Mizzza54

Яндекс, Осенняя стажировка 2021

1 авг 2021, 16:44:09
старт: 1 авг 2021, 09:53:16
финиш: 1 авг 2021, 15:53:16
длительность: 06:00:00

A. Медиана с вычитанием

Ограничение времени	1 секунда
Ограничение памяти	512Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Дан массив a длины 3 из целых чисел.
Определим операцию изменения массива: выбирается два различных индекса i и j ($1 \le i, j \le 3$, $i \ne j$), после чего $a[i]$ становится равным $a[i] - a[j]$.

Пример операции: дан массив $[1, -3, 2]$, выбрали $i = 2, j = 1$, получили массив $[1, -3 - 1, 2]$ = $[1, -4, 2]$.

Определим для массива a медиану m как значение, расположенное на позиции 2 при сортировке элементов массива a.

К примеру, медианой массива a = $[1, -3, 2]$ является m = 1, так как в сортированном массиве $[-3, 1, 2]$ именно 1 стоит на позиции 2.

Назовём медианным индексом такой индекс i, что a_i = m.

Обратите внимание, что медианный индекс необязательно единственный: в массиве a = $[3, 0, 3]$ медиана m = 3, а медианными индексами являются i_1 = 1 (a_1 = m) и i_2 = 3 (a_3 = m).

Для каждого индекса i массива a выясните, может ли он стать медианным, если можно сделать не более одной операции изменения массива (можно не делать операций вовсе).

Формат ввода

В единственной строке даны 3 целых числа a_i ($-10^9 \le a_i \le 10^9$), разделенные пробелами.

Формат вывода

Для каждого индекса i ($1 \le i \le 3$) выведите в отдельной строке ответ: `YES`, если после не более, чем одной операции изменения массива i может стать медианным индексом; `NO` — иначе.

Пример 1

Ввод	Вывод
2 6 5	YES YES YES

Пример 2

Ввод	Вывод
0 -3 1	YES NO YES

Примечания

В первом тесте a = $[2, 6, 5]$.
Если сделать операцию изменения $i = 2, j = 3$, то получится массив $[2, 1, 5]$, медиана будет равна 2, а значит i = 1 будет являться медианным индексом.

Если сделать операцию изменения $i = 2, j = 1$, то получится массив $[2, 4, 5]$, медиана будет равна 4, а значит i = 2 будет являться медианным индексом.

Если не делать никаких операций изменения, то медианой массива $[2, 6, 5]$ будет 5, а значит i = 3 будет являться медианным индексом. Аналогично $i = 3$ будет медианным индексом после операции изменения $i = 3, j = 1$.

Во втором тесте единственной операцией изменения, делающей индекс $i = 2$ медианным, является операция $i = 2, j = 2$, но такая операция не является корректной, так как индексы i и j должны быть различны.

Язык Oracle Java 8

Набрать здесь | Отправить файл

```
1
```

Отправить

Следующая

Mizzza54

Яндекс, Осенняя стажировка 2021

1 авг 2021, 16:46:28
старт: 1 авг 2021, 09:53:16
финиш: 1 авг 2021, 15:53:16
длительность: 06:00:00

B. Яндекс.Бар

Ограничение времени	1 секунда
Ограничение памяти	512Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Яндекс запускает новый бизнес-юнит – Яндекс.Бар. При разработке меню появилась необходимость рисовать слоистые коктейли. Для поддержания концепции IT-бара было принято решение рисовать коктейли, используя ASCII-арт. Вам необходимо считать форму стакана и список ингредиентов слоистого коктейля, заполнить стакан ингредиентами в нужном порядке и распечатать результат. Описание стакана – это символьное поле из n строк по m символов в каждой. Поле состоит из символов `.` (точек), `\` (обратных слешей), `/` (прямых слешей), `|` (вертикальных черт), `_` (нижних подчеркиваний) и пробелов.

Дно стакана расположено на последней n-й строке и состоит из символов `_` (нижних подчеркиваний), расположенных слитно.

Слева и справа от дна нарисованы грани стакана. Грань – это «ломаная» из символов `\` (обратных слешей), `/` (прямых слешей), `|` (вертикальных черт). Каждая грань состоит из ровно n символов, начинается в последней n-й строке и заканчивается в первой. Для любой пары соседних строк i и $i+1$ символы грани расположены в одном или соседних столбцах. Грани не пересекаются и не касаются друг друга ни по стороне, ни по углу. В частности из этого следует, что в любой строке, кроме последней n-й, есть пустое место, обозначаемое пробелами.

