Задания финального тура

«Стажировка Николая Иваныча в крупной федеральной компании»

Олимпиада по веб-программированию, Академия «1С-Битрикс», фирма «1С», 2018 год

A. Работа над ошибками

Имя входного файла: input.html
Имя выходного файла: output.html

Не прошло и года с начала работы на новом месте, как Николай Иваныч получил повышение и должность “Старший программист”. Теперь нужно было не только писать код, но и приглядывать за 2-я стажерами. Стажеры были очень деятельными но крайне не внимательными. Регулярно допускали ошибки на ровном месте.

Стажеры по ошибке импортировали в базу данных ссылки без специальных символов (:, /, .). Вся информация склеилась и потеряла смысл. Необходимо написать скрипт который восстановит ссылки.

Известно что ссылки имели формат:
“протокол”://”домен”.ru|com”/<контекст>”,
где протокол - http или https,
домен - непустая строка из строчных латинских букв,
контекст - может отсутствовать, если присутствует то это непустая строка из строчных латинских букв.

Входные данные

В единственной строке содержится битая ссылка.

Выходные данные

Вывести одну строку - восстановленную ссылку.
Гарантируется, что восстановить ссылку возможно. Если существует несколько вариантов исходной ссылки, вывести тот, в котором протокол имеет наибольшую длину, а хост - наименьшую.
Не печатать слеш в конце ссылки если за ним нет url-адреса.

Пример

Входные данные	Результат работы
https1c-bitrixruproducts	https://1c-bitrix.ru/products

B. Города

Имя входного файла: input.html
Имя выходного файла: output.html

Николай Иваныч открыл почту и увидел письмо от заказчика с текстом: “Необходимо определить города жители которых посещают наш сайт”.

На просторах интернета есть много баз данных с привязками IP адресов к городам. После анализа стало ясно, что базы данных содержат частичную информацию. Не все IP адреса есть в базах данных. Потратив несколько часов на анализ, Николай Иваныч заметил, что адреса из одной подсети всегда относятся к одному городу. Таким образом, определив маску подсети, можно определить, к какому городу относится IP адрес.

Входные данные

В первой строке два целых числа n и k через пробел ($1 \le k \le n \le 10^5$), где n - количество ip-адресов, k - количество сетей к которым они принадлежат.
Далее в n-строках указываются ip-адреса, по одному в строке.

Выходные данные

В единственной строке необходимо вывести маску подсети.

Пример

Входные данные	Результат работы
5 3 0.0.0.1 0.1.1.2 0.0.2.1 0.1.1.0 0.0.2.3	255.255.254.0

C. Фильтр

Имя входного файла: input.html
Имя выходного файла: output.html

Николай Иванычу представилась возможность поработать над интернет-магазином бытовой техники. Это был крупный проект, в каталоге несколько десятков тысяч единиц бытовой техники: телевизоры, смартфоны, холодильники, умные часы и т.д.

Все товары лежат в одной базе данных, но отличаются значениями свойств. Для удобной работы с каталогом товаров в нем предусмотрен фильтр, который позволяет отобрать товары с определенными значениями свойств. Например все телевизоры производителя Sony с диагональю 40 дюймов.

После того, как пользователь расставил галочки в фильтре, он формирует JSON, в котором перечислены параметры. Николай Иванычу нужно написать php-скрипт, который будет преобразовывать JSON в запрос к MySQL.

Входные данные

На вход подается json следующей структуры:

```
{
    "select": ["field1", "field2"],
    "from": "table_name",
    "where": {"op_field": "value", "or": {"op_field": "value", "and": {"op_field": "value"}}},
    "order": {"field": "how"},
    "limit": n
}
```

Где:

1. select - массив полей для итоговой выборки (необязательное поле, может быть пустым)
2. from - название таблицы (обязательное поле, не может быть пустым)
   order - объект с одним ключом, в котором ключ обозначает поле, а значение - порядок сортировки по данному полю (необязательное поле, может быть пустым)
3. limit - целое число - ограничение на количество выбираемых строк (необязательное поле, может быть пустым)
4. where - объект, описывающий условия выборки (необязательное поле, может быть пустым):
   a. Ключами являются [операция]название поля, а значениями - соответствующие значения условий
   b. Между всеми условиями внутри where стоит логическое "И". Например: {"<first": 10, "=second": 10} представляет условие first < 10 and second = 10
   c. Значениями могут быть числа, строки, булевы значения и null. При генерации sql только строки должны быть обернуты в кавычки.
   d. Используемые операции: =, <, <=, >, >=, !
      При этом операции <, <=, >, >= для всех типов транслируются в одни и те же sql операторы:
      {"<field": "val"} => field < val

