**《软件工程》实验报告一**

姓名：吴怀文

学号：U201516993

班级：软工1501班

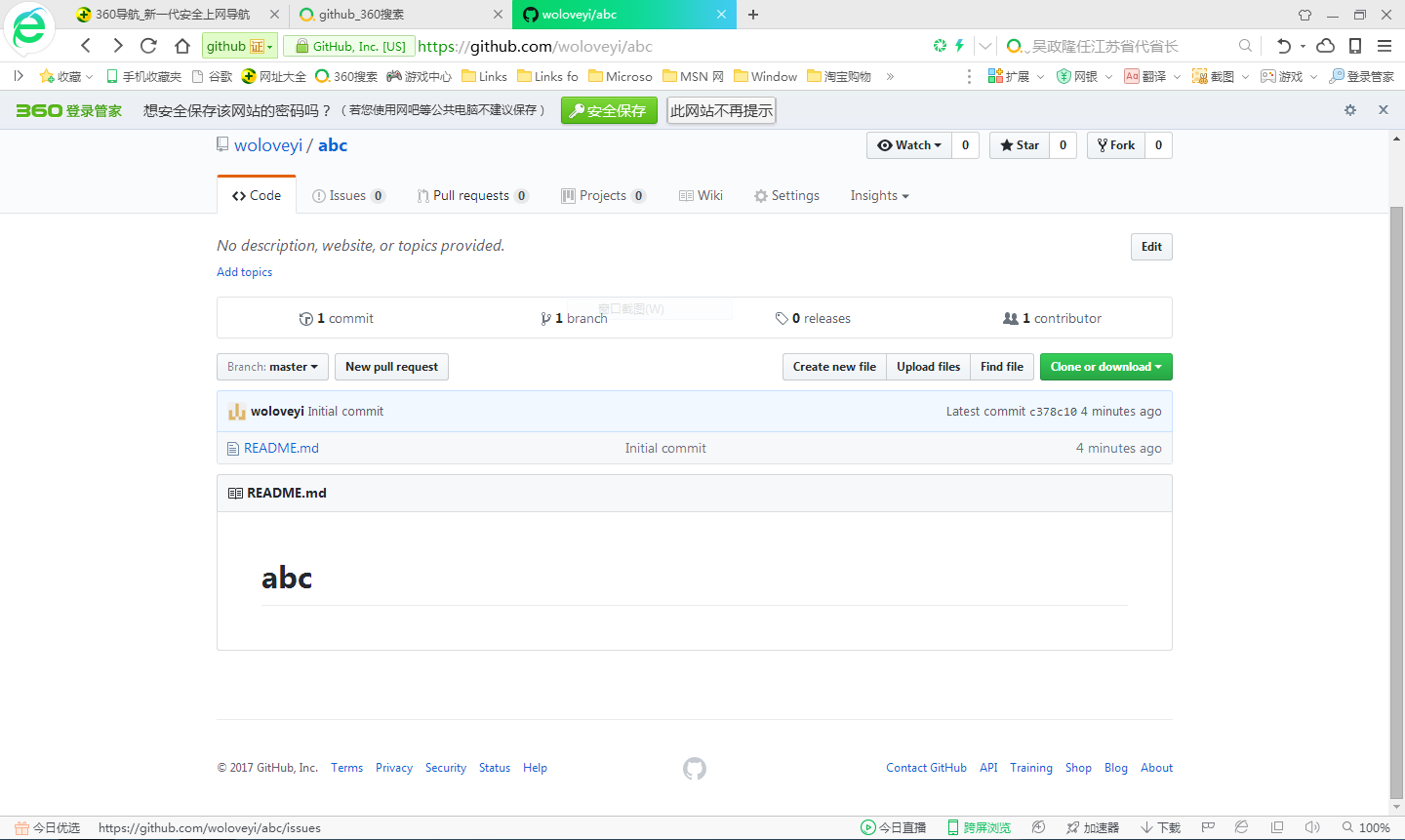
**Introduction：**

在一个边界为[-1, 1]的盒子里，给定m个半径r和位置mu可变的气球（它们不能重叠），找到最优的r和mu使得r^2之和最大。

**Algorithm：** 首先确定圆心遍历的步长（例如0.01），半径增加的步长（0.00001）和气球个数m。在给定区域内遍历所有点的位置，并依次增加步长，并确定在该处不会产生重叠的最大圆，取其中面积最大的圆并将其填入区域中。先确定第一个半径最大的圆，而后在剩下的空间中确定第二个半径最大的圆，直至所有m个圆填入区域中，此时m个圆的半径平方之和最大。

**Github**

[**https://github.com/woloveyi/abc**](https://github.com/woloveyi/abc)

****

**Structure and Test:**

typedef struct ball

{

double x; //圆心x坐标

double y; //圆心y坐标

double r; //圆半径

}Ball;

//用来储存已经放置的气球

typedef struct ballList

{

struct ballList \* next;

Ball ball;

}BallList;

void putBall() //关键函数，找到所剩空间里最大的球

{

Ball ball = {-1 + step, -1 + step, 0};

Ball maxBall = ball;

while (ball.x < 1)

{

ball.x += step;

ball.y = -1 + step;

while(ball.y < 1)

{

ball.y += step;

ball.r = 0;

double rstep = 0.1;

while(rstep > 0.00001)

{

if(ball.r > maxBall.r)

{

maxBall = ball;

}

ball.r += rstep;

if(!judge(ball))

{

ball.r -= rstep;

rstep /= 10;

}

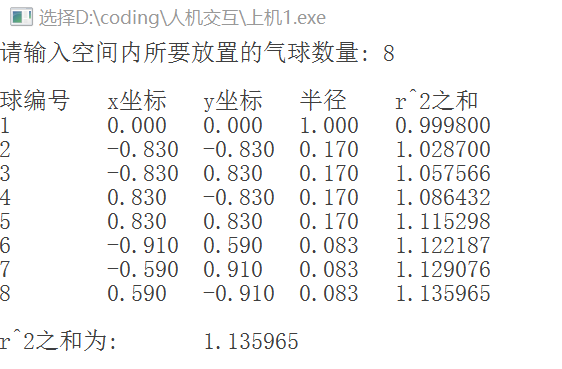
}

}

}

**Result**

（取m=8） ：



**Conclusion:** 由于步长的原因，得出的解并不是准确的解，但是在误差范围内可视为该问题的解。在给定边界的区域内，按顺序依次填入最大的圆，最后可使r^2之和最大。在该程序内，是通过遍历圆心得出最大圆的。代码思路简单但是过于暴力，运行开销比较大不算比较好的算法。