Фон изображения стакана состоит из символов `.` (точек), расположенных слева от левой грани стакана и справа от правой.

Формат ввода

Первая строка содержит два числа n и m ($2 \le n \le 100$, $3 \le m \le 100$), которые обозначают размеры поля – изображения стакана. Следующие n строк по m символов в каждой содержат описание стакана в формате, указанном выше.

Следующая строка содержит число k ($0 \le k \le min\,(n-1,\quad 89)$) – количество ингредиентов коктейля.

Следующие k строк содержат описания ингредиентов, по одному в каждой строке. Описание имеет вид $name_i \quad count_i \quad symbol_i$.

$name_i$ – это название i-го ингредиента (строка из строчных латинских букв длиной не менее 1 и не более 10).

$count_i$ – это количество слоев i-го ингредиента.

$symbol_i$ – это символ, которым i-й ингредиент должен быть представлен в изображении (любой символ с ASCII кодом больше 32 и меньше 127, кроме тех, которые используются в описании изображения стакана).

Гарантируется, что сумма всех $count_i$ не превосходит высоты стакана, то есть $n-1$. Также гарантируется что все символы $symbol_i$ уникальны.

Формат вывода

В n строках по m символов в каждой выведите описание стакана в указанном выше формате.

Пример 1

Ввод **Вывод**

Ввод

```
8 15
\             /
.|           |.
.|           |.
..\         /..
...|       |...
...|       |...
....\     /....
.....\___/.....
2
gin 2 %
tonic 4 *
```

Вывод

```
\             /
.|***********|.
.|***********|.
..\*********/..
...|*******|...
...|%%%%%%%|...
....\%%%%%/....
.....\___/.....
```

Пример 2

Ввод

```
10 23
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
1
acid 5 X
```

Вывод

```
.........\   /.........
.........\   /.........
.........\   /.........
.........\   /.........
......./xxxxxxxxx\......
......./xxxxxxxxxxx\.....
..../xxxxxxxxxxxxx\....
.../xxxxxxxxxxxxxxx\...
../xxxxxxxxxxxxxxxxx\..
./_____________________\.
```

Пример 3

Ввод

```
16 28
...|                    |...
...|                    |...
...|                    |...
...|                    |...
...|                    |...
...|                    |...
...|                    |...
....|                  |....
....|                  |....
....|                  |....
....|                  |....
....|                  |....
....|                  |....
.....|                |.....
.....|                |.....
.....|                |.....
.....|                |.....
......|______________|......
4
kahlua 4 k
baileys 5 b
cointreau 3 c
fire 1 !
```

Вывод

```
...|                    |...
...|                    |...
...|!!!!!!!!!!!!!!!!!!!!|...
...|cccccccccccccccccc|...
...|cccccccccccccccccc|...
...|cccccccccccccccccc|...
....|bbbbbbbbbbbbbbbbbb|....
....|bbbbbbbbbbbbbbbbbb|....
....|bbbbbbbbbbbbbbbbbb|....
....|bbbbbbbbbbbbbbbbbb|....
....|bbbbbbbbbbbbbbbbbb|....
.....|kkkkkkkkkkkkkkkk|.....
.....|kkkkkkkkkkkkkkkk|.....
.....|kkkkkkkkkkkkkkkk|.....
.....|kkkkkkkkkkkkkkkk|.....
......|______________|......
```

Пример 4

Ввод

```
10 16
...|     \......
...|      /.....
../        /....
...\        /...
....|        \..
...../       |...
......\      /...
.......|    |....
....../      \...
.......\_____\..
4
some 1 (
thing 2 ?
really 3 )
strange 1 ^
```

Вывод

```
...|     \......
...|      /.....
../^^^^^^^^/....
...\)))))))))/...
....|)))))))))\..
...../)))))))|...
......\?????/...
.......|???|....
....../(((((\...
.......\_____\..
```

Примечания

В первом примере из условия ингредиент gin наливается в седьмую и шестую строки изображения стакана, а ингредиент tonic в пятую, четвертую, третью и вторую.

