1. getComparedArrayElements

**Первый аргумент:**

Матрица, которая строится следующим образом (на примере сервисов VF5 и VF6):

В VF5 два сервиса, а в VF6 три => матрица будет строиться так:

Берётся первый сервис из VF5 и делается compareTo с каждым сервисом из VF6 получается 3 числа, со вторым сервисом из VF5 делается тоже самое и получается ещё 3 числа, тем самым мы получаем следующее:

gradesMin:

|  |  |  |
| --- | --- | --- |
| 105 | 55 | 155 |
| 202 | 51 | 51 |

**Второй аргумент:**

Худшая оценка без учёта «additional» полей, эта оценка для каждого объекта своя и равна она максимальному значения compareTo объекта по mandatory полям, пример:

Сервис имеет 6 mandatory полей, за каждое несовпадение такого поля compareTo начисляет +50, конкретно у сервиса есть поле hashes это объект, который имеет свои mandatory поля, поэтому compareTo обратится сначала к hashes, в котором 2 mandatory поля за каждое несовпадение которых даётся ещё +50, таким образом сервис на самом деле имеет 7 mandatory полей, поэтому худшая оценка для сервиса будет:

gradesWorst = 50\*7 = 350

**Алгоритм поиска сравниваемых объектов:**

1. Ишим минимальное число в матрице и записываем его координату:

* 51
* [1,1]

1. Проверяем есть ли ещё в строке 51

* Да, есть:

|  |  |  |
| --- | --- | --- |
| 105 | 55 | 155 |
| 202 | 51 | 51 |

1. Для того, чтобы понять какой элемент нам выгоднее сравнить, надо посмотреть, что больше подходит для 0-ого элемента для этого, находим 55 и 155 и сравниваем их и выбираем ту 51, над которой стоит 155, потому что для первого больше подходит элемент с индексом 1 и оценкой 55 (чем меньше оценка, тем больше подходит). Записываем в результат координаты: [1,2]
2. Далее происходит поиск следующего минимального элемента по оставшейся матрице (вторая строка и третий столбец не просматриваются, тк они уже записаны в результат)

|  |  |  |
| --- | --- | --- |
| 105 | 55 | **155** |
| **202** | **51** | **51** |

1. Минимальное число 55 записываем координату [0,1]
2. Проверяем есть ли ещё в строке 55, если нету то сразу записываем в результат координату
3. На выходе получаем: {0=1, 1=2}

Пример для объяснения:

|  |  |  |  |
| --- | --- | --- | --- |
| 203 | 201 | 202 | 204 |
| 200 | 47 | 47 | **47** |
| 105 | 65 | 65 | 65 |
| 111 | 67 | 68 | 69 |

|  |  |  |  |
| --- | --- | --- | --- |
| 203 | 201 | 202 | 204 |
| 200 | 47 | 47 | **47** |
| 105 | 65 | 65 | 65 |
| 111 | 67 | 68 | 69 |

|  |  |  |  |
| --- | --- | --- | --- |
| 203 | 201 | 202 | 204 |
| 200 | 47 | 47 | **47** |
| 105 | 65 | 65 | 65 |
| 111 | 67 | 68 | 69 |

|  |  |  |  |
| --- | --- | --- | --- |
| 203 | 201 | 202 | 204 |
| 200 | 47 | 47 | **47** |
| 105 | 65 | 65 | 65 |
| 111 | 67 | 68 | 69 |

Ответ: {0=0, **1=3**, 2=2, 3=1}