

# 1. (10)

Ограничение времени	1 секунда
Ограничение памяти	64 Мб

Ниже представлен скрипт, создающий одну из таблиц базы данных, описывающей библиотеку музыкального стримингового сервиса:

```
CREATE TABLE IF NOT EXISTS "Track"
(
    "TrackId"      INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    "Name"         NVARCHAR(200)                     NOT NULL,
    "AlbumId"      INTEGER,
    "MediaTypeId"  INTEGER                           NOT NULL,
    "GenreId"      INTEGER,
    "Composer"     NVARCHAR(220),
    "Milliseconds" INTEGER                           NOT NULL,
    "Bytes"        INTEGER,
    "UnitPrice"    NUMERIC(10, 2)                     NOT NULL,

    FOREIGN KEY ("AlbumId") REFERENCES "Album" ("AlbumId")
        ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY ("GenreId") REFERENCES "Genre" ("GenreId")
        ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY ("MediaTypeId") REFERENCES "MediaType" ("MediaTypeId")
        ON DELETE NO ACTION ON UPDATE NO ACTION
);
```

Новому стажёру отдела бизнес-аналитики было поручено изучить имеющиеся данные и написать некоторый запрос. В итоге стажёр написал следующий параметризованный запрос:

```
SELECT "TrackId",
       "Milliseconds",
       Sum("Milliseconds") OVER (
           PARTITION BY "GenreId"
           ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING
           EXCLUDE CURRENT ROW
       ) "Result"
FROM "Track"
WHERE "GenreId" = $genre
ORDER BY "TrackId";
```

Через некоторое время глава отдела, увидев этот запрос, решил проверить, будет ли выполняться быстрее этот запрос или запрос с подзапросом, однако глава отдела не знал, какие данные нужно было получить стажёру. Помогите ему и напишите запрос, получающий те же данные, что и запрос выше, но с использованием подзапроса.

В качестве ответа напишите один параметризованный SQL-запрос, который будет использовать ровно два оператора SELECT.

## Формат ввода

## Формат ввода

При выполнении запроса будет передан один параметр: ID жанра без кавычек. В запросе этот параметр должен указываться как `$genre`.

## Формат вывода

В качестве результата запрос должен возвращать выборку, описанную в условии. Ваш запрос должен возвращать такую же выборку, как и запрос в условии, с учётом порядка строк и столбцов.

## Примечания

У вас есть **5 попыток на отправку ответа** к заданию. Запрос с синтаксическими ошибками будет засчитан как неправильный ответ и использует попытку, поэтому рекомендуем проверить корректность синтаксиса запроса на локальном компьютере до отправки. При подсчёте баллов будет учитываться только последний отправленный ответ.

Для проверки используется СУБД SQLite, но гарантируется, что необходимый запрос выполнится корректно в СУБД SQLite, PostgreSQL и MS SQL Server. При необходимости вместо кавычек при указании названия полей и таблиц можно использовать квадратные скобки.

## 2. (12)

Одной из основных функций операционной системы является эффективное распределение ресурсов между процессами. Возьмем уже известную по предыдущему этапу олимпиады модель, в которой под управлением операционной системы работает два процесса:  $p_1$  и  $p_2$ . Каждый процесс попеременно реализует некоторое количество операций вычисления на процессоре и некоторое количество операции ввода-вывода. В системе один процессор, который в один момент времени может исполнять вычислительные операции только одного процесса. Подсистема ввода-вывода может независимо и параллельно обрабатывать операции ввода-вывода обоих процессов без изменения производительности. Время модели дискретно и измеряется в условных тактах. Для каждого процесса заданы в этих условных тактах два параметра: CPU-burst и I/O-burst – время непрерывного выполнения вычислительных операций и время непрерывного выполнения операций ввода-вывода.

Для управления распределением процессорного времени используется алгоритм Round Robin. Каждому процессу поочередно дается квант непрерывного выполнения на процессоре, равный  $K$  тактов. Реализация алгоритма отвечает следующим правилам:

