

А. Совпадение? Не думаю

	Все языки	GNU C++20 10.2
Ограничение времени	5 секунд	1 секунда
Ограничение памяти	512Mb	512Mb
Ввод	стандартный ввод или input.txt	
Вывод	стандартный вывод или output.txt	

А. Совпадение? Не думаю

В. Фигуры высшего пилотажа

С. Корпоративные закупки

Д. План эвакуации

Е. Облачные вычисления

Алиса Селезнева была невероятно счастлива: она наконец запустила свой новый стартап по распознаванию увиденных облаков, который назвала строкой A длины N . Но вдруг она узнала, что Зелибоба также запустил свой стартап по распознаванию облаков и назвал его строкой B длины N .

Алиса уверена, что Зелибоба спланировал её идею! Для начала она хочет вычислить метрику похожести названий их стартапов — если название будет похоже, то ей будет сильно проще в дальнейших доказательствах и разбирательствах.

Более формально, пусть есть строки A — название стартапа Алисы и строка B — название стартапа Зелибобы. Обе строки имеют одинаковую длину N . Для каждой позиции $1 \leq i \leq N$ строки B , нужно вычислить тип совпадения в этой позиции со строкой A .

Если $B_i = A_i$, то в позиции i тип совпадения должен быть равен P (от слова *plagiarism*).

Если $B_i \neq A_i$, но существует другая позиция $1 \leq j \leq N$, такая что $B_i = A_j$, то в позиции i тип совпадения должен быть равен S (от слова *suspicious*).

Обратите внимание:

- Буквы в рамках одной строки могут повторяться.
- Каждую букву строки A можно использовать не более чем в одном совпадении типа *plagiarism* или *suspicious*.
- Предпочтение всегда отдается типу *plagiarism*.
- В случае совпадения типа *suspicious*, предпочтение всегда отдается самой левой позиции в строке A .

В остальных позициях тип совпадения должен быть равен I (от слова *innocent*).

Формат ввода

В первой строке задана строка A ($1 \leq |A| \leq 10^6$). Вторая строка — строка B ($1 \leq |B| \leq 10^6$).

Формат ввода

В первой строке задана строка A ($1 \leq |A| \leq 10^6$) — загаданное слово.
Во второй строке задана строка B ($|B| = |A|$) — попытка игрока.
Гарантируется, что строки A и B содержат только заглавные латинские буквы.

Формат вывода

Выведите единственную строку $C(|C| = |B|)$, где C_i — тип совпадения символа B_i ($1 \leq i \leq |B|$):

- для типа plagiarism $C_i = P$.
- для типа suspicious $C_i = S$.
- для типа innocent $C_i = I$.

Пример 1

Ввод 

CLOUD
CUPID

Вывод 

PSIIP

Пример 2

Ввод 

ALICE
ELIBO


Вывод 

SPPII

Пример 3

Ввод 

ABCBCYA
ZBBACAA

Вывод 

IPSSPIP

I Примечания

Пояснение к первому тесту

- $B_1 = A_1$ и $B_5 = A_5$, поэтому для позиций 1 и 5 ответ Р.
- $B_2 \neq A_2$, но $B_2 = A_4$, поэтому для позиции 2 ответ S.
- Буквы Р и I не встречаются в строке A, поэтому для позиций 3 и 4 ответ I.

Пояснение ко второму тесту

- $B_2 = A_2$ и $B_3 = A_3$, поэтому для позиций 2 и 3 ответ Р.
- $B_1 \neq A_1$, но $B_1 = A_5$, поэтому для позиции 1 ответ S.
- Буквы В и О не встречаются в строке A, поэтому для позиций 4 и 5 ответ I.

Пояснение к третьему тесту

- $B_2 = A_2$, $B_5 = A_5$ и $B_7 = A_7$, поэтому для позиций 2, 5 и 7 ответ Р.
- $B_3 \neq A_3$, но $B_3 = A_2 = A_4$. A_2 уже задействовано в соответствии $B_2 = A_2$, поэтому выбирается соответствие $B_3 = A_4$ — для позиции 3 ответ S.
- $B_4 \neq A_4$ и $B_6 \neq A_6$, но $B_4 = B_6 = A_1 = A_7$.
 - A_7 уже задействовано в соответствии $B_7 = A_7$;
 - $4 < 6$, поэтому для позиции 4 выбирается соответствие $B_4 = A_1$ (ответ S);
 - для позиции 6 соответствий не осталось (ответ I).
- Буква Z не встречается в строке A, поэтому для позиции 1 ответ I.

