Лабораторная работа № 8 по курсу дискретного анализа: «Жадные алгоритмы»

Выполнил студент группы М8О-307Б-21: Меркулов Фёдор Алексеевич

Условие:

Вариант: 3 Максимальный треугольник

Заданы длины N отрезков, необходимо выбрать три таких отрезка, которые образовывали бы треугольник с максимальной площадью.

Формат ввода

На первой строке находится число N, за которым следует N строк с целыми числами-длинами отрезков.

Формат вывода

Если никакого треугольника из заданных отрезков составить нельзя – 0, в противном случае на первой строке площадь треугольника с тремя знаками после запятой, на второй строке – длины трёх отрезков, составляющих этот треугольник. Длины должны быть отсортированы.

Пример

Ввод	**Вывод**
4 1 2 3 5	0

Метод решения

Для максимизации площади треугольника не обязательно максимизировать все его стороны. Формула Герона, которая используется для вычисления площади треугольника, зависит от длин всех трех его сторон, но это не означает, что увеличение всех сторон приведет к максимальной площади. Давайте разберемся, почему это так.
Формула Герона для площади треугольника выглядит следующим образом:
$S = \sqrt{p(p - a)(p - b)(p - c)}$
где:

S - площадь треугольника,
a, b, c - длины сторон треугольника,
p - полупериметр треугольника, который вычисляется как $\frac{a+b+c}{2}$.
Чтобы максимизировать площадь треугольника, нам необходимо максимизировать выражение под знаком квадратного корня. Из этой формулы можно сделать следующие выводы:
Если одна из сторон (например, сторона a) стремится к нулю, то площадь такого треугольника также стремится к нулю. То есть, уменьшение длины хотя бы одной стороны уменьшает площадь треугольника.
Если одна из сторон (например, сторона a) увеличивается, то площадь треугольника увеличивается, при условии, что сумма длин двух других сторон (b и c) остается постоянной.
Итак, чтобы максимизировать площадь треугольника, нам необходимо увеличивать длины его сторон так, чтобы сумма длин двух других сторон оставалась постоянной. Это позволит максимизировать выражение под знаком квадратного корня в формуле Герона и, следовательно, максимизировать площадь треугольника.
То есть если зафиксировать a и b, то c обязательно попытается стать самым большим отрезком из возможных, теперь для попытки максимизации площади треугольника зафиксируем c и, допустим, b, тогда а выберет максимально возможный отрезок и повторим эту попытку максимизации площади c тоже попытается стать максимальным отрезком из возможных. То есть нет смысла рассматривать все группировки троек отрезков и проверять можно ли из них составить треугольник, а достаточно будет рассматривать 3 идущие подряд отрезка и проверять можно ли из них сделать треугольник и максимальна ли площадь получившегося треугольника.

Описание программы

Считываем входные данные в вектор $lengths$, сортируем его по убыванию и для каждого i выделяем 3 подряд идущих отрезка (максимальную тройку из нерассмотренных), проверяем можно ли из них составить треугольник и смотрим является ли площадь получившегося треугольника больше максимально найденной до этого. В итоге находим тройку отрезков из которых получается треугольник максимальной площади и выводим ответ.

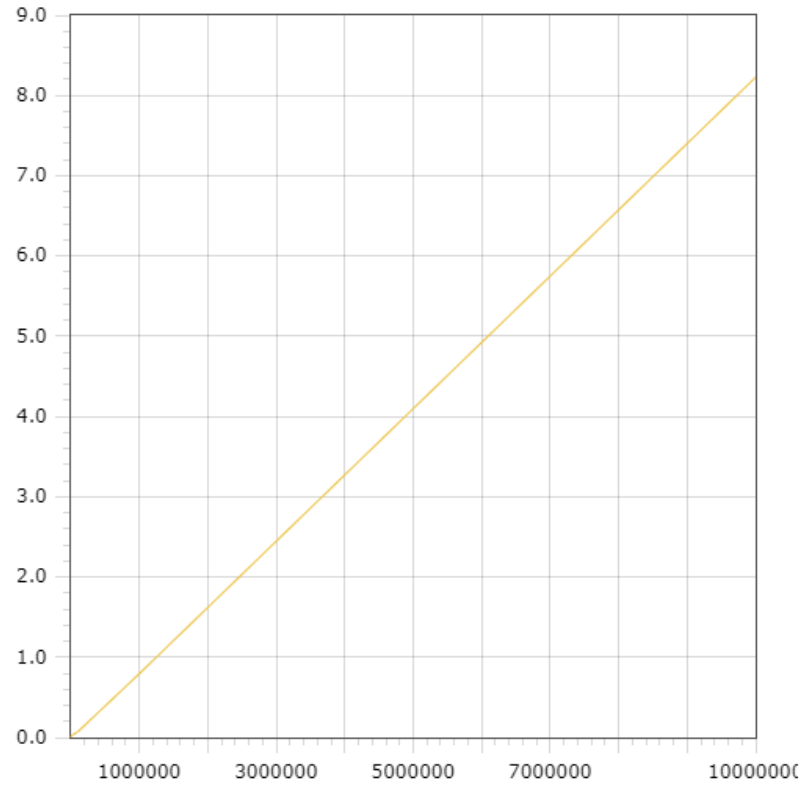
Дневник отладки

$WA3$ из-за неправильной логики (остановка при первой возможности сделать треугольник из максимально возможных отрезков)

Тест производительности

Время работы алгоритма я решил проверить с помощью утилиты time. Тесты имеют название $testN.txt$, где N означает количество отрезков $= 10\hat{}N$

```
papik@papik-VirtualBox:~/DA8$ time ./DA8 < test1.txt
real    0m0,007s
user    0m0,000s
sys     0m0,007s
papik@papik-VirtualBox:~/DA8$ time ./DA8 < test2.txt
real    0m0,008s
user    0m0,000s
sys     0m0,008s
papik@papik-VirtualBox:~/DA8$ time ./DA8 < test3.txt
real    0m0,008s
user    0m0,004s
sys     0m0,004s
papik@papik-VirtualBox:~/DA8$ time ./DA8 < test4.txt
real    0m0,025s
user    0m0,020s
sys     0m0,005s
papik@papik-VirtualBox:~/DA8$ time ./DA8 < test5.txt
real    0m0,070s
user    0m0,065s
sys     0m0,005s
papik@papik-VirtualBox:~/DA8$ time ./DA8 < test6.txt
real    0m0,795s
user    0m0,745s
sys     0m0,036s
papik@papik-VirtualBox:~/DA8$ time ./DA8 < test7.txt
real    0m8,226s
user    0m8,070s
sys     0m0,150s
```



Согласно данным измерениям, можно сказать, что с ростом количества входных данных время работы согласуется с предполагаемой сложностью $O(nlogn)$ (Сортировка всех отрезков по убыванию с помощью функции sort : $nlogn$, поиск подходящих, подряд идущих, троек отрезков: n. то есть $O(nlogn + n) = O(nlogn)$)

Вывод

Выполнив лабораторную работу №8 по курсу «Дискретный анализ», я изучил жадные алгоритмы и убедился, что это довольно интересный способ решения задач, его достаточно легко понять. Однако стоит заметить, что не во всех задачах можно применить данный подход. Что, в конечном итоге, приводит нас к тому, что нужно понимать, когда нужно применять такой метод решения.