Язык Oracle Java 8

Набрать здесь | Отправить файл

```java
import java.util.Scanner;

/**
 * @author Michael Gerasimov
 */
public class YandexBar {
    Scanner scanner = new Scanner(System.in);
    int n, m, k;
    char[][] description;
    Cocktail[] composition;

    private void run() {
        n = scanner.nextInt();
        m = scanner.nextInt();
        scanner.nextLine();
        description = new char[n][m];
        for (int i = 0; i < n; i++) {
            String string = scanner.nextLine();
            description[i] = string.toCharArray().clone();
        }

        k = scanner.nextInt();
        scanner.nextLine();
        composition = new Cocktail[k];
        for (int i = 0; i < k; i++) {
            String string = scanner.nextLine();
            String[] splitted = string.split(" ");
            composition[i] = new Cocktail(splitted[0], Integer.parseInt(splitted[1]), splitted[2].charAt(0));
        }

        for (int i = n - 2, layer = 0, count = 0; i > -1; i--) {

            if (count == composition[layer].count) {
                layer++;
                count = 0;
                if (layer == k) {
                    break;
                }
```

Отправить

Предыдущая | Следующая

Mizzza54

Яндекс, Осенняя стажировка 2021

1 авг 2021, 16:46:44
старт: 1 авг 2021, 09:53:16
финиш: 1 авг 2021, 15:53:16
длительность: 06:00:00

C. Защитники башни

Ограничение времени	1 секунда
Ограничение памяти	512Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

В не очень далёкой галактике на планете Икс-Бомбикс тоже проводится чемпионат мира по футболу. На этой планете солнечные лучи в течение всего дня падают ровно вертикально вниз.
Возле главного стадиона расположена сторожевая башня, состоящая из n плит, расположенных друг над другом так, что их левые края прикреплены к общей колонне. Длина i-й плиты равна a_i.

Занимать пост на сторожевой башне стадиона вызвались m добровольцев - охранников. j-й доброволец имеет ширину плеч b_j, а высота любого из них меньше, чем расстояние между соседними плитами.

С башни открывается отличный вид на стадион, поэтому очень много добровольцев хотят на неё попасть. В то же время из-за техники безопасности при распределении охранников по плитам башни должны выполняться следующие условия:

1. Охранник будет стоять на плите боком, поэтому ширина плеч охранника не должна превышать длины плиты.

2. Охранник должен быть расположен на плите полностью - по краям плиты стоят защитные ограждения (чтобы с неё нельзя было упасть).

3. Охранник должен полностью находиться под солнечными лучами (если он будет в тени хотя бы частью тела, то за время матча замерзнет).

4. На одной плите может быть не более одного охранника (два добровольца не поделят место под солнцем).

Изучите **графическое представление** первого теста **ниже в примечании** для лучшего понимания задачи.

Вам необходимо расположить максимальное число добровольцев по плитам с выполнением описанных условий. Скорее, матч начнётся с минуты на минуту, а добровольцы так и не знают, кого возьмут охранять главный стадион галактики и кто сможет насладиться крутым видом на игру!

Формат ввода

Первая строка входных данных содержит два числа n и m ($1 \le n, m \le 2 \times 10^5$) — количество плит, находящееся в башне, и количество добровольцев соответственно.
Вторая строка входных данных содержит n натуральных чисел a_i ($1 \le a_i \le 10^{18}$) — длина i-й плиты в порядке снизу вверх.

Третья строка входных данных содержит m натуральных чисел b_j ($1 \le b_j \le 10^{18}$) — ширина плеч j-го добровольца.

Формат вывода

В единственной строке выведите максимальное число добровольцев-охранников, которых можно расположить на плитах, с учетом описанных условий.

Пример 1

Ввод	Вывод
5 3 7 3 4 2 2 3 2 1	3

Пример 2

Ввод	Вывод

Ввод	Вывод
2 1 2 10 11	0

Пример 3

Ввод	Вывод
5 4 100 98 96 40 30 2 4 60 3	3

Примечания

В первом тесте из условия есть 3 светлых участка:
1. 5-ю (самую верхнюю) плиту солнце освещает полностью, поэтому на ней находится солнечный участок размера 2;