В. Фигуры высшего пилотажа

	Все языки	GNU C++20 10.2
Ограничение времени	4 секунды	1 секунда
Ограничение памяти	512Mb	512Mb
Ввод	стандартный ввод или input.txt	
Вывод	стандартный вывод или output.txt	

Алиса Селезнева и Зелибоба помирились и создали объединенный стартап по распознаванию увиденных облаков. В качестве рекламы они решили провести соревнование по витанию в облаках.

Соревнование проходит в два этапа:

- в отборочном этапе все участники показывают владение заданными приёмами облачной акробатики.
- в финальном этапе прошедшие участники выступают в выбранной ими специальной дисциплине.

В первом этапе участникам было предложено 12 различных приёмов для показа. В первую очередь приоритет отдавался участникам, исполнившим большее число приёмов.

При равенстве количества выполненных приёмов участники сравниваются по штрафу — чем он ниже, тем выше приоритет у кандидата. Штраф определяется командой судей по особой формуле, что гарантирует отсутствие двух кандидатов с одинаковой парой количества исполненных приёмов и штрафа.

Для каждой специальной дисциплины финального этапа определено максимальное количество участников — больше звать нельзя, иначе зрителям станет скучно смотреть.

На вас возложена очень важная задача — по информации о результатах отборочного этапа и специальных дисциплинах финального этапа вывести всех участников, прошедших в финальный этап.

Формат ввода

В первой строке содержится целое число n ($1 \leq n \leq 10^4$) — число специальных дисциплин финального этапа.

Следующие n строк имеют вид s_i, m_i ($1 \leq |s_i| \leq 30$, $1 \leq m_i \leq 10^4$) — название и максимальное число участников на i -ю дисциплину. Название дисциплины содержит только строчные латинские буквы и знак подчеркивания '_'.

Далее следует строка, содержащая целое число k ($1 \leq k \leq 10^5$) — число участников отборочного этапа соревнования.

Последующие k строк имеют вид c_j, q_j, r_j, p_j ($1 \leq |c_j| \leq 30$, $0 \leq r_j \leq 12$, $0 \leq p_j \leq 10^9$) — строковый идентификатор j -го участника, название интересующей его специальной дисциплины, количество исполненных участником приёмов и начисленный ему штраф соответственно. Идентификатор участника содержит только строчные латинские буквы и знак подчеркивания '_'.

Гарантируется, что:

- Интересующие участников специальные дисциплины q_j обязательно представлены во входных данных;
- Названия всех специальных дисциплин s_i попарно различны между собой;
- Идентификаторы участников c_j попарно различны между собой;
- Не существует двух участников a и b таких, что $r_a = r_b$ и $p_a = p_b$ (или различаются количества исполненных приёмов, или различается начисленный участникам штраф).

Все строки используют разделитель ',' (запятая).

Формат вывода

Выведите список идентификаторов всех участников, прошедших в финальный этап соревнования на интересующие их специальные дисциплины в лексикографическом (алфавитном) порядке идентификаторов.

Участник считается прошедшим по дисциплине i , если существует строго менее m_i участников, подавших заявки на i -ю дисциплину с большим, чем у данного участника, приоритетом (см. условие про сравнение приоритета участников).

Пример 1


Ввод 

```
2
ear_flying,1
sun_bathing,1
3
cheburashka,ear_flying,11,100
dambo,ear_flying,10,0
crocodile_gena,sun_bathing,11,10
```


Вывод 

```
cheburashka
crocodile_gena
```

Пример 2

Ввод 

```
1
ear_flying,1
2
dambo,ear_flying,7,234
cheburashka,ear_flying,7,123
```


Вывод 

```
cheburashka
```

Пример 3

Ввод 

```
2
cloud_developer,2
cloud_hacker,3
5
anonymous,cloud_hacker,6,0
bjarne_stroustrup,cloud_developer,1
julian_assange,cloud_hacker,5,10050
bill_gates,cloud_developer,3,1
guccifer,cloud_hacker,2,0
```

Вывод 

```
anonymous
bill_gates
bjarne_stroustrup
guccifer
julian_assange
```

Пример 4

Ввод



Вывод



2	cactus
sun_charging,2	helios
racing,1	palm
5	
helios,racing,12,135	
acacia,sun_charging,0,5	
cactus,sun_charging,0,1	
figus,sun_charging,0,4	
palm,sun_charging,0,3	

Примечания

В первом тестовом примере на дисциплину «eag_flying» претендуют два кандидата — 'cheburashka' показал 11 приёмов со 100 единицами штрафа, а 'dambo' — 10 приёмов с 0 штрафа. В первую очередь сравнение идет по приёмам: $11 > 10$, поэтому в следующий этап проходит именно 'cheburashka'.