1. Процессам поочередно предоставляется возможность непрерывного использования процессора, продолжительностью, не превышающей квант непрерывного исполнения. В первый такт такая возможность предоставляется процессору  $p_1$ . При этом процесс  $p_2$  также готов исполняться на процессоре.
2. Если процесс использовал целиком квант непрерывного исполнения (возможно, полностью исчерпав очередной CPU-burst), он вытесняется и процессор передается другому процессу. На переключение между процессами тратится фиксированное время – 5 тактов.
3. Если процесс, не использовав целиком квант непрерывного исполнения полностью, исчерпывает свой CPU-burst, он незамедлительно начинает операции ввода-вывода, продолжающиеся количество тактов, равное I/O-burst этого процесса. Одновременно с этим начинается переключение на другой процесс, которое также длится 5 тактов.
4. Если после переключения процессов, у очередного процесса есть остаток CPU-burst в т.ч. он только что завершил очередной цикл операций ввода-вывода, процесс выполняется в соответствии с пунктами 2 или 3.
5. Если после переключения процессов оказывается, что у очередного процесса не завершился цикл операций ввода-вывода, незамедлительно запускается переключение на другой процесс, которое длится также 5 тактов.
6. Операции ввода-вывода не прерываются и могут выполняться параллельно с вычислениями другого процесса или операциями переключения процессов.

Модель была использована для проведения двух экспериментов, отличающихся параметром I/O-burst процесса  $p_2$ . Параметры модели приведены в таблице.

Процесс	CPU-burst	I/O-burst 1	I/O-burst 2
$p_1$	13	17	17
$p_2$	23	47	57

В качестве метрики качества было выбрано суммарное количество тактов, в которые на процессоре вычислялся один из процессов в течение первых 11000 тактов работы модели (то есть количество тех тактов, из первых 11000, в которые процессор вычислял или процесс  $p_1$  или процесс  $p_2$ ).

Определите для каждого эксперимента минимальное значение кванта непрерывного выполнения  $K$ , при котором значение метрики будет максимальным.

*В ответе укажите через пробел два числа: сначала значение для эксперимента 1 и затем значение для эксперимента 2.*

### 3. (10)

Интернет состоит из отдельных крупных сетей, управляемых отдельными организациями. Между такими сетями распределяется пространство IP адресов и устанавливается техническое и организационное взаимодействие. Несколько упрощая, можно сказать, что такие крупные сети и называются автономными системами (AS). За каждой AS закрепляется IP подсеть. AS бывают транзитными, через которые трафик передается в другие AS и конечными.

Для обмена маршрутной информацией между AS используется внешний протокол динамической маршрутизации BGP (Border Gateway Protocol). BGP выбирает наилучший маршрут для передачи пакетов на основе ряда атрибутов и метрик.

BGP-роутер при маршрутизации пакета выбирает IP адрес следующего по маршруту маршрутизатора, который в BGP называется «соседом» (neighbor), по набору атрибутов маршрутов. Будем считать, что реализуется следующий алгоритм. По адресу назначения в IP пакете выбираются подходящие маршруты из таблицы маршрутизации по полю «сеть назначения». Потом из них выбирается лучший для чего **последовательно** проверяются атрибуты маршрутов (то есть сначала проверяется первый атрибут всех подходящих маршрутов и если у возможных маршрутов он имеет равное значение, проверяется второй атрибут и т.д.).

В задаче используются следующие атрибуты в указанном порядке:

#### 1. Локальное предпочтение (Local Preference)

Local Preference — это атрибут, который используется для выбора исходящего трафика из автономной системы. Маршрут с наибольшим значением Local Preference считается предпочтительным.

#### 2. Маршруты, объявленные локально (Local Routes)

Если маршрут был объявлен локально, он предпочитается другим маршрутам.

#### 3. Наименьший Путь AS (AS Path)

BGP предпочитает маршруты с наименьшим количеством автономных систем в пути (AS Path). Чем короче путь, тем лучше.

#### 4. Наименьший Origin Type (Тип происхождения)

BGP использует три типа происхождения маршрутов (перечислены в порядке уменьшения приоритета):

- **IGP** (Internal Gateway Protocol) — маршрут, полученный из внутреннего протокола маршрутизации (например, OSPF, EIGRP). Это наиболее предпочтительный тип.
- **EGP** (Exterior Gateway Protocol) — маршрут, полученный через протокол EGP (устаревший).
- **Incomplete** — маршрут, источник которого неизвестен. Это наименее предпочтительный тип.

#### 5. Наименьший IGP Metric до Next Hop

Если есть несколько маршрутов с одинаковыми атрибутами, BGP выбирает маршрут с наименьшим значением метрики IGP до Next Hop (следующего прыжка).