{"<=field": "val"} => field <= val
{">field": "val"} => field > val
{">=field": "val"} => field >= val

А остальные операции зависят от типов:
{"field": "str"} => field like 'str'
{"field": "num"} => field = num
{"field": "bool"} => field is bool
{"field": "null"} => field is null
{"=field": "str"} => field = 'str'
{"=field": "num"} => field = 'num'
{"=field": "bool"} => field is bool
{"=field": "null"} => field is null
{"!field": "str"} => field != 'str'
{"!field": "int"} => field != int
{"!field": "bool"} => field is not bool
{"!field": "null"} => field is not null

e. Также ключ может иметь вид and_S или or_S (где S - произвольная уникальная строка), значением которого должен быть ещё один объект, описывающий условия выборки.
f. Между условиями вложенного объекта, лежащего по ключу and или or стоит соответственно логическое "И" или "ИЛИ"
Например: {"<first": 10, "=second": 10, "or": {"third": "val", "<=fourth": 20, "and": {">fifth": 30, "!sixth": 40}}}
Обозначает условие: first < 10 and second = 10 and (third like 'val' or fourth <= 20 or (fifth > 30 and sixth != 40))

Выходные данные

Получившийся sql-запрос

Пример

Входные данные

<pre>{
"select": ["first", "second", "third", "fourth", "fifth", "sixth"],
"from": "test_table",
"where": {"<first": 10, "=second": 10, "or_1": {"third": "val", "<=fourth": 20, "and_2": {">fifth": 30, "!sixth": 40}}},
"order": {"first": "asc"},
"limit": 10
}</pre>

Выходные данные

<pre>select first, second, third, fourth, fifth, sixth
from test_table
where first < 10 and second = 10 and (third like 'val' or fourth <= 20 or (fifth > 30 and sixth != 40))
order by first asc
limit 10;</pre>

D. Карты, сайты, XML

Имя входного файла: input.html
Имя выходного файла: output.html

Николай Иваныча вместе с командой пригласили помочь с большим проектом. Это был портал Администрации г.Москвы. Сайт огромный, очень много разделов, масса документов, постановлений, нормативных актов и отчетов.

Для того чтобы сайт хорошо и быстро индексировался поисковыми системами необходимо иметь на сайте специальный XML файл. Такой файл называется “карта сайта”. Он отвечает отлично отвечает на вопрос, как разделы и страницы вложены друг в друга и когда менялись последний раз.

Входные данные

Список разделов по одному в каждой строке.
Каждый раздел описан следующими полями, разделенными точкой с запятой: ID;URL;PARENT_ID;TIME. Где:

1. ID - идентификатор раздела: положительное целое число
2. URL - url раздела: может содержать буквы, цифры и спецсимволы. не может содержать точки с запятой
3. PARENT_ID - идентификатор родительского раздела: неотрицательное целое число, у корневых разделов равен нулю
4. TIME - время последнего изменения: timestamp

Выходные данные

Необходимо вывести sitemap сайта в формате:

```
<urlset xmlns="https://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
    <loc>http://example.com/</loc>
    <lastmod>2018-05-30T14:15:27+03:00</lastmod>
  </url>
</urlset>
```

Где loc - адрес раздела, lastmod время последнего изменения в формате стандарта ISO 8601.
Время последнего изменения во входных данных отражает время изменения непосредственно раздела. Если у текущего раздела есть дочерние разделы, то время его последнего изменения будет зависеть от них и должно быть вычислено как максимальное среди времени всех потомков и времени текущего раздела.
Разделы в sitemap'е должны быть отсортированы по ID.

Пример

Входные данные

```
6;https://site.org/;0;1548933472
3;https://site.org/contacts/;6;1549521832
1;https://site.org/vacancy/;3;1548933472
```

Выходные данные

```
<urlset
xmlns="https://www.sitemaps.org/schemas/sitemap/0.9"><url><loc>https://site.org/vacancy/</loc><
lastmod>2019-01-
31T14:17:52+03:00</lastmod></url><url><loc>https://site.org/contacts/</loc><lastmod>2019-02-
07T09:43:52+03:00</lastmod></url><url><loc>https://site.org/</loc><lastmod>2019-02-
07T09:43:52+03:00</lastmod></url></urlset>
```