Во втором тестовом примере у обоих претендентов одинаковое количество показанных приёмов, но у 'cheburashka' штраф 123, что меньше штрафа 234 у 'dambo'. Поэтому в следующий этап проходит 'cheburashka'.

В третьем тестовом примере важно отметить, что в ответе прошедшие участники идут в объединенном списке отсортированные в лексикографическом порядке независимо от своего приоритета при отборе и выбранной ими специальной дисциплине.

В четвертом тестовом примере развернулась борьба за дисциплину «зарядка на солнце». Хотя все участники не смогли показать владение ни одним из приёмов, 'cactus' и 'palm' набрали меньше всего штрафа, поэтому именно их приглашают в финальный этап соревнования.

С. Корпоративные закупки



	Все языки	GNU C++20 10.2
Ограничение времени	2 секунды	1 секунда
Ограничение памяти	512Mb	512Mb
Ввод	стандартный ввод или input.txt	
Вывод	стандартный вывод или output.txt	

Стартап Алисы Селезневой и Зелибобы привлек к себе внимание крупных инвесторов. Часть полученных от инвесторов денег было решено потратить на обновление офиса — новая мебель, оргтехника и другие прикольные штуки.

Алиса и Зелибоба выдвинули 5 критериев — товар должен удовлетворять всем данным критериям, чтобы его закупили в обновленный офис.

- «Наименование товара содержит подстроку **в любом регистре**» (критерий 'NAME_CONTAINS');
- «Цена **больше либо равна** чем» (критерий 'PRICE_GREATER_THAN');
- «Цена **меньше либо равна** чем» (критерий 'PRICE_LESS_THAN');
- «Товар поступил в продажу **не позднее** чем» (критерий 'DATE_BEFORE');
- «Товар поступил в продажу **не ранее** чем» (критерий 'DATE_AFTER');

Закупаться было решено в Выньдекс.Рынке. Для такого крупного клиента Выньдекс.Рынок предоставил стартапу персонального менеджера — да-да, именно вас.

Первым делом вам необходимо из имеющегося списка товаров на складе выбрать все товары, удовлетворяющие выданным Алисой и Зелибобой критериям.

Формат ввода

Общее описание формата входных данных:

Первая строка входных данных содержит список всех имеющихся на складе Выньдекс.Рынка товаров в формате JSON.

Следующие 5 строк имеют вид $q_i v_i$ — фильтр и соответствующее ему актуальное значение.

Подробное описание формата списка товаров

Гарантии по формату JSON:

Формат ввода

Общее описание формата входных данных:

Первая строка входных данных содержит список всех имеющихся на складе Выньдекс.Рынка товаров в формате JSON.

Следующие 5 строк имеют вид $q_i v_i$ — фильтр и соответствующее ему актуальное значение.

Подробное описание формата списка товаров

Гарантии по формату JSON:

- нет запятых после последнего элемента массива;
- все имена полей и строки обернуты в двойные кавычки.

Обозначим количество товаров в списке через N . Гарантируется, что $0 \leq N \leq 1000$.

Каждый товар в списке содержит следующую информацию (порядок полей не является фиксированным):

- целое число id ($0 \leq id \leq 2^{31} - 1$) — уникальный идентификатор. Гарантируется, что идентификаторы всех товаров попарно различны;
- строка $name$ ($1 \leq |name| \leq 100$) — наименование. Гарантируется, что наименование содержит только строчные и заглавные латинские буквы, а так же пробел;
- целое число $price$ ($0 \leq price \leq 2^{31} - 1$) — цена;
- строка $date$ в формате «dd.MM.yyyy» ($01.01.1970 \leq date \leq 31.12.2070$) — дата поступления в продажу.