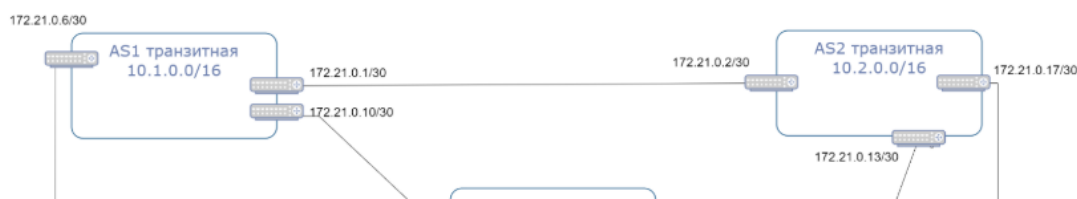
#### 6. Старейший маршрут (Oldest Route)

Если все предыдущие атрибуты равны, BGP предпочитает старейший маршрут, так как он считается более стабильным.

#### 7. Наименьший IP-адрес соседа

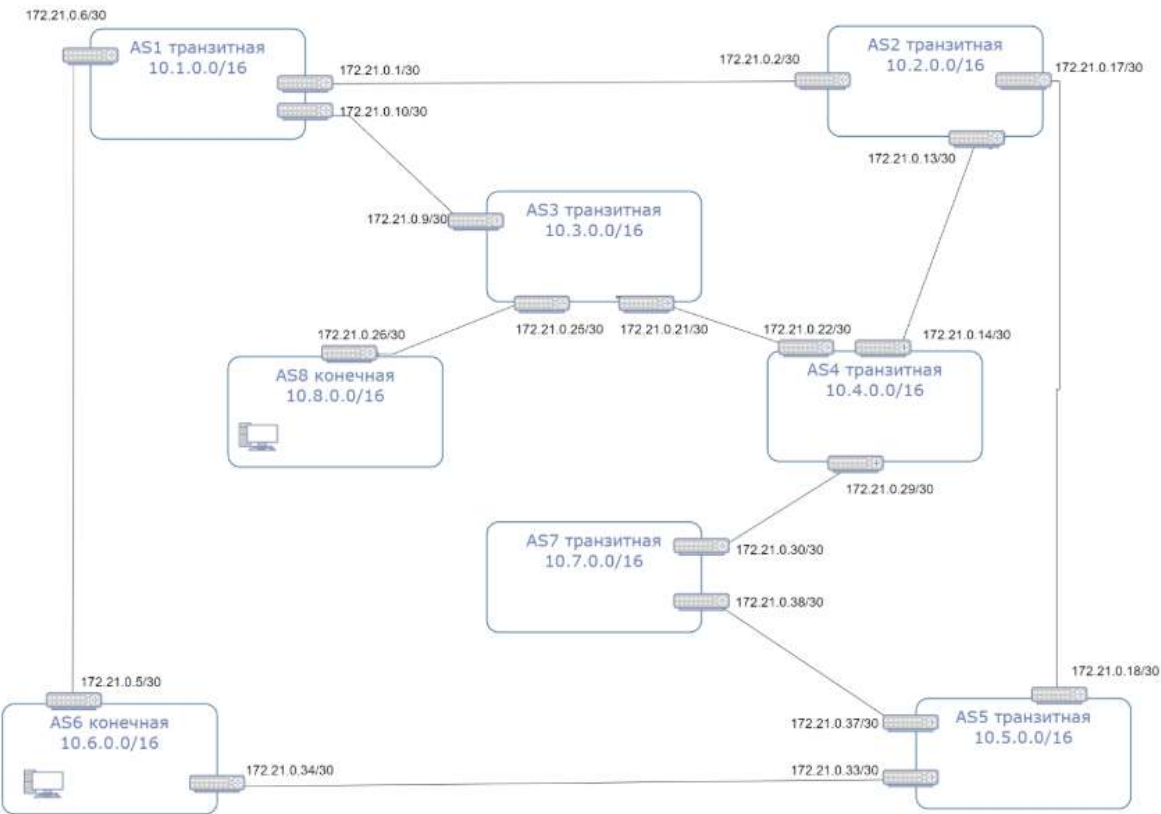
В крайнем случае, если все предыдущие атрибуты равны, BGP выбирает маршрут, полученный от соседа с наименьшим IP-адресом.

Далее приведены схема сети и фрагменты таблиц маршрутов для каждой из AS (сделано это для простоты, в реальности эти данные есть в каждом граничном маршрутизаторе AS).





Далее приведены схема сети и фрагменты таблиц маршрутов для каждой из AS (сделано это для простоты, в реальности эти данные есть в каждом граничном маршрутизаторе AS).



