**南京理工大学计算机科学与工程学院**

**C++课程设计 报告**

**班 级 9191068403**

**学生姓名 孙傲歆**

**学 号 919106840333**

**起止时间 2020年9月1日-2020年9月23日**

**指导教师 衷宜**

**南京理工大学计算机科学与工程学院制**

目 录

[1 课题一：名称 1](#_Toc51771212)

[1.1课题功能 1](#_Toc51771213)

[1.2 开发平台 1](#_Toc51771214)

[1.3 总体方案 1](#_Toc51771215)

[1.3.1 总体流程图 1](#_Toc51771216)

[1.3.2 相关数据结构 1](#_Toc51771217)

[1.4 详细设计 1](#_Toc51771218)

[1.4.1函数 1](#_Toc51771219)

[1.4.2 main函数 1](#_Toc51771220)

[1.4.3 double pigsty::getprice()函数 1](#_Toc51771221)

[1.4.4 void buypig(pigsty ps[])函数 1](#_Toc51771222)

[…… 1](#_Toc51771223)

[1.5 测试与运行 1](#_Toc51771224)

[1.6 调试记录 1](#_Toc51771225)

[1.7 版本记录 2](#_Toc51771226)

[2 课题二：名称 2](#_Toc51771227)

[2.1课题功能及来源 2](#_Toc51771228)

[2.2 开发平台 2](#_Toc51771229)

[2.3 总体方案 2](#_Toc51771230)

[2.3.1 总体流程图 2](#_Toc51771231)

[2.3.2 相关数据结构 2](#_Toc51771232)

[2.4 详细设计 2](#_Toc51771233)

[2.4.1函数 2](#_Toc51771234)

[2.4.2 main函数 2](#_Toc51771235)

[2.4.3 bool CanMove(int moveid,int row,int col,int killid)函数 2](#_Toc51771236)

[2.4.4 virtual void mouseReleaseEvent(QMouseEvent\*)函数 3](#_Toc51771237)

[…… 3](#_Toc51771238)

[2.5 测试与运行 3](#_Toc51771239)

[2.6 调试记录 3](#_Toc51771240)

[2.7 版本记录 3](#_Toc51771241)

[2.8 分工情况 3](#_Toc51771242)

[3 心得体会 3](#_Toc51771243)

[4毕业要求达成度分析 3](#_Toc51771244)

# 1 课题一：养猪模拟

## 1.1课题功能

一款养猪场模拟游戏，可以实现基本的初始化、读入读出存档、买猪卖猪、输出猪场内各种信息、计算第一头猪被感染病毒后多久整个猪场的猪会全部被感染以及在猪被感染时实行合理的防控措施。

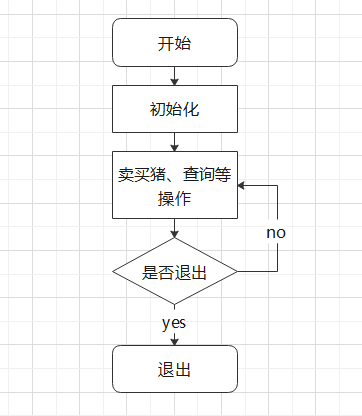
## 1.2 开发平台

操作系统：windows10

编译工具：Qt creator、Dev C++

## 1.3 总体方案

### 1.3.1 总体流程图



### 1.3.2 相关数据结构

struct pig

{

int species; //黑猪为1 小花猪为2 大白花猪为3

int id; //编号

int styid; //猪圈编号

int gday; //成长天数(一个月按30天计)

int infected=0; //被感染为1，没被感染为0

double weight; //体重

pig \*next;

};

class pigsty

{

private:

pig \*head=NULL;

int isblack=0; //猪圈为黑猪圈为1，非黑为0

int sum=0;

int isinfected=0; //猪圈有感染猪为1, 没有为0

int infectsum=0;

public:

int getspecies(int i); //获取种类信息

int getgday(int i); //获取成长天数信息

double getweight(int i);//获取重量

double getprice(); //获取价钱

void addpig(pig\*p); //加猪，主要用于初始化

void insert(int i,int j); //加猪，主要用于购买插入

void clear(); //出现感染猪，离开将该猪圈的猪全部清空

int getsumspecies(int i); //获取某种猪的数量

int getsum()；

int getisblack()；

int getisinfect()；

int getinfectsum()；

pig\* gethead()；

void setsum(int i)；

void sethead(pig \*p)；

void setisblack(int i)；//设置黑猪圈,1为黑猪圈，0为非黑猪圈

void setisinfect(int i)；

void infectgame()；

void addinfect()；

};