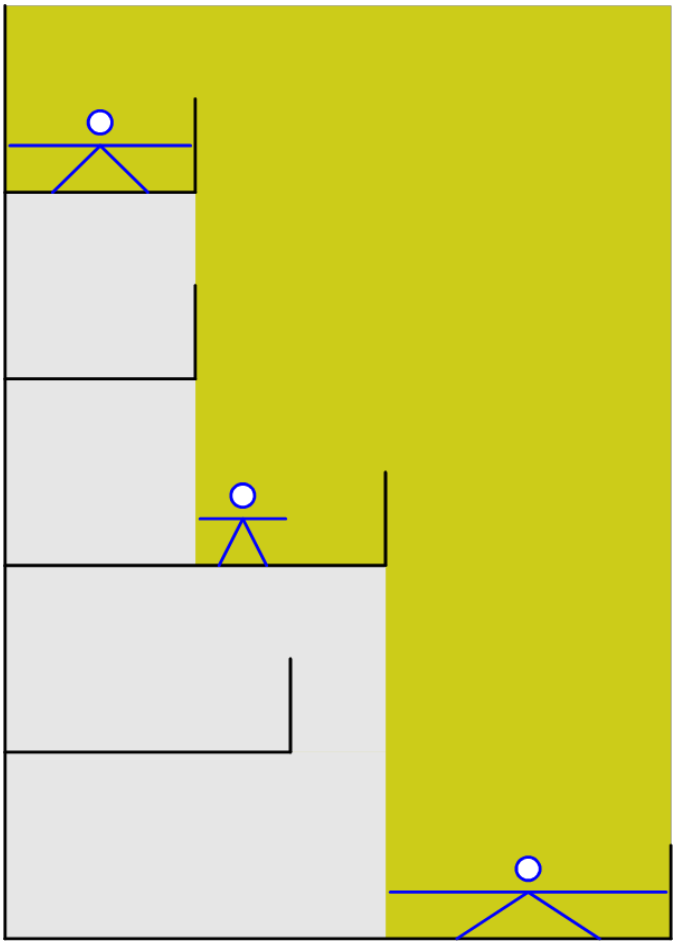
2. 4-я плита - так же размера 2, поэтому она полностью закрыта 5-й.

3. 3-я плита имеет общий размер 4, поэтому солнечный участок на ней имеет размер 2 (над остальной частью плиты нависают плиты 5 и 4);

4. 2-я плита имеет размер 3 и полностью закрыта от солнца плитой 3.

5. 1-я плита имеет общий размер 7, поэтому солнечный участок на ней имеет размер 3 (остальную часть закрывает 3-я плита).

Соответственно, доброволец с шириной плеч 3 займет место на солнечном участке 1-й плиты, а добровольцы с шириной плеч 1 и 2 могут встать на плиты 3 и 5 в любом порядке.



Язык GNU C++20 10.2

Набрать здесь | Отправить файл

```cpp
#include <iostream>
#include <vector>
#include <set>

using namespace std;
typedef unsigned long long ll;

int n, m;
vector<ll> plates;

multiset<ll> people;

multiset<ll> sunnyPlaces;

void foundSunnyPlaces() {
    sunnyPlaces.insert(plates[n - 1]);
    ll maxLength = plates[n - 1];
    for (int i = n - 2; i > -1; i--) {
        if (plates[i] > maxLength) {
            sunnyPlaces.insert(plates[i] - maxLength);
            maxLength = plates[i];
        }
    }
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    cout.tie(nullptr);
    cin >> n >> m;
    plates.resize(n);
    for (int i = 0; i < n; i++) {
        cin >> plates[i];
    }
    for (int i = 0; i < m; i++) {
        ll temp;
        cin >> temp;
        people.insert(temp);
```

Отправить

Предыдущая Следующая

Mizzza54

Яндекс, Осенняя стажировка 2021

1 авг 2021, 16:46:51
старт: 1 авг 2021, 09:53:16
финиш: 1 авг 2021, 15:53:16
длительность: 06:00:00

D. Считаем клетки

Язык	Ограничение времени	Ограничение памяти	Ввод	Вывод
Все языки	1 секунда	512Mb	стандартный ввод или input.txt	стандартный вывод или output.txt
Oracle Java 7	2 секунды	512Mb		
Python 3.9.1	3 секунды	512Mb		
Python 2.7 (PyPy 4.0.0)	3 секунды	512Mb		
Python 3.7 (PyPy 7.3.3)	3 секунды	512Mb		
Python 2.7	3 секунды	512Mb		
Oracle Java 8	2 секунды	512Mb		
Oracle Java 7 x32	2 секунды	512Mb		

На огромной квадратной доске, разделенной на квадратные белые клетки, покрасили некоторые клетки в черный цвет так, что образованная черными клетками фигура удовлетворяет следующим условиям:
1. Из любой черной клетки можно добраться в любую черную, переходя только по другим черным клеткам. Переходить можно только вправо, влево, вверх или вниз, но не по диагонали через угол.