AS1

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.5.0.0/16	AS2	100	Нет	2	IGP	20	2023-10-02	172.21.0.2
10.1.0.0/16	Локально	500	Да	0	IGP	0	2023-10-01	-
10.7.0.0/16	AS2	100	Нет	3	IGP	20	2023-10-03	172.21.0.2
10.8.0.0/16	AS3	100	Нет	2	IGP	20	2023-10-01	172.21.0.9
10.6.0.0/16	AS6	100	Нет	2	IGP	20	2023-10-01	172.21.0.5
10.6.0.0/16	AS3	100	Нет	2	IGP	20	2023-10-01	172.21.0.9

AS2

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.2.0.0/16	Локально	500	Да	0	IGP	0	2023-10-01	-
10.6.0.0/16	AS1	100	Нет	1	IGP	20	2023-10-01	172.21.0.1
10.7.0.0/16	AS4	100	Нет	2	IGP	20	2023-10-02	172.21.0.14
10.8.0.0/16	AS4	100	Нет	2	IGP	20	2023-10-03	172.21.0.14
10.6.0.0/16	AS5	100	Нет	1	IGP	20	2023-10-03	172.21.0.18

AS3

AS3

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.5.0.0/16	AS4	100	Нет	2	IGP	20	2023-10-02	172.21.0.22
10.6.0.0/16	AS1	100	Нет	2	IGP	20	2023-10-01	172.21.0.10
10.6.0.0/16	AS4	200	Нет	3	IGP	10	2023-10-03	172.21.0.22
10.7.0.0/16	AS4	100	Нет	3	IGP	10	2023-10-04	172.21.0.22
10.3.0.0/16	Локально	500	Да	0	IGP	0	2023-10-01	-
10.8.0.0/16	AS8	100	Нет	3	IGP	10	2023-10-04	172.21.0.26

AS4

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.5.0.0/16	AS2	100	Нет	2	IGP	20	2023-10-02	172.21.0.13
10.6.0.0/16	AS2	100	Нет	2	IGP	5	2023-10-03	172.21.0.13
10.7.0.0/16	Локально	500	Да	0	IGP	0	2023-10-01	-
10.6.0.0/16	AS7	100	Нет	3	IGP	10	2023-10-01	172.21.0.30
10.8.0.0/16	AS3	100	Нет	1	IGP	10	2023-10-01	172.21.0.21
10.6.0.0/16	AS3	100	Нет	3	Incomplete	5	2023-10-01	172.21.0.21

AS5

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.4.0.0/16	AS2	100	Нет	1	IGP	20	2023-10-02	172.21.0.17
10.5.0.0/16	Локально	500	Да	0	IGP	0	2023-10-01	-
10.6.0.0/16	AS6	100	Нет	1	IGP	20	2023-10-03	172.21.0.34
10.8.0.0/16	AS7	100	Нет	3	IGP	20	2023-10-01	172.21.0.38

AS6

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.5.0.0/16	AS1	100	Нет	3	IGP	20	2023-10-02	172.21.0.6
10.6.0.0/16	Локально	500	Да	0	IGP	0	2023-10-01	-
10.7.0.0/16	AS1	100	Нет	4	IGP	20	2023-10-03	172.21.0.6
10.8.0.0/16	AS1	100	Нет	3	IGP	20	2023-10-01	172.21.0.6

**AS7**

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.5.0.0/16	AS4	100	Нет	3	IGP	20	2023-10-02	172.21.0.29
10.6.0.0/16	AS5	100	Нет	2	IGP	20	2023-10-03	172.21.0.37
10.7.0.0/16	Локально	500	Да	0	IGP	0	2023-10-01	-
10.8.0.0/16	AS4	100	Нет	3	IGP	20	2023-10-01	172.21.0.29

**AS8**

Сеть назначения	Номер следующей AS	Local Preference	Флаг локального объявления маршрута	Длина AS Path	Origin Type	IGP Metric до Next Hop	Дата появления маршрута	Адрес соседа
10.6.0.0/16	AS3	200	Нет	3	IGP	10	2023-10-01	172.21.0.25
10.7.0.0/16	AS3	200	Нет	3	EGP	10	2023-10-05	172.21.0.25
10.5.0.0/16	AS3	200	Нет	2	Incomplete	10	2023-10-03	172.21.0.25

Определите маршрут прохождения пакета из AS8 в AS6.

В ответ запишите номера автономных систем через запятую начиная с 8 и заканчивая 6.

Пример ввода ответа: 8,10,12,6.