Подробное описание формата фильтров

Гарантируется, что:

- все q_i различны между собой;
- q_i является строкой из множества (NAME_CONTAINS, PRICE_GREATER_THAN, PRICE_LESS_THAN, DATE_BEFORE, DATE_AFTER);
- в фильтре 'NAME_CONTAINS' v_i представляет из себя строку ($1 \leq |v_i| \leq 100$), содержащую только строчные и заглавные латинские буквы;
- в фильтрах 'PRICE_GREATER_THAN' и 'PRICE_LESS_THAN' v_i представляет из себя целое число ($0 \leq v_i \leq 2^{31} - 1$);
- в фильтрах 'DATE_BEFORE' и 'DATE_AFTER' v_i представляет из себя строку в формате «dd.MM.yyyy» ($01.01.1970 \leq v_i \leq 31.12.2070$).

Формат вывода

Формат вывода

Выведите в формате JSON список товаров, удовлетворяющих всем указанным во входных данных критериям. Каждый товар должен быть выведен ровно один раз в отсортированном по возрастанию *id* порядке.

Выводить JSON допустимо как с дополнительными отступами и переводами строк, так и в одну строку.

Имена полей необходимо выводить в двойных кавычках.

Допустимо выводить запятую после последнего поля объекта или последнего элемента массива.

Каждый товар должен содержать информацию, аналогичную информации из входных данных:

- целое число *id* — уникальный идентификатор;
- строка *name* — наименование;
- целое число *price* — цена;
- строка *date* в формате «dd.MM.yyyy» — дата поступления в продажу.

Пример

Ввод



Вывод



```
[{"id": 1, "name": "Asus notebook", "price": 2300, "date": "2021-09-23"}, {"id": 2, "name": "EaRPoDs", "price": 2400, "date": "2022-01-02"}]
```

Примечания

При написании решения на Java можно выбрать комплятор «Java 8 + json-simple». В этом случае вы сможете воспользоваться библиотекой [json-simple](#) для парсинга и сериализации JSON.

При написании решения на C++ можно подключить `#include "json.hpp"` для использования библиотеки [json](#) для парсинга и сериализации JSON.

Рассмотрим тестовый пример.

В нем представлено 5 товаров:

- `"id": 1, "name": "Asus notebook"price": 1564, "date": "23.09.2021"`
- `"id": 2, "name": "EaRPoDs"price": 2200, "date": "01.01.2022"`
- `"id": 3, "name": "Keyboardpods"price": 2500, "date": "05.06.2020"`
- `"id": 4, "name": "Dell notebook"price": 2300, "date": "23.09.2021"`
- `"id": 5, "name": "Airpods"price": 2300, "date": "23.09.2021"`

и следующие критерии:

- название включает подстроку `pods` в любом регистре;
- цена находится в промежутке $2200 \leq price \leq 2400$;
- дата поступления в продажу находится в промежутке $23.09.2021 \leq date \leq 02.01.2022$.

Только товары с идентификаторами 2 и 5 удовлетворяют всем критериям:

- Товар с идентификатором 1 не удовлетворяет критериям имени (нет заданной подстроки) и цены (слишком низкая);
- Товар с идентификатором 3 не удовлетворяет критериям цены (слишком высокая) и даты (слишком ранняя);
- Товар с идентификатором 4 не удовлетворяет только критериям имени (нет заданной подстроки).

Обратите внимание, что выводить необходимо товары в порядке возрастания идентификаторов (заметьте, что во входных данных товар с идентификатором 5 стоит раньше товара с идентификатором 2).

D. План эвакуации

	Все языки	GNU C++20 10.2
Ограничение времени	2 секунды	1 секунда
Ограничение памяти	512Mb	512Mb
Ввод	стандартный ввод или input.txt	
Вывод	стандартный вывод или output.txt	

Стартап Зелибобы и Алисы Селезневой расширился настолько, что пришло время переехать в новый офис. Первым делом в новом офисе было решено повесить план эвакуации. Офис занимает целый этаж прямоугольного здания площадью $N \times M$ метров. Алиса распечатала схему этажа в виде $N \times M$ клеток (каждая клетка задаёт пространство площадью 1×1 метров), где «#» обозначает кусок мебели или стены, а «.» — пространство, доступное для перемещения сотрудников.

Также на карте ровно одна клетка обозначена как «S» — участок, на котором находится эвакуационный выход с этажа.