2. Не существует такой черной клетки, перекраска которой в белый цвет приведет к нарушению условия 1.

3. Не существует такой черной клетки, что в двух противоположных направлениях по стороне (сверху и снизу и/или слева и справа) находится белая клетка или граница поля.

4. Из любой белой клетки можно добраться до границы доски, переходя только по белым клеткам. Переходить можно только вправо, влево, вверх или вниз, но не по диагонали через угол.

Назовем контуром фигуры множество черных клеток, у которых хотя бы в одном из 8 направлений (по **стороне** или **углу**) расположена белая клетка или граница поля.

После покраски фигуру выпилили из доски. Причем все распилы были параллельны сторонам доски и проходили только через середины клеток контура. После этого во все углы полученной фигуры поставили отметки.

Графическое представление примеров возможных и невозможных фигур и разрезов представлено **ниже в примечании**.

Вам даны координаты клеток с отметками $x_i,\ y_i$ (отметки расположены в центрах соответствующих клеток) в порядке обхода против часовой стрелки. Порядок обхода такой, что фигура остается по левую руку от каждого отрезка $[x_i,\ y_i;\quad x_{i+1},\ y_{i+1}]$. Ваша задача – посчитать, сколько клеток каждого типа содержит фигура. Клетки бывают четырех типов:

- Полные клетки (без распилов).
- Половинные клетки (по которым проходит один прямой распил).
- Внешние углы (клетки с двумя перпендикулярными распилами, у которых четверть площади принадлежит фигуре).
- Внутренние углы (клетки с двумя перпендикулярными распилами, у которых три четверти площади принадлежит фигуре).

Формат ввода

Первая строка входных данных содержит два целых числа n и m ($1 \le n, m \le 500$) — размер поля.
Вторая строка входных данных содержит число k ($1 \le k \le n \cdot m$) — количество отметок .

В следующих k строках содержатся пары чисел x_i, y_i ($1 \le x_i \le n, 1 \le y_i \le m$) — координаты очередной клетки с отметкой в порядке обхода. Гарантируется, что во всех тестах обход фигуры против часовой стрелки, а отметки стоят только в центрах клеток, являющихся углами фигуры.

Формат вывода

В одной строке через пробел выведите четыре числа a, b, c, d, где:

- a – количество полных клеток.
- b – количество половинных клеток.
- c – количество внешних углов.
- d – количество внутренних углов.

Пример 1

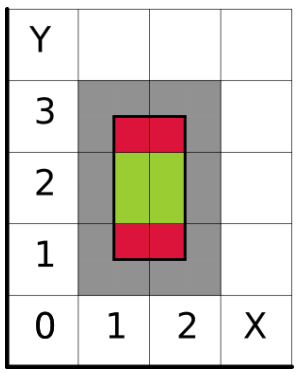
Ввод	Вывод
2 3 4 1 1 2 1 2 3 1 3	0 2 4 0

Пример 2

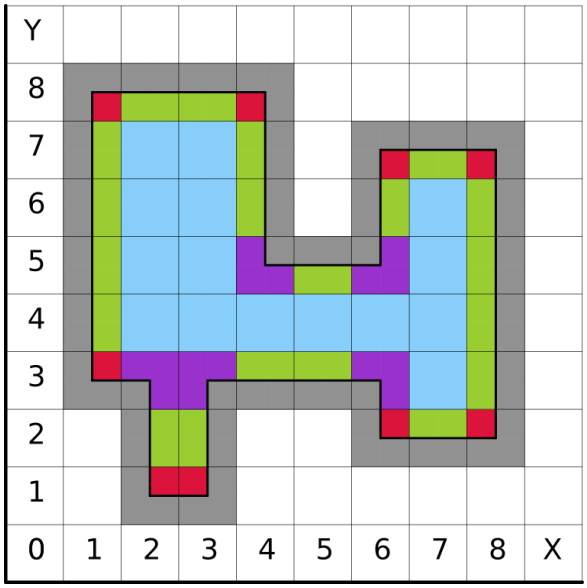
Ввод	Вывод
8 8 14 2 1 3 1 3 3 6 3 6 2 8 2 8 7 6 7 6 5 4 5 4 8 1 8 1 3 2 3	15 20 9 5