## 4. (8)

При перефразировании исходного предложения путем различных перестановок пяти частей текста было получено шесть вариантов текстов с сохранением естественности текста.

Исходный текст: *по последним вторникам месяца (L) в главном корпусе университета (T) проходят (V) встречи (S) спортивные соревнования (M).*

Перефразированные предложения представлены следующими последовательностями пяти частей текста LTVSM:

1. LTVMS,
2. TLVMS,
3. TVMSL,
4. TVSML,
5. LVMST,
6. LVSMT.

Для сравнения близости перефразированного и исходного текстов используется метрика BLEU (BiLingual Evaluation Understudy):

$$BLEU_n = \left( \prod_{i=1}^n \frac{k_i}{l_i} \right) \cdot 100\%$$

где  $k_i$  – количество совпадающих  $i$ -грамм у исходного и перефразированного текста;  $l_i$  – общее количество  $i$ -грамм перефразированного текста.

Необходимо вычислить значение  $BLEU_4$  для оценки близости исходного предложения и перефразированных предложений. После этого нужно отсортировать перефразированные предложения по мере убывания значения  $BLEU_4$  и в ответе перечислить номера перефразированных предложений, для которых значение  $BLEU_4$  больше 50%.

Например, если значения  $BLEU_4$  равны 48%, 59%, 78%, 20%, 96%, 54%, в ответе необходимо записать **5326**.



## 5. (9)

В связном неориентированном графе с 2025 вершинами есть ровно три точки сочленения. Какое минимальное значение может принимать сумма степеней этих трех вершин?

## 6. (10)

Для контекстно-свободной грамматики  $G$  построили дерево разбора для некоторого слова  $x$  длиной 2025. В этом дереве оказалось суммарно 2500 вершин, включая как вершины, которые соответствуют нетерминалам, так и вершины-листья, соответствующие терминалам. Для какого минимального значения  $L$  может выполняться условие, что в грамматике  $G$  правые части всех правил имеют длину не больше  $L$ ?

# 7. (7)

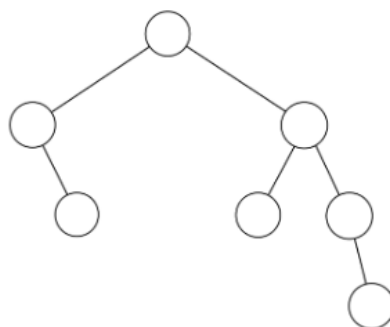
Ограничение времени	1 секунда
Ограничение памяти	512 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

АВЛ-дерево — структура данных, изобретенная Адельсоном-Вельским и Ландисом.

АВЛ-дерево представляет собой подвешенное дерево, каждая вершина которого может иметь левого и правого сына. Один или оба сына могут отсутствовать. Листом называется вершина, у которой нет детей. Множество вершин, на пути от которых до корня лежит вершина  $u$ , называется поддеревом вершины  $u$ . Назовем высотой вершины максимальное количество вершин на пути от этой вершины до листа в ее поддереве.

Чтобы подвешенное дерево было АВЛ-деревом, должно выполняться следующее свойство: для каждой вершины высота поддерева её левого сына и её правого сына должны отличаться не более чем на один. Высоту пустого дерева будем считать равной нулю.

На рисунке показано АВЛ-дерево, содержащее 7 вершин. Корень дерева имеет высоту 4, поддерево его левого сына имеет высоту 2, поддерево его правого сына имеет высоту 3. Легко убедиться, что условие, что высоты поддеревьев различаются не более чем на один, выполняется для всех вершин.



Задано число  $n$ . Определите максимальную высоту АВЛ-дерева, которое содержит ровно  $n$  вершин.


## Формат ввода


На вход подается одно целое число  $n$  ( $1 \leq n \leq 10^{18}$ ).

## Формат вывода

Выведите одно целое число  $h$  — максимальную высоту АВЛ-дерева, которое содержит ровно  $n$  вершин.

## Пример

Ввод 

Вывод 

# 8. (11)

Ограничение времени	1 секунда
Ограничение памяти	512 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

Дано целое положительное число  $n$ . Выпишем в десятичной системе счисления все числа от 1 до  $n$ , включительно. Для каждой цифры от 0 до 9 посчитайте, сколько раз она суммарно встречается в записи этих чисел.


## Формат ввода

На вход подается одно целое число  $n$  ( $1 \leq n \leq 10^{18}$ ).