Гарантируется, что планировка офиса удовлетворяет следующим условиям:

- Все клетки в первых и последних строках / столбцах схемы являются стенами.
- От любой пустой клетки можно добраться до эвакуационного выхода, перемещаясь только вверх / вниз / влево / вправо.
- Между любой парой пустых клеток на схеме существует ровно один путь, возможно проходящий через эвакуационный выход.

Зелибоба просит вас для каждого участка на заданной схеме отобразить направление движения к эвакуационному выходу. Гарантируется, что такое направление определяется однозначно.

Помогите Зелибобе и выведите для каждого участка направление, в котором сотрудник должен проследовать в направлении к эвакуационному выходу.

Формат ввода

В первой строке даны два целых числа N и M ($3 \leq N, M \leq 500$) — количество строк и столбцов на схеме этажа.

В следующих N строках расположено по M символов из множества $(\#, ., S)$.

Гарантируется, что

- Все клетки в первых и последних строках / столбцах схемы равны $\#$.
- На схеме расположена ровно одна клетка S .
- От любой пустой клетки можно добраться до клетки S , перемещаясь только вверх / вниз / влево / вправо.
- Между любой парой пустых клеток на схеме существует ровно один путь, возможно проходящий через клетку S .


Формат вывода

Выведите N строк по M символов в каждой — схему этажа, где каждая пустая клетка $.$ заменена на направление в сторону эвакуационного выхода.

Занумеруем все строки от 1 до N сверху вниз, все столбцы — от 1 до M слева направо. В таком случае пустая клетка (r, c) должна содержать:

- R — если из клетки (r, c) необходимо проследовать в клетку $(r, c + 1)$;
- L — если из клетки (r, c) необходимо проследовать в клетку $(r, c - 1)$;
- D — если из клетки (r, c) необходимо проследовать в клетку $(r + 1, c)$;
- U — если из клетки (r, c) необходимо проследовать в клетку $(r - 1, c)$.

Пример 1


Ввод 

```
5 8
#####
#.....#
#.#S#.#
##...###
#####
```


Вывод 

```
#####
#RRDLLL#
#U#S#U##
##RUL###
#####
```

Пример 2

Ввод 

```
3 3
###
#S#
###
```

Вывод 

```
###
#S#
###
```

Примечания

Рассмотрим первый тестовый пример.

Эвакуационным выходом является клетка (3, 4).

В эвакуационный выход сотрудник может попасть из клетки (2, 4), сделав шаг вниз, и из клетки (4, 4), пройдя вверх.

В клетку (2, 4) можно пройти вправо из клеток (2, 3) и (2, 2); в клетку (2, 2) можно попасть из клетки (3, 2), пройдя вверх.

Также в клетку (2, 4) можно пройти влево из клеток (2, 5), (2, 6) и (2, 7); в клетку (2, 6) можно попасть из клетки (3, 6), пройдя вверх.

В клетку (4, 4) можно попасть всего из двух клетки — пройти направо из (4, 3) и налево из (4, 5).

Во втором тестовом примере сотрудник может находиться только на клетке с выходом.

Е. Облачные вычисления

Язык	Ограничение времени	Ограничение памяти	Ввод	Вывод
Все языки	6 секунд	42Mb	input.txt	стандартный вывод или output.txt
GNU C++20 10.2	4 секунды	16Mb		
C# (MS .Net 6.0)+ASP	4 секунды	16Mb		
Python 3.7 (PyPy 7.3.3)	10 секунд	42Mb		

Для распознавания облаков Зелибоба и Алиса Селезнева пишут программы на специальном языке Cloud++. Его синтаксис содержит лишь следующие элементы:

- строчные английские буквы $a - z$;
- знаки математических операций: $+$, $-$, $*$, $/$, $=$;
- скобки $\{$ и $\}$;
- пробел.

По правилам языка Cloud++ фигурные скобки должны образовывать «правильную скобочную последовательность» — последовательность из « $\{$ » и « $\}$ » такая, что существует хотя бы одно разбиение всех скобок последовательности на пары « $\{$ » и « $\}$ », для которых верно:

- в любой паре порядковый номер « $\{$ » в последовательности меньше, чем у соответствующей ей « $\}$ ».
- каждая скобка относится ровно к одной паре из разбиения.

Например, последовательность « $\{\{\}\{\}\{\}\}$ » правильная, так как существует разбиение (1, 6), (2, 3), (4, 5), (7, 8) (скобки нумеруются с 1 слева направо).

