

Минимумы на дугах

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	3 секунды
Ограничение по памяти:	1024 мегабайта

Данную задачу можно решать только на языке программирования C++.

По кругу расположено n различных чисел от 1 до n , в i -й вершине при обходе против часовой стрелки записано число p_i . Для каждой пары значений x и y есть два способа добраться из x в y — пойти по часовой стрелке или против. В этой задаче n всегда нечетное, поэтому эти два пути будут иметь различную длину.

Скажем, что $f(x, y)$ — это минимальное значение, которое мы встретим по пути, если добираться из числа x в число y по кратчайшему из путей.

Например, если $p = [1, 5, 3, 2, 7, 4, 6]$, то значения 5 и 7 соединены двумя путями, и значения вершин на этих путях, соответственно, равны $[5, 3, 2, 7]$ и $[5, 1, 6, 4, 7]$. Кратчайшим является первый указанный путь, а значит $f(5, 7) = \min\{5, 3, 2, 7\} = 2$.

Скажем, что две перестановки p и q — *эквивалентные*, если значение функции f для этих перестановок совпадает при всех x и y . Вам загадана перестановка p , требуется отгадать любую эквивалентную ей перестановку q , задавая запросы к функции $f(x, y)$.

Формат решения

Это необычная задача. Она имеет формат тестирования с грейдером, где вам необходимо реализовать только функцию `guess` с решением. Эта функция будет вызвана тестирующей программой жюри (грейдером), и возвращаемое значение функции будет принято как решение задачи.

В частности, это означает, что в отправленном вами коде **не должно быть ввода или вывода**. Ваш код **не должен** содержать функцию `main`. При необходимости вы можете реализовать произвольное количество вспомогательных функций, структур, классов и глобальных переменных, но весь код вашего решения должен находиться в одном файле.

Вы должны реализовать следующую функцию:

```
std::vector<int> guess(int n);
```

Функция `guess` принимает на вход число n — длину перестановки.

В реализации функции `guess` вы можете использовать функцию `min_value`, которую реализует грейдер. Данная функция принимает на вход два значения x и y , и возвращает $f(x, y)$ в качестве результата. Если сделать некорректный запрос или исчерпать количество запросов, то программа автоматически завершит свою работу.

Для того, чтобы получить доступ к функции `min_value` в решении, первой строчкой вашего кода нужно подключить заголовочный файл следующей строчкой:

```
#include "checkpoint.h"
```

Определение функции `min_value` в заголовочном файле следующее:

```
int min_value(int x, int y);
```

Все параметры (значения x , y , а так же возвращаемая вами перестановка q) заданы в **1-индексации**.

Ваша функция `guess` с помощью вызовов функции `min_value` должна определить неизвестную перестановку p , с точностью до указанной эквивалентности. То есть если жюри загадали перестановку p , а функция `guess` вернула любую эквивалентную ей перестановку q , то такой ответ считается корректным.

За один запуск грейдера **жюри может делать несколько вызовов функции `guess`**, в таком случае функция `min_value` будет возвращать значения $f(i, j)$, определенные для текущей загаданной перестановки p в рамках конкретного вызова функции `guess`.

Грейдер не является адаптивным, то есть перестановки p фиксируются заранее и не зависят от реализации функции `guess`.

Тестирование

Вам предоставлен шаблон решения `checkpoint.cpp`, а так же заголовочный файл `checkpoint.h`, содержащий определение функций `min_value` и `guess`.

Для удобства тестирования вам предоставлен грайдер — файл `grader.cpp`. В этом файле реализован ввод входных данных со стандартного потока ввода, запуск функции `guess` и вывод на стандартный поток вывода возвращаемого значения функции `guess`. В тестирующей системе эти файлы грайдеров могут отличаться.

Чтобы скомпилировать ваш код `checkpoint.cpp` на языке C++, используйте команду `g++ -std=c++20 grader.cpp checkpoint.cpp -o grader`

После выполнения данной команды будет создан исполняемый файл грайдера `grader` или `grader.exe`, в зависимости от вашей операционной системы, запустив который можно ввести тест в формате, указанном далее.

Если компиляция через команды вызывает у вас трудности, для локального тестирования вы можете скопировать реализацию функции `guess` в файл `grader.cpp` и запустить файл `grader.cpp`. При этом перед отправкой решения в тестирующую систему вам нужно будет оставить только реализацию функции `guess`, не забыв подключить заготовочный файл в начале кода.

Формат входных данных

Грайдер читает тест в следующем формате:

В первой строке указано число t ($1 \leq t \leq 30\,000$) — количество наборов входных данных.

Первая строка каждого набора входных данных содержит **нечетное** число n ($1 \leq n \leq 30\,000$) — длину перестановки.

Вторая строка содержит n различных чисел p_1, \dots, p_n ($1 \leq p_i \leq n$) — загаданную перестановку.

Формат выходных данных

Грайдер выводит результаты функции `guess` — угаданную перестановку для каждого набора входных данных.

В файле `grader.cpp` есть переменная `verbose`, которая исходно равна 0. При увеличении ее значения грайдер будет более подробно писать информацию о вашем решении и его запросах.

Пример

стандартный ввод	стандартный вывод
2	queries count => 3
3	queries count => 10
1 2 3	
5	
1 4 2 3 5	

Система оценки

Тесты к этой задаче состоят из четырех групп.

Баллы за каждую из первых трех групп ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Балл за последнюю группу равняется минимальному из баллов, полученных за каждый из тестов в четвертой группе.

Для каждого теста обозначим за N сумму n по всем наборам входных данных, за Q — ограничение на суммарное количество вызовов функции `min_value` по всем наборам входных данных.

Группа	Баллы	Доп. ограничения		Необх. группы	Комментарий
		N	Q		
0	0	–	$Q = 4950$	–	Тесты из условия
1	10	$N \leq 100$	$Q = 4950$		$p_{2i-1} = \frac{n-1}{2} + i$ для $1 \leq i \leq \frac{n+1}{2}$
2	20	$N \leq 100$	$Q = 4950$	0, 1	
3	20	$N \leq 1\,000$	$Q = 100\,000$	0, 1, 2	
4	60	$N \leq 30\,000$	$Q = 1\,000\,000$	–	

При этом последняя группа оценивается по формуле. Если в тесте вы сделали x запросов, то ваш балл будет равен:

$$\text{score} = \min \left(60, \left\lfloor 95 \cdot \left(1 - \frac{x}{10^6} \right) \right\rfloor \right)$$

Ваш балл за последнюю группу равняется минимальному из полученных баллов на тестах четвертой группы.