## Формат вывода

Выведите 10 целых чисел: для каждой цифры от 0 до 9 выведите суммарное количество вхождений этой цифры в десятичные записи чисел от 1 до  $n$ , включительно.

## Пример

Ввод		Вывод	
17		1 10 2 2 2 2 2 2 1 1	



## 9. (8)

Ограничение времени	2 секунды
Ограничение памяти	512 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

Вы — опытный разработчик в департаменте ИТ блока «Сеть продаж» Сбера. Ваш департамент отвечает за ПО в банкоматах и офисах Сбера, а также добавляет в сервисы функционал ИИ.

С самого открытия этого направления вашей мечтой было работать на нём с лучшими умами Сбера и получить задание по обучению ИИ. В этой задаче вы получили шанс реализовать мечту, правда необычным образом, вам предстоит изучить предложенный ИИ алгоритм шифрования.

ИИ предложил следующий алгоритм шифрования пароля. Пароль — строка  $s$  из строчных английских букв длины  $n$ . Запишем символы строки в массив  $a$  в виде их числовых ASCII кодов: «a» соответствует число 97, «b» — число 98, «c» — число 99 и так далее.

Далее над этим числовым массивом выполняется операция «xor» — побитовое исключающее или — для соседних чисел и получается массив  $x$ , где  $x[i] = a[i] \wedge a[i + 1]$  для  $1 \leq i < n$ ,  $x[n] = a[n] \wedge a[1]$ .

Вам на вход дан массив  $x$ . Требуется восстановить пароль — строку  $s$ . Если подходящих строк  $s$  несколько, паролем является лексикографически минимальная подходящая строка  $s$ . Гарантируется, что хотя бы одна строка подходит.

Напомним, что строка  $s$  лексикографически меньше строки  $t$  такой же длины, если они имеют (возможно пустой) общий префикс, а следующий символ в строке  $s$  меньше следующего символа строки  $t$ .


### Формат ввода

В первой строке задано число  $n$  ( $3 \leq n \leq 100\,000$ ). Во второй строке находятся элементы массива  $x$  ( $0 \leq x[i] \leq 255$ ).


### Формат вывода

Выведите расшифрованный пароль — минимальную лексикографически строку  $s$ , состоящую из строчных английских букв, для которой при шифровании описанным в условии методом получается заданный во входе массив  $x$ . Гарантируется, что хотя бы одна подходящая строка существует.

#### Пример 1


Ввод 

7  
3 3 2 2 3 3 0


Вывод 

abacaba

#### Пример 2

Ввод 

10  
13 9 0 3 24 24 29 30 8 12

Вывод 

helloworld

# 10. (15)

Ограничение времени	1 секунда
Ограничение памяти	512 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

Рассмотрим три строки:  $a$ ,  $b$  и  $s$ , состоящие из строчных букв английского алфавита. Будем обозначать как  $s[l : r]$  подстроку строки  $s$ , состоящую из символов с  $l$ -го включительно по  $r$ -й не включительно при нумерации с нуля (подобное обозначение используется, например, в языке программирования Python). Например, если  $s = \text{"abacaba"}$ , то  $s[2 : 5] = \text{"aca"}$ .

Будем называть две подстроки  $x = s[l_1 : r_1]$  и  $y = s[l_2 : r_2]$  строки  $s$  *интересной парой*, если для них выполняются следующие условия:

- $x$  является префиксом строки  $a$ ;
- $y$  является префиксом строки  $b$ ;
- подстроки  $s[l_1 : r_1]$  и  $s[l_2 : r_2]$  имеют хотя бы один общий символ, иначе говоря, найдется такой индекс  $i$ , что  $l_1 \leq i < r_1$  и  $l_2 \leq i < r_2$ .

Весом интересной пары будем называть сумму длин строк  $x$  и  $y$ , то есть  $r_1 - l_1 + r_2 - l_2$ .

Необходимо найти интересную пару подстрок  $s$ , которая имеет наибольший вес, и вывести этот вес. Если интересных пар подстрок в  $s$  нет, будем считать, что ответ равен 0.

## Формат ввода


Ввод состоит из трех строк, содержащих  $a$ ,  $b$  и  $s$ , соответственно.

Все строки во вводе состоят из строчных букв английского алфавита. Длина каждой строки лежит в диапазоне от 1 до 200 000.


## Формат вывода

Выведите одно целое число: максимальный вес интересной пары. Если интересных пар нет, необходимо вывести число 0.

## Пример

Ввод 

```
aaabbb
ababab
aaaababba
```

Вывод 

8