华东师范大学计算机科学与技术学院实验报告

实验课程：计算机图形学	**年级：**2018 级	**实验成绩：**
实验名称：圆算法	**姓名：**	
实验编号：4	**学号：**	**实验日期：**2021-3-30
指导教师：王长波、李洋	**组号：**	**实验时间：**13:00-14:30

一、实验目的

利用操作系统 API 实现基本图元圆的绘制功能。

二、实验内容与实验步骤

实验内容与实验步骤：

根据 TA 所提供模板

实现 DDA 圆算法

实现 Bresenham 圆算法

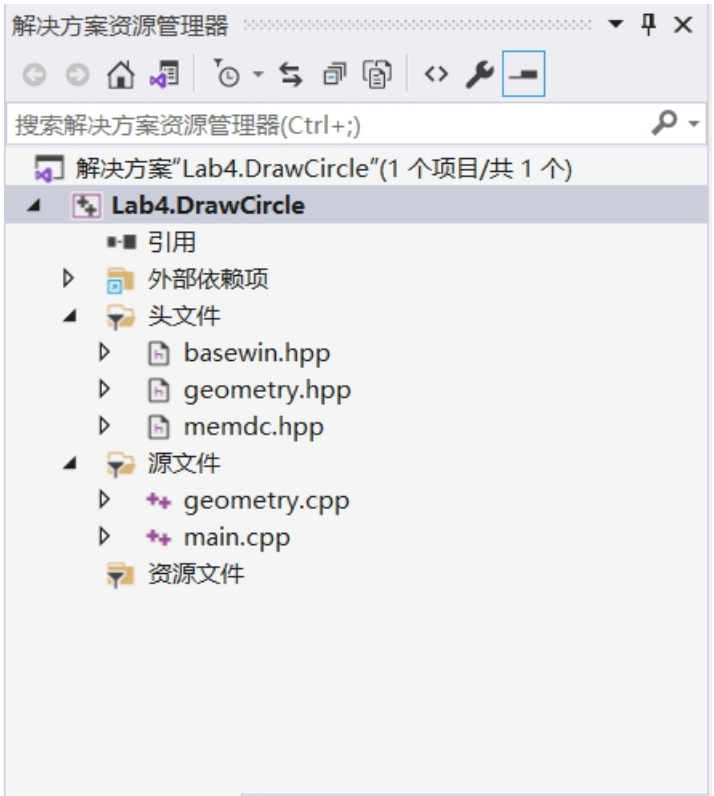
三、实验环境

Windows 10

Visual studio 2015

四、实验过程与分析

1.助教非常 nice 地写好了一个框架，下载之后先创建空项目，然后一个一个导入。



2. 点开 geometry.cpp 发现有两个需要补充的代码。

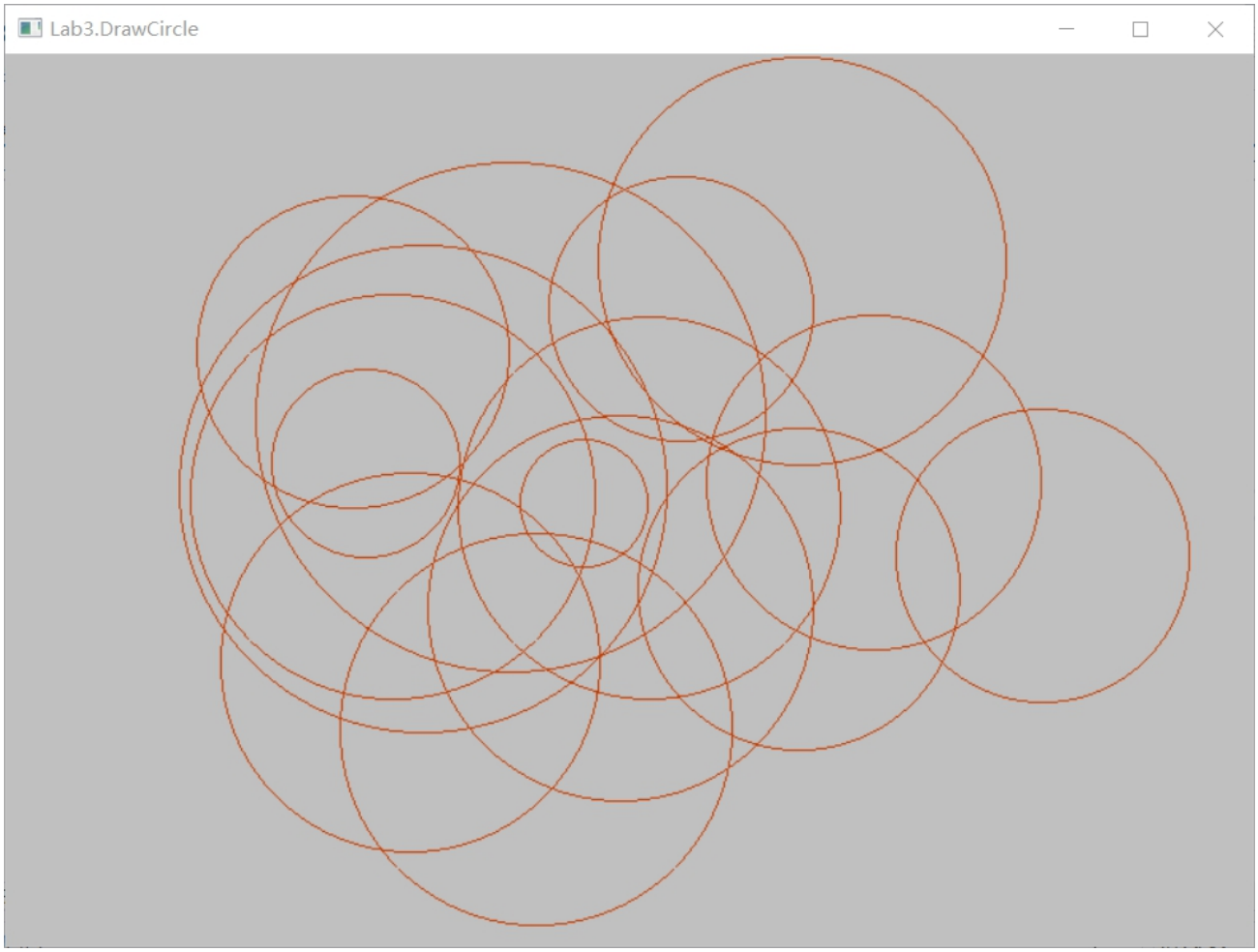
第一个是用简单算法画圆，需要注意的是，圆心不一定在（0,0）处，所以画点的时候注意加上偏移量。主要代码如下

```
void draw_circle_simple(int xc, int yc, int r,
                                    std::vector<std::pair<int, int>> &buffer) {
    for (int i = 0; i <= r/sqrt(2); i++) {
        int y = round(sqrt(pow(r, 2) - pow(i, 2)));
        int x = i;
        buffer.emplace_back(x + xc, y + yc);
        buffer.emplace_back(-x + xc, y + yc);
        buffer.emplace_back(x + xc, -y + yc);
        buffer.emplace_back(-x + xc, -y + yc);
        buffer.emplace_back(y + xc, x + yc);
        buffer.emplace_back(-y + xc, x + yc);
        buffer.emplace_back(y + xc, -x + yc);
        buffer.emplace_back(-y + xc, -x + yc);
    }
}
```

第二个是用 bresenham 算法，根据课堂上讲解的递推式实现，主要代码如下

```
55  void draw_circle_midpoint_bresenham(int xc, int yc, int r,