Примечания

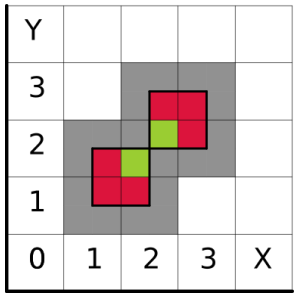
В первом примере из условия фигура выглядит так:



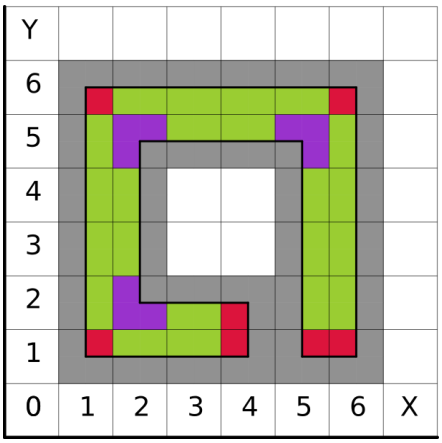
Во втором примере из условия фигура выглядит так:



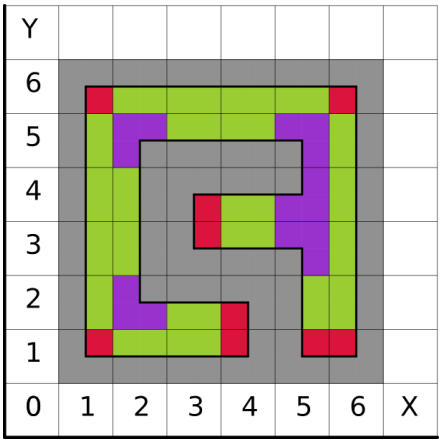
Изображенная далее фигура невозможна в рамках задачи, так как при удалении клетки $(2, 2)$ нарушается связность между клетками фигуры (нарушается условие 2).



Следующая фигура невозможна в рамках задачи, так как клетки $(3, 3)$, $(3, 4)$, $(4, 3)$ и $(4, 4)$ не связаны с границей поля (нарушается условие 4).



Приведенный ниже разрез является некорректным в рамках задачи, так как он проходит не только по контуру фигуры. К примеру, у клетки $(3, 3)$ все 8 соседних клеток так же входят в фигуру, а значит она не принадлежит контуру.



Язык GNU C++20 10.2

Набрать здесь | Отправить файл

```
1
```

Отправить

Предыдущая Следующая

Mizzza54

Яндекс, Осенняя стажировка 2021

1 авг 2021, 16:47:05
старт: 1 авг 2021, 09:53:16
финиш: 1 авг 2021, 15:53:16
длительность: 06:00:00

E. Версия 0.9.9.9.9.9....

Язык	Ограничение времени	Ограничение памяти	Ввод	Вывод
Все языки	2 секунды	512Mb	стандартный ввод или input.txt	стандартный вывод или output.txt
Oracle Java 7	3 секунды	512Mb		
Python 3.9.1	5 секунд	512Mb		
Python 2.7 (PyPy 4.0.0)	5 секунд	512Mb		
Python 3.7 (PyPy 7.3.3)	5 секунд	512Mb		
Python 2.7	5 секунд	512Mb		
Oracle Java 8	3 секунды	512Mb		
Oracle Java 7 x32	3 секунды	512Mb		

Есть сложная программная система, состоящая из 3 модулей — A, B, C. Модули независимо друг от друга обновляются, у каждого из них есть номер версии (например A23, B2 или C10). 3 номера версий модулей составляют версию системы - версии из примера образуют версию системы $[23, 2, 10]$.
Минимальная версия каждого модуля равна 1, а максимальные версии для каждого модуля свои — N_a, N_b, N_c.

Хотя модули довольно независимы, но некоторые связи между ними есть. Вследствие этого существуют правила вида «Если версия модуля X равна K или выше, то версия модуля Y обязательно должна быть M или выше», причем правила односторонние.

Пример такого правила: ($A5$, $B3$) — «Если модуль A версии 5 или выше, то модуль B должен быть версии 3 или выше».

Такому правилу будут удовлетворять следующие пары версий модулей (A, B): $(5, 3)$, $(5, 4)$, $(6, 3)$, $(4, 2)$, $(4, 3)$, — но не будет удовлетворять пара $(5, 2)$. Обратите внимание, что пара $(4, 2)$ является корректной, так как $A4 < A5$, а значит правило на неё не распространяется.

