Task 1. Namuhs

Инопланетяне-гуманоиды с планеты Хтрае (они зовутся намуанцы) находятся на пике своего развития. После большого путешествия вокруг нашей вселенной специальная команда намуанцев составила список планет, отсортированных по расстоянию от своей планеты Хтрае. Команда отдала список “Верховному совету” намуанцев, который детально проанализировал информацию и решил присвоить каждой планете целое число, описывающее ее потенциал для эксперимента. Глава “Верховного совета” Дени решает, какие планеты войдут в эксперимент. Так как расстояния в космосе очень большие даже для продвинутых путешественников, должно быть выбрано только одно множество планет, которые находятся рядом друг с другом в представленном списке, а именно, они должны формировать отрезок этого списка. Общий потенциал одного множества планет равен сумме потенциалов каждой планеты в нем. “Верховный совет” решил, что выбранное множество должно иметь максимальный возможный суммарный потенциал. Дени помнила, что она знает одну планету неподалеку – планету Земля, где цивилизация (также называемая человечеством) не так продвинута, но на ней существуют живые существа, которые помогут ей создать программу для решения проблемы выбора оптимального множества. Намуанцы не хотят делиться собранной секретной информацией, поэтому они готовы только сравнивать суммы двух интервалов планет, о которых вы решите спросить.

Задача

Вам необходимо реализовать функцию *find_max*, которая будет компилироваться с исходным файлом жюри (написанным Дени). Она должна находить начало и конец отрезка находящихся подряд планет, которые имеют максимально возможный потенциал. Это функция на вход получает одно число ***N*** – количество планет. Жюри знает значение потенциалов планет в описанном выше порядке. Ваша цель, задавая вопросы для сравнения отрезков расположенных подряд в этом списке планет, найти отрезок с максимально возможным потенциалом. Гарантируется, что существует только один искомый отрезок!

Детали реализации

Функция *find_max* имеет следующий прототип:

```
void find_max (int N, int& left, int& right);
```

Она вызывается один раз программой жюри с аргументом *N* – количеством планет в списке. Когда ваша программа находит ответ, вы должны сохранить его в параметрах *left* и *right* – левой и правой границе отрезка соответственно (нумерация начинается с 1).

Для взаимодействия с программой жюри у вас есть функция:

```
bool compare_segments (int left1, int right1, int left2, int right2);
```

Она определяет, верно ли, что общий потенциал планет отрезка от *left1* до *right1* (включительно) больше или равен потенциалу планет отрезка от *left2* до *right2* (включительно). При этом должны выполняться следующие условия: $1 \le left1 \le right1 \le N$ и $1 \le left2 \le right2 \le N$. Заметим, что функция *compare_segments* работает за линейное время от входных данных, то есть ей требуется (*right1*-*left1*+1)+(*right2*-*left2*+1) действий, чтобы найти ответ!

Вы должны посылать на проверку только файл **namuhs.cpp**, который содержит функцию *find_max*. Этот файл может содержать также другой код и функции, необходимые для работы *find_max*, но не должен содержать функцию *main*. В начале вашего файла вы должны написать: **#include “namuhs.h”**.

Ограничения

$2 \le N \le 10^5$

В 10% тестов $N \le 10^2$

В других 30% тестов $N \le 10^3$

Пример взаимодействия с программой жюри

Пусть список потенциалов планет 2, -3, 5, -2, 3. Здесь ответом является отрезок планет с позиции 3 до позиции 5 включительно. Общее число планет равно 5, поэтому программа жюри будет вызывать вашу функцию *find_max* как *find_max(5,left,right);*

Пример запросов для нахождения ответа может быть следующим:

Вызываемая у программы жюри функция	**Результат**	**Пояснение**
compare_segments(3,5,1,1)	true	Ваша программа спрашивает, верно ли, что общий потенциал отрезка с 3 по 5 больше или равен потенциала планеты 1 (с 1 по 1). Так как (5+(-2)+3) ≥ 2, функция возвращает true.
compare_segments(1,1,3,5)	false	Это означает, что общий потенциал планет отрезка с 3 по 5 строго больше, чем потенциал с 1 по 1.
compare_segments(3,5,2,2)	true	Здесь требуется 4 действия для выполнения функции
compare_segments(3,5,3,3)	true	Здесь требуется 4 действия для выполнения функции
compare_segments(3,5,4,4)	true	Здесь требуется 4 действия для выполнения функции
compare_segments(3,5,5,5)	true	Здесь требуется 4 действия для выполнения функции
compare_segments(3,5,1,2)	true	Здесь требуется 5 действий для выполнения функции
compare_segments(3,5,2,3)	true	Здесь требуется 5 действий для выполнения функции
compare_segments(3,5,3,4)	true	Здесь требуется 5 действий для выполнения функции
compare_segments(3,5,4,5)	true	Здесь требуется 5 действий для выполнения функции
compare_segments(3,5,1,3)	true	Здесь требуется 6 действий для выполнения функции
compare_segments(3,5,2,4)	true	Здесь требуется 6 действий для выполнения функции
compare_segments(3,5,1,4)	true	Здесь требуется 7 действий для выполнения функции
compare_segments(3,5,2,5)	true	Здесь требуется 7 действий для выполнения функции
compare_segments(3,5,1,5)	true	Мы уже проверяли, что общий потенциал планет на отрезке с 3 по 5 больше или равен потенциалу любого другого отрезка. Поэтому этот отрезок дает нам оптимальный ответ. Теперь можно установить значение параметров *left*=3 и *right*=5, и функция должна закончить свою работу.

Локальное тестирование

Вам даны файлы **Lgrader.cpp** и **namuhs.h** для локального тестирования. Их необходимо поместить в ту же папку, что и вашу программу **namuhs.cpp**, и вы должны компилировать файл **Lgrader.cpp**. Таким образом, вы получите программу, проверяющую корректность вашей функции. Эта программа требует следующих входных данных:

- в первой строке одно положительное число – количество планет в списке.
- во второй строке потенциалы планет из списка.

Выходные данные равны ответу, полученному вашей функцией.

Посылка ваших тестов в систему

Вы можете посылать ваши тесты в систему. Формат входных данных тот же, что и у программы-грейдера для локального тестирования. В качестве выходных данных вы получите число больше 0, если ответ вашей программы верный и 0, если он не верен.