

# Задание 2

## Условие

Программа из стандартного ввода читает данные. Данные хранятся построчно. Каждая строка состоит из трех полей, разделенных одним символом табуляции, и завершается символом конца строки.

Формат строки:

```
text1 \t text2 \t text3 \n
```

Поля text2 и text3 игнорируются. Поле text1 имеет следующую структуру (ip4 address):

n1.n2.n3.n4

где n1..4 – целое число от 0 до 255.

Необходимо загрузить список ip-адресов в память и отсортировать их в обратном лексикографическом порядке. Пример лексикографической сортировки (по первому числу, затем по второму и так далее):

1.1.1.1

1.2.1.1

1.10.1.1

Соответственно, обратная:

1.10.1.1

1.2.1.1

1.1.1.1

Далее выводим в стандартный вывод следующее:

1. Полный список адресов после сортировки. Одна строка - один адрес.
2. Сразу следом список адресов, первый байт которых равен 1. Порядок сортировки не меняется. Одна строка - один адрес. Списки ничем не разделяются.
3. Сразу продолжается список адресов, первый байт которых равен 46, а второй 70. Порядок сортировки не меняется. Одна строка - один адрес. Списки ничем не разделяются.
4. Сразу продолжается список адресов, любой байт которых равен 46. Порядок сортировки не меняется. Одна строка - один адрес. Списки ничем не разделяются.

## Требования к реализации

В приложенном к заданию исходном файле необходимо заменить, где это возможно, конструкции на аналогичные из стандарта C++14. Реализовать недостающий функционал.

## Лишний раз проверьте

1. лексикографическая сортировка понятна как для строки, так и для контейнера
2. выбрана соответствующая задаче структура данных

## Самопроверка

Макет исходного кода, а также тестовый файл с данными ip\_filter.tsv прилагается к материалам занятия. Проверить себя можно следующим образом:

```
cat ip_filter.tsv | ip_filter | md5sum  
24e7a7b2270daee89c64d3ca5fb3da1a -
```