Примеры неправильных скобочных последовательностей:

- « $\{\{\{\}\}$ » неправильная, так как в единственно возможном разбиении (1, 4), (2, 4), (3, 4) скобка 4 используется во всех трёх парах.
- « $\{\}\{\}$ » неправильная, так как в единственно возможном разбиении (2, 1) позиция скобки « $\{$ » больше, чем позиция « $\}$ ».
- « $\{\{\}$ » неправильная, так как невозможно построить ни одной пары без скобок « $\}$ ».

Алиса написала новую программу, но поймала ошибку компиляции. Компилятор утверждает, что последовательность скобок в программе отличается от правильной наличием ровно **одной лишней** скобки (правда не уточняет, какая это скобка и в какой она позиции).

По правилам языка Cloud++ фигурные скобки должны образовывать «правильную скобочную последовательность» — последовательность из «{» и «}» такая, что существует хотя бы одно разбиение всех скобок последовательности на пары «{» и «}», для которых верно:

- в любой паре порядковый номер «{» в последовательности меньше, чем у соответствующей ей «}».
- каждая скобка относится ровно к одной паре из разбиения.

Например, последовательность «{ { } } { }» правильная, так как существует разбиение (1, 6), (2, 3), (4, 5), (7, 8) (скобки нумеруются с 1 слева направо).

Примеры неправильных скобочных последовательностей:

- «{ { { } }» неправильная, так как в единственно возможном разбиении (1, 4), (2, 4), (3, 4) скобка 4 используется во всех трёх парах.
- «{ } {» неправильная, так как в единственно возможном разбиении (2, 1) позиция скобки «{» больше, чем позиция «}».
- «{ { {» неправильная, так как невозможно построить ни одной пары без скобок «}».

Алиса написала новую программу, но поймала ошибку компиляции. Компилятор утверждает, что последовательность скобок в программе отличается от правильной наличием ровно **одной лишней** скобки (правда не уточняет, какая это скобка и в какой она позиции).

Зелибоба не доверяет компиляторам (слишком они подозрительно себя ведут). Поэтому он просит вас, как главного специалиста по ремонту кофемашин, проверить правдивость слов компилятора.

Для заданного текста программы, написанной Алисой, выведите наименьшую позицию символа скобки (открывающей или закрывающей), **удаление** которой сделает последовательность скобок в программе правильной.

Формат ввода

Обратите внимание, что в данной задаче разрешен только ввод из файла «input.txt». В единственной строке файла содержится строка S ($1 \leq |S| \leq 5 \cdot 10^7$) — программа на языке Cloud++. Строка гарантированно завершается символом перевода строки.

Гарантируется, что строка S содержит символы только из указанных ниже:

- строчные английские буквы $a - z$;
- знаки математических операций: $+$, $-$, $*$, $/$, $=$;
- скобки $\{$ и $\}$;
- пробел.

Формат вывода

Если последовательность скобок в программе можно сделать правильной, удалив ровно одну скобку, то выведите наименьшую позицию такой скобки. В ином случае выведите число -1 . Позиции считаются, начиная от 1. Пробелы также считаются символами и учитываются при подсчёте позиции.

Пример 1

Ввод

Вывод

$a + b = b + a$

-1

Пример 2

Ввод

Вывод

$d + \{a + \{b + c\} = \{a + b + \} c + ($ 5

Пример 3

Ввод

Вывод

$\{a\{\{b + c\} = ab + bc$

-1

Примечания

Первый тестовый пример

Последовательность скобок пустая, а значит она уже правильная, ничего удалять не нужно — поэтому ответом является —1.

Второй тестовый пример

Обратите внимание на нумерацию символов в начале строки:

1. d
2. пробел
3. +
4. пробел
5. {
6. a
7. пробел
8. +
9. пробел
10. {
11. a
12. т.д.

Последовательность скобок «{ { } { }» отличается от правильной ровно одной **лишней** скобкой. Чтобы сделать её правильной, необходимо удалить одну из двух открывающих скобок в начале скобочной последовательности (позиции в строке 5 и 10). Так как требуется удалить скобку на наименьшей позиции, то ответом является 5.

Обратите внимание, что всё выражение даже после удаления не должно быть / стать корректным — необходимо лишь восстановить правильность скобочной последовательности.

Третий тестовый пример

Последовательность скобок «{ { { } }» невозможно сделать правильной удалением ровно одной скобки — поэтому ответ —1.