Задача 1

Даны значения температуры, наблюдавшиеся в течение N подряд идущих дней. Найдите номера дней (в нумерации с нуля) со значением температуры выше среднего арифметического за все *N* дней. Гарантируется, что среднее арифметическое значений температуры является целым числом.

Формат ввода

Вводится число *N*, затем *N* неотрицательных целых чисел — значения температуры в 0-й, 1-й, ... (*N*−1)-й день.

Формат вывода

Первое число *K* — количество дней, значение температуры в которых выше среднего арифметического. Затем *K* целых чисел — номера этих дней.

Пример

Ввод	Вывод
5 7 6 3 0 9	3 0 1 4

Задача 2

Люди стоят в очереди, но никогда не уходят из её начала, зато могут приходить в конец и уходить оттуда. Более того, иногда некоторые люди могут прекращать и начинать беспокоиться из-за того, что очередь не продвигается.

Будем считать, что люди в очереди нумеруются целыми числами, начиная с 0.

Реализуйте обработку следующих операций над очередью:

- **WORRY *i***: пометить ***i***-го человека с начала очереди как беспокоящегося;
- **QUIET *i***: пометить ***i***-го человека как успокоившегося;
- **COME *k***: добавить ***k*** спокойных человек в конец очереди;
- **COME -*k***: убрать ***k*** человек из конца очереди;
- **WORRY_COUNT**: узнать количество беспокоящихся людей в очереди.

Изначально очередь пуста.

Формат ввода

Количество операций *Q*, затем описания операций.

Для каждой операции **WORRY *i*** и **QUIET *i*** гарантируется, что человек с номером ***i*** существует в очереди на момент операции.

Для каждой операции **COME -*k*** гарантируется, что ***k*** не больше текущего размера очереди.

Формат вывода

Для каждой операции **WORRY_COUNT** выведите одно целое число — количество беспокоящихся людей в очереди.

Пример

Ввод	Вывод
8 COME 5 WORRY 1 WORRY 4 COME -2 WORRY_COUNT COME 3 WORRY 3 WORRY_COUNT	1 2

Задача 3

У каждого из нас есть повторяющиеся ежемесячные дела, каждое из которых нужно выполнять в конкретный день каждого месяца: оплата счетов за электричество, абонентская плата за связь и пр. Вам нужно реализовать работу со списком таких дел на месяц, а именно, реализовать набор следующих операций:

ADD *i s*
Назначить дело с названием ***s*** на день ***i*** текущего месяца.

DUMP *i*
Вывести все дела, запланированные на день ***i*** текущего месяца.

NEXT
Перейти к списку дел на новый месяц. При выполнении данной команды вместо текущего (старого) списка дел на текущий месяц создаётся и становится активным (новый) список дел на следующий месяц: все дела со старого списка дел копируются в новый список. После выполнения данной команды новый список дел и следующий месяц становятся текущими, а работа со старым списком дел прекращается. При переходе к новому месяцу необходимо обратить внимание на разное количество дней в месяцах:

- если следующий месяц имеет больше дней, чем текущий, «дополнительные» дни необходимо оставить пустыми (не содержащими дел);
- если следующий месяц имеет меньше дней, чем текущий, дела со всех «лишних» дней необходимо переместить на последний день следующего месяца.

Замечания

- Историю списков дел хранить **не требуется**, работа ведется только с текущим списком дел текущего месяца. Более того, при создании списка дел на следующий месяц, он **«перетирает»** предыдущий список.
- Обратите внимание, что количество команд **NEXT** в общей последовательности команд при работе со списком дел может превышать 11.
- Начальным текущим месяцем считается январь.
- Количества дней в месяцах соответствуют Григорианскому календарю с той лишь разницей, что в феврале всегда 28 дней.

Формат ввода

Сначала число операций *Q*, затем описания операций.

Названия дел ***s*** уникальны и состоят только из латинских букв, цифр и символов подчёркивания. Номера дней ***i*** являются целыми числами и нумеруются от 1 до размера текущего месяца.

Формат вывода

Для каждой операции типа **DUMP** в отдельной строке выведите количество дел в соответствующий день, а затем их названия, разделяя их пробелом. Порядок вывода дел в рамках каждой операции значения не имеет.

Пример

Ввод	Вывод
12 ADD 5 Salary ADD 31 Walk ADD 30 WalkPreparations NEXT DUMP 5 DUMP 28 NEXT DUMP 31 DUMP 30 DUMP 28 ADD 28 Payment DUMP 28	1 Salary 2 WalkPreparations Walk 0 0 2 WalkPreparations Walk 3 WalkPreparations Walk Payment

Указание

Для дописывания всех элементов вектора v2 в конец вектора v1 удобно использовать метод insert:

```
v1.insert(end(v1), begin(v2), end(v2));
```

Кроме того, элементом вектора может быть любой тип, в том числе и другой вектор. Например, vector<vector<int>> — это вектор, элементами которого являются вектора целых чисел (то есть двумерный массив). Пример использования:

```
vector<vector<int>> m(10); // Создаём вектор из десяти векторов целых чисел
m[0].push_back(5);  // Добавляем элементы в первый вектор
m[0].push_back(15);
cout << m[0][1]; // Выведет 15 – второй элемент первого вектора

m[1].push_back(3);
for (int x : m[1]) {
    // Перебираем все элементы второго
}
```