Комбинации правил могут в принципе исключать возможность использования той или иной версии (особенности разработки), к примеру правила ($C5$, $B3$) и ($B2$, $C6$) делают версию $C5$ невозможной, так как для $C5$ требуется не менее $B3$, а для $B3$ (так как она выше $B2$) требуется минимум $C6$.

Назовем версию системы корректной, если она удовлетворяет всем заданным правилам.

Вычислите количество корректных версий системы с учетом заданных правил и максимальных версий модулей.

Формат ввода

В первой строке находятся три целых числа N_a, N_b, $N_c (1 \le N_a, N_b, N_c \le 500\ 000)$ — максимальные версии модулей A, B, C соответственно.
Во второй строке расположено целое число $Q (0 \le Q \le 200\ 000)$ — количество правил.

Каждая из следующих Q строк содержит четыре целых числа X_i, K_i, Y_i, $M_i (1 \le X_i, Y_i \le 3, X_i \ne Y_i; 1 \le K_i \le N_{X_i}; 1 \le M_i \le N_{Y_i})$ — i-е правило вида «Если версия модуля X равна K или выше, то версия модуля Y обязательно должна быть M или выше».

Модули в правилах занумерованы в алфавитном порядке: A — 1, B — 2, C — 3.

Формат вывода

В единственной строке выведите количество корректных версий системы - наборов версий модулей, удовлетворяющих всем заданным правилам.

Пример 1

Ввод	Вывод
1 2 3 0	6

Пример 2

Ввод	Вывод
3 3 3 1 1 1 2 2	18

Пример 3

Ввод	Вывод
6 7 8 3 3 5 2 3 2 2 3 6 1 4 3 3	108

Примечания

В первом тесте нет правил, поэтому все версии системы являются корректными: $(1, 1, 1)$, $(1, 1, 2)$, $(1, 1, 3)$, $(1, 2, 1)$, $(1, 2, 2)$, $(1, 2, 3)$. Во втором тесте некорректными являются все версии системы с версией $B1$. Таких версий ровно 9 из 27 возможных — все комбинации из трёх версий модуля A и трёх версий модуля C.

Первые два правила третьего теста описаны в условии — это правила ($C5$, $B3$) и ($B2$, $C6$), которые взаимоисключают любые версии системы с версией модуля $C5$.

Язык GNU C++20 10.2

Набрать здесь | Отправить файл

```
1
```

Отправить

Предыдущая Следующая

Mizzza54

Яндекс, Осенняя стажировка 2021

1 авг 2021, 16:47:21
старт: 1 авг 2021, 09:53:16
финиш: 1 авг 2021, 15:53:16
длительность: 06:00:00

F. В поисках сломанного коммита

Язык	Ограничение времени	Ограничение памяти	Ввод	Вывод
Все языки	3 секунды	512Mb	стандартный ввод или input.txt	стандартный вывод или output.txt
Python 3.9.1	6 секунд	512Mb		
Python 2.7 (PyPy 4.0.0)	6 секунд	512Mb		
Python 3.7 (PyPy 7.3.3)	6 секунд	512Mb		
Python 2.7	6 секунд	512Mb		

В одной небезызвестной компании, менеджер старшего разряда Андрей с командой создают новый продукт « Он.Компановщик ». Как и всегда, разработка большого проекта предполагает множество коммитов в репозиторий с проектом, количество которых временами доходит до 100 в день. Так как производить тестирование проекта после каждого коммита не представляется возможным, то в самый неудобный для этого момент (по всем законам жанра ужасов) проект перестал проходить тесты. Молодой, но опытный менеджер Андрей поставил перед собой задачу найти тот коммит, который всё ломает.
Андрей знает, что коммиты в репозитории делятся на два типа: **большие** «B» (big) и **маленькие** «S» (small). Также Андрей знает, что время пересборки проекта зависит только от того, был ли большой коммит между ревизиями сборки. Например, если последовательность коммитов выглядит как «BSBS», а у Андрея есть в текущий момент собранный проект после последнего (4-го) коммита, то он может собрать проект после 3-го коммита за a минут, так как четвёртый коммит маленький, а проект после второго коммита за b минут, так как 4-й коммит маленький, но 3-й большой.