## 1.4 详细设计

### 1.4.1函数

int pigsty::getspecies(int i); //获取种类信息

int pigsty::getgday(int i); //获取成长天数信息

double pigsty::getweight(int i); //获取重量

double pigsty::getprice(); //获取价钱

void pigsty::addpig(pig\*p); //加猪，主要用于初始化

void pigsty::insert(int i,int j); //加猪，主要用于购买插入

void pigsty::clear(); //出现感染猪，离开将该猪圈的猪全部清空

int pigsty::getsumspecies(int i); //获取某种猪的数量

int pigsty::getsum()； //获取猪圈总猪数

int pigsty::getisblack()； //获取猪圈是否为黑猪圈

int pigsty::getisinfect()； //获取猪圈是否已被感染

int pigsty::getinfectsum()； //获取猪圈被感染猪的数量

pig\* pigsty::gethead()； //获取猪圈的头结点

void pigsty::setsum(int i)； //设置猪圈猪数

void pigsty::sethead(pig \*p)； //设置猪圈头结点

void pigsty::setisblack(int i)； //设置黑猪圈,1为黑猪圈，0为非黑猪圈

void pigsty::setisinfect(int i)； //设置猪圈感染状态，感染为1，未被感染为0

void pigsty::infectgame()； //猪圈第一只猪感染

void pigsty::addinfect()； //猪圈内部感染

int getallpig(pigsty ps[])； //获取猪场内猪总数

int getallinfectpig(pigsty ps[])； //获取猪场内感染猪总数

int getallpig1(pigsty ps[])； //获取猪场黑猪总数

