

编号: 1-1



山东师范大学
SHANDONG NORMAL UNIVERSITY

信息科学与工程学院实验报告

《面向对象程序设计》

Object-Oriented Programming

姓名: 熊淑贤

学号: 201911010517

班级: 计本 1901

时间: 2020 年 10 月 21 日



《面向对象程序设计》实验报告

基本要求：请围绕实验目的、实验内容、实验过程、实验结果（附图）、实验总结（重点阐述）五个部分进行撰写。若报告中若涉及源代码内容，请在附录部分提供完整源码及 GitHub 源码托管地址。报告撰写完毕后请提交 PDF 格式版本到云班课。

一、实验目的

- 1、理解 c++对 c 的各项改进和扩展基本原理
- 2、熟练运用 C++特色函数解决实际问题
- 3、理解并掌握 C++指针和引用的本质机理
- 4、熟练掌握 C++动态内存申请和释放方法
- 5、掌握 visual studio 代码调试方法

二、实验内容

【编程设计题】给定 m 根木棍，每根木棍的长度记为 l_i , ($3 \leq i \leq m$)，下面欲从这 m 根木棍中选择 3 根木棍组成周长尽可能最长的三角形、面积尽可能最大的三角形，分别输出最大的周长和面积。如果怎么选都无法构成三角形，请直接输出 0。

要求：

1. 算法具有良好的可读性、稳健性和通用性（适合整数长度，浮点数长度）。
2. 给出算法的复杂度分析, 算法复杂度尽可能越低越好。
3. 算法设计时采用指针，引用，重载函数，及动态内存申请等 C++核心特性。

输入：

$M = 5$

$L = 2 \ 3 \ 4 \ 5 \ 10$

输出:最大周长 12，最大面积 6(选择 3, 4, 5)

二、实验过程

1. 首先考虑三重 for 循环遍历所有的边，找出所有组可以构成三角形的边，分别求出周长、面积，与目前为止最大的周长、面积比较，如果改组边周长或面积大于目前为止记录的周长\面积，更新。时间复杂度为 $O(n^3)$ 。



2. 结合题目要求, 尽量降低算法复杂度, 考虑对给出的边从大到小排序。

对于周长, 向后遍历, 看连续的三边能否构成三角形, 如果可以, 计算周长, 如果不可以, 就把最大的边长舍掉 (因为该边和近邻较大的两边无法构成三角形, 更不可能再和其他边构成三角形), 继续遍历, 找到的第一个可构成三角形的三边, 计算周长, 即为答案。

对于面积, 遍历同理, 但找到的第一个可构成三角形的三边未必是答案, 所以要遍历完所有的边。

时间复杂度为 $O(n \log n)$ 。

三、实验结果

```
----- 输入数据 -----
请输入木棍的总个数:
5
请依次输入 5 根木棍的长度
2 3 4 5 10

----- 输出结果 -----
最大周长: 12
边长分别为: 5, 4, 3

最大面积: 6
边长为: 5, 4, 3

请按任意键继续. . .
```

四、实验总结

程序实践是重要教学环节之一。我们需要通过程序实践, 增强工程化意识, 提高 c 语言实践能力。强化计算机应用技能, 从而巩固和充实所学的理论知识, 加深对相关内容的理解, 拓宽知识面, 培养的创新精神和实践能力。

这次的实验是 ACM 中的一道题, 较为锻炼思辨能力, 直接三重 for 循环和容易想到, 是 $O(n^3)$ 的复杂度, 排序也想到了, 但最初思考的仍是排序后再 for 循环, 对于老师给出的解法, 一重遍历连续的三边, 最大周长是第一组可构成三角形的边, 最大面积也只要遍历一次, 这个想法, 是很有借鉴意义的, 计算机大大简化了人类的枯燥计算, 但我们仍不能放弃思考简单精炼的解决方式。

总体而言, 本次收获颇丰, 理解了 c++ 对 C 的各项改进和扩展基本原理, 掌握了 C++ 指针和引用、C++ 动态内存申请和释放方法, 能熟练运用 C++ 特色函数解决实际问题,



■ 附录：程序源码（建议基于 Highlight 软件导入）

GitHub 源码托管地址：

```
01 //define _CRT_SECURE_NO_WARNINGS
02 #include <iostream>
03 #include <math.h>
04 #include <stdlib.h>
05 #include <stdio.h>
06 #include <windows.h>
07 #include <algorithm>
08 using namespace std;
09
10 bool judge(const double &a, const double &b, const double &c) //判断三边能否构成三角形
11 {
12     if ((a + b > c) && (a + c > b) && (b + c > a))
13         return 1;
14     return 0;
15 }
16
17 double C(const double &a, const double &b, const double &c) //求周长
18 {
19     return a + b + c;
20 }
21
22 double S(const double &a, const double &b, const double &c) //求面积，海伦公式
23 {
24     double tp = (a + b + c) / 2.0;
25     return sqrt(tp * (tp - a) * (tp - b) * (tp - c));
26 }
27
28 bool cmp(const double &a, const double &b) //用于sort排序，使各边从小到大排序
29 {
30     return a > b;
31 }
32
33 void MaxC(double *len, const int num) //求最大周长
34 {
35     sort(len, len + num, cmp);
36     int flag = 0;
37     int i, j, k;
38     i = 0, j = 1, k = 2;
39     while (k < num)
40     {
```



```
41     if (judge(len[i], len[j], len[k]))
42     {
43         flag = 1;
44         cout << "最大周长: " << C(len[i], len[j], len[k]) << " \n边长分别为: " << len[i]
45         << ", " << len[j] << ", " << len[k] << endl << endl;
46         return; //第一组可构成三角形的边即为答案
47     }
48     ++i;
49     ++j;
50     ++k; //右移一条边
51 }
52 if (!flag)
53     cout << "当前木棍长度无法构成任何三角形!" << endl;
54 }
55 void MaxS(double *len, const int num) //求最大面积
56 {
57     sort(len, len + num, cmp);
58     int flag = 0;
59     int i, j, k;
60     i = 0, j = 1, k = 2;
61     double tp, ans, a, b, c;
62     ans = 0;
63     while (k < num)
64     {
65         if (judge(len[i], len[j], len[k]))
66         {
67             flag = 1;
68             tp = S(len[i], len[j], len[k]);
69             if (tp > ans)
70             {
71                 ans = tp;
72                 a = len[i];
73                 b = len[j];
74                 c = len[k];
75             }
76         }
77         ++i;
78         ++j;
79         ++k;
80     }
81     if (flag)
82         cout << "最大面积: " << ans << " \n边长为: " << a << ", " << b << ", " << c << endl
83         << endl;
84     else
```



```
84     cout << "当前木棍长度无法构成任何三角形!" << endl;
85 }
86
87 int main()
88 {
89     int m;
90     cout << "----- 输入数据 -----" << endl;
91     cout << "请输入木棍的总个数:\n";
92     cin >> m;
93     double *len = new double[m];
94     cout << "请依次输入 " << m << " 根木棍的长度" << endl;
95     for (int i = 0; i < m; ++i)
96         cin >> len[i];
97     cout << endl << "----- 输出结果 -----" << endl;
98     /*-----*
99     1. 搜索最大周长,并输出周长
100    *-----*/
101    MaxC(len, m);
102    /*-----*
103    2. 搜索最大面积,并输出面积
104    *-----*/
105    MaxS(len, m);
106    delete[] len;
107    system("pause");
108    return 0;
109 }
```