56                                      std::vector<std::pair<int, int>> &buffer) {
57      int d = 1.25 - r;
58      int x = 0, y = r;
59      while (x <= y)
60      {
61          if (d < 0) {
62              d = d + 2*x + 3;
63              x++;
64              buffer.emplace_back(x + xc, y + yc);
65              buffer.emplace_back(-x + xc, y + yc);
66              buffer.emplace_back(x + xc, -y + yc);
67              buffer.emplace_back(-x + xc, -y + yc);
68              buffer.emplace_back(y + xc, x + yc);
69              buffer.emplace_back(-y + xc, x + yc);
70              buffer.emplace_back(y + xc, -x + yc);
71              buffer.emplace_back(-y + xc, -x + yc);
72          }
73          else
74          {
75              d = d + 2 * (x - y) + 5;
76              x++;
77              y--;
78              buffer.emplace_back(x + xc, y + yc);
79              buffer.emplace_back(-x + xc, y + yc);
80              buffer.emplace_back(x + xc, -y + yc);
81              buffer.emplace_back(-x + xc, -y + yc);
```

五、实验结果总结



这次实验遇到了两个坑，一个是刚开始 main 函数里面有一行中文注释，导致了运行报错，一个是在转换计算出来的点和实际的点的时候，应该先计算(-x ，y)(x，-y)等等，在把他加上 xc、yc，最后要写成这个样子(-x + xc, y + yc)(x + xc, -y + yc)，这里还有一个坑，+xc+yc 应该分别加在横坐标和纵坐标上面，而不是加在 x,y 上，如应该是(y + xc, x + yc)，而不是(y + yc, x + xc)。