Более формально, если в текущий момент проект собран после i-го коммита, а Андрей хочет собрать проект после j-го коммита, то пересборка займёт b минут, если между коммитами i и j есть большой коммит, и a минут в противном случае.

К сожалению, в сфере сборки Андрей ещё слишком неопытен, поэтому он может хранить только один бинарный файл (бинарь) проекта единовременно, и сейчас перед ним собранный бинарь после последнего коммита, который гарантированно не проходит тесты. Можно считать, что бинарь перед первым коммитом был протестирован и все тесты проходил. Заметьте, что в отличие от сборки, время тестирования собранного бинаря равно нулю.

Помогите Андрею рассчитать, за какое минимальное гарантированное время он может найти сломанный коммит.

Гарантируется, что среди коммитов есть ровно один ломающий, а также, что ни один коммит не чинит код после поломки. Иными словами, для некоторого k код проходит тесты после первых k коммитов и не проходит их в дальнейшем.

Формат ввода

В первой строке вводятся три числа n, a, b ($1 \le n \le 3000$, $1 \le a \le b \le 1000$) — количество коммитов в проекте, время сборки только из маленьких коммитов и время сборки, если есть большой коммит соответственно.
Вторая строка содержит n символов «B» и «S», соотвествующие размерам коммитов.

Формат вывода

В единственной строке выведите минимальное гарантированное время, за которое можно найти сломанный коммит.

Пример 1

Ввод	Вывод
5 1 100 SBSSS	101

Пример 2

Ввод	Вывод
8 2 4 BBSSBBBS	12

Пример 3

Ввод	Вывод
13 324 845 SSSBSSBBSSSBB	2859

Примечания

В первом примере из условия собран бинарь после последнего коммита, будем обозначать текущий бинарь символом «x».

1. 0 минут состояние «SBSSSx».
2. Сборка релиза после второго коммита будет занимать 1 минуту, так как разница между ревизиями состоит только из трёх маленьких коммитов («SSS»).
3. 1 минута состояние «SBxSSS».
4. В зависимости от того, проходит ли тесты данная ревизия, следующий шаг должен быть либо «SxBSSS», либо «SBSxSS»). В первом случае пересборка будет занимать 100 минут, а после её тестирования мы однозначно поймём, какой именно коммит содержит ошибку.

   Во втором случае пересборка занимает 1 минуту, но однозначно определить коммит получится не всегда, так как будет необходима сборка ревизии «SBSSxS», если ревизия «SBSxSS» не содержит ошибок. В итоге максимальное время проверки будет $\max(1+100, 1+1, 1+1+1) = 101$.

Во втором примере из условия изначальное состояние «BBSSBBBSx».

1. Сначала собираем за 4 минуты ревизию «BBSSxBBBS».
2. Если в ревизии содержится ошибка, то собираем «BBxSSBBBS» за 2 минуты, а дальше либо «BxBSSBBBS» за 4, либо «BBSxSBBBS» за 2 и находим нужный коммит.
3. Если в ревизии не содержится ошибка, то собираем «BBSSBBxBS» за 4 минут, дальше либо «BBSSBxBBS» за 4, либо «BBSSBBBxS» за 4 и находим нужный коммит.
4. В итоге максимальное время проверки будет $\max(4+2+4, 4+2+2, 4+4+4, 4+4+4) = 12$.

Язык Oracle Java 8

Набрать здесь | Отправить файл

```java
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.NavigableSet;
import java.util.StringTokenizer;
import java.util.TreeSet;

/**
 * @author Michael Gerasimov
 */
public class BigAndSmallCommits {
    FastReader scanner = new FastReader(System.in);
    int n, a, b;
    char[] history;
    NavigableSet<Integer> bigIndexes = new TreeSet<>();

    private void run() {
        n = scanner.nextInt();
        a = scanner.nextInt();
        b = scanner.nextInt();
        history = scanner.next().toCharArray();
        for (int i = 0; i < history.length; i++) {
            if (history[i] == 'B') {
                bigIndexes.add(i);
            }
        }
        System.out.println(solve(0, n - 1, 0, Trend.LEFT));
    }

    private int solve(int l, int r, int sum, Trend trend) {
        if (l == r || r < 0 || l > r) {
//            switch (trend) {
//                case LEFT:
//                    sum -= history[l + 1] == 'S' ? a : b;
//                    break;
//                case RIGHT:
//                    sum -= history[l - 1] == 'S' ? a : b;
//                    break;
```

Отправить

Предыдущая