальные имена.

Команды `sizeof` и `alignof` печатают на экран на новой строке размер и выравнивание типа соответственно. Например, `sizeof unsigned char[8]` напечатает на экран `8`.

Гарантируется, что размер входных данных по объёму не превосходит одного мегабайта. Размер каждого используемого типа не превосходит одного эксабайта.

Формат выходных данных

Для каждой команды `sizeof` или `alignof` в отдельной строке напечатайте результат выполнения соответствующего оператора.

Примеры

стандартный ввод	стандартный вывод
typedef eightbytes unsigned char[8] sizeof eightbytes alignof eightbytes	8 1

typedef verylong struct {
long[2];
unsigned short int[4];
}
sizeof verylong
alignof verylong
	24
	8

typedef verylong struct {
long[2];
unsigned short int[4];
}
typedef evenlonger struct {
verylong[4];
}
sizeof evenlonger
alignof evenlonger
typedef arr evenlonger[123]
sizeof arr
alignof arr
	96
	8
	11808
	8