**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»**

# ОТЧЕТ

## о выполнении лабораторной работы №7

## «Жадные алгоритмы»

**по дисциплине**

*«Дискретный анализ»*

Выполнил студент группы М8О-308Б-23:
**Ибрагимов Далгат Магомедалиевич**

Проверил:
**Макаров Н.К.**

Москва, 2025

**Постановка задачи**

**Вариант 3: Максимальный треугольник**

## Цель работы

Изучение применения жадных алгоритмов для оптимизации вычислений.

## Задание

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.

## Реализация программы

В данной лабораторной работе была реализована программа на языке C++ с использованием жадных алгоритмов. Программа включает следующие функции:

- `double calculateTriangleArea` - вычисляет площадь треугольника по формуле Герона.

## Листинг кода

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <cmath>
#include <iomanip>

double calculateTriangleArea(double s1, double s2, double s3, double halfSum)
{
    return sqrt(halfSum * (halfSum - s1) * (halfSum - s2) * (halfSum - s3));
}
```

```
int main()
{
    int count;
    std::cin >> count;
    std::vector<int> elements(count);
    for (int i = 0; i < count; ++i)
    {
        std::cin >> elements[i];
    }

    if (count < 3)
    {
        std::cout << 0 << "\n";
        return 0;
    }

    std::sort(elements.rbegin(), elements.rend());

    int index = 0;
    std::cout << std::fixed << std::setprecision(3);

    double maxTriangleArea = -1.0;
    int foundIndex = -1;
    bool isPossible = false;

    while (index < count - 2)
    {
        if (elements[index] < elements[index + 1] + elements[index + 2])
        {
            double side1 = (double)elements[index], side2 = (double)elements[index +
            double halfSum = (side1 + side2 + side3) / 2.0;

            double currentArea = calculateTriangleArea(side1, side2, side3, halfSum);

            if (currentArea > maxTriangleArea)
            {
```

```
                maxTriangleArea = currentArea;
                foundIndex = index;
            }
            isPossible = true;
        }
        ++index;
    }

    if (isPossible)
    {
        std::cout << maxTriangleArea << "\n";
        std::cout << elements[foundIndex + 2] << " " << elements[foundIndex + 1] << "
    }
    else
    {
        std::cout << 0 << "\n";
    }
}
```

## Описание работы программы

Программа принимает количество отрезков и их длины в качестве входных данных. После этого она проверяет, возможно ли сформировать треугольник из заданных отрезков. Для этого используется жадный подход, который включает сортировку длин отрезков в порядке убывания и проверку условия существования треугольника: сумма длины двух сторон должна быть больше длины третьей стороны. Программа вычисляет площадь потенциального треугольника с помощью формулы Герона и выбирает максимальную из возможных площадей. Если удается найти хотя бы один треугольник, программа выводит его площадь и длины отрезков, формирующих этот треугольник.

## Дневник отладки

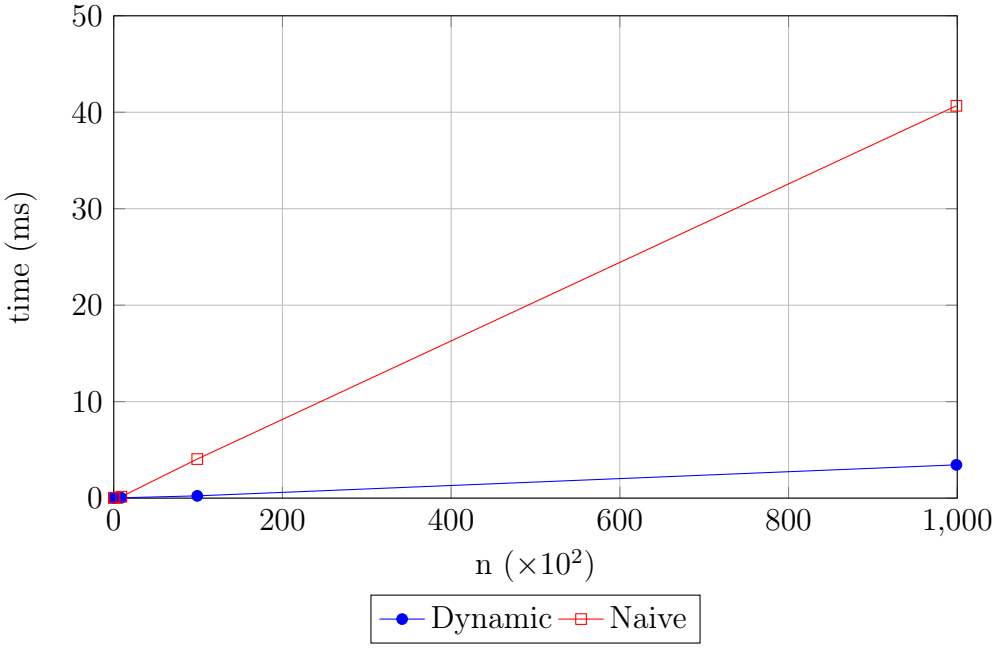
1. Все этапы разработки программы проходили без значительных ошибок.
2. В процессе тестирования не было выявлено логических ошибок, и программа корректно обрабатывала различные входные данные.

## Результаты тестирования

Программа успешно продемонстрировала корректные результаты для различных наборов входных данных, включая крайние случаи, такие как минимальные и максимальные значения длины отрезков. Это подтверждает правильность выбранного алгоритма и его эффективность в решении поставленной задачи.

### Тест производительности

Сложность алгоритма O(n log n) Для оценки производительности алгоритма была проведена серия тестов, сравнивающих время выполнения данного решения с наивным методом. Входные данные были выбраны таким образом, чтобы включать большие значения длины отрезков, что позволяет проанализировать эффективность алгоритма.



## Выводы

В ходе выполнения лабораторной работы была разработана программа, использующая жадный алгоритм для решения задачи выбора отрезков, формирующих треугольник максимальной площади. Эффективность алгоритма была подтверждена через тестирование и сравнение с наивными методами. Результаты показывают, что применение жадного подхода значительно ускоряет процесс вычислений, что делает его предпочтительным для данной задачи.