int getallpig2(pigsty ps[])； //获取猪场小花猪总数

int getallpig3(pigsty ps[])； //获取猪场大花白猪总数

void buypig(pigsty ps[])； //购猪并入卷

void start(pigsty ps[])； //程序开始，猪圈初始化

void oneday(pigsty ps[])； //猪生长一天

double getsumprice(pigsty ps[])；//获取卖猪获得的总钱数

void searchsty(pigsty ps[],int i)； //获取i号猪圈的信息

void searchpig(pigsty ps[],int i,int j)；//获取i号猪圈的j号猪信息

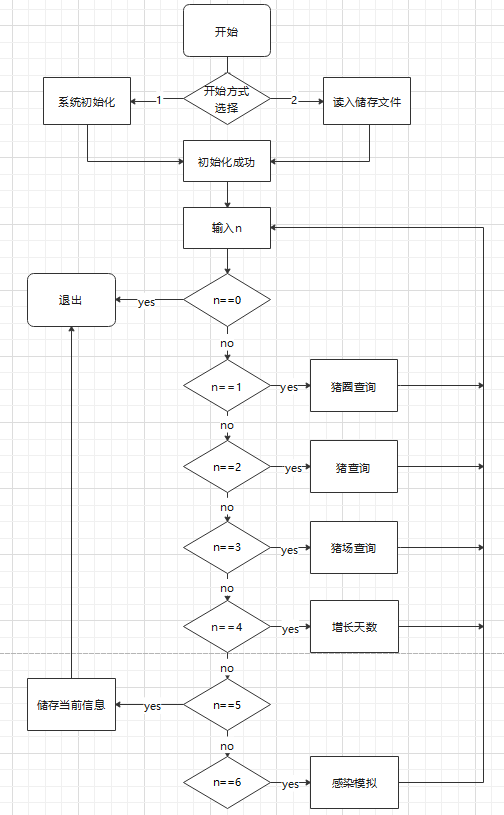
void printallsty(pigsty ps[])； //获取猪场信息

void fstart(pigsty ps[])； //文件载入

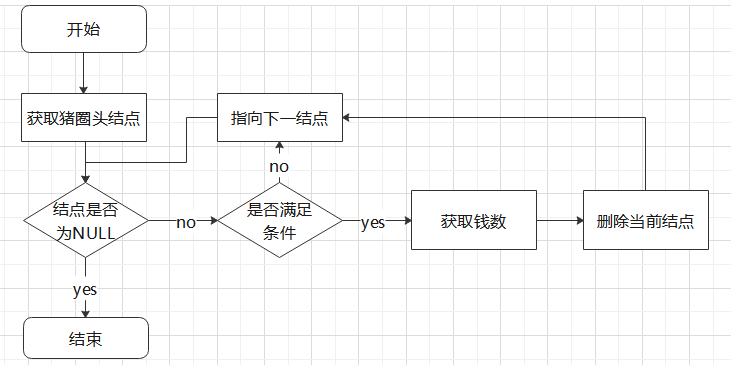
void fend(pigsty ps[])； //文件储存

void infectday(pigsty ps[])； //一天感染模拟

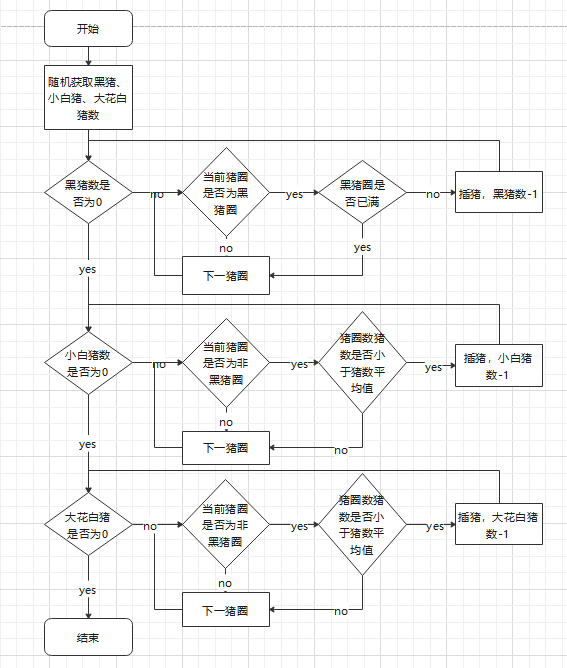
### 1.4.2 main函数



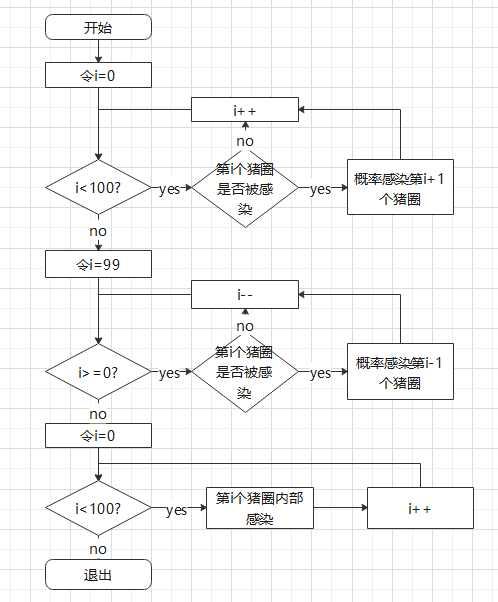
### 1.4.3double pigsty::getprice()函数



### 1.4.4void buypig(pigsty ps[])函数

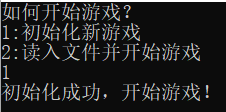


## 1.4.5void infectday(pigsty ps[])函数

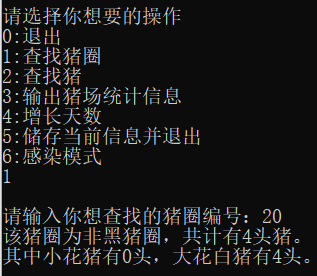


## 1.5 测试与运行

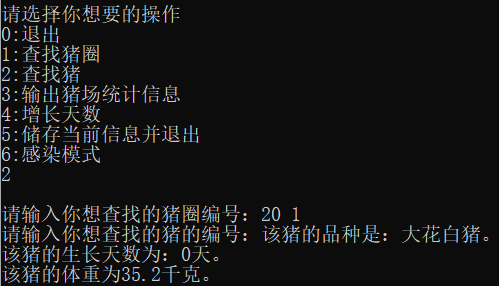
C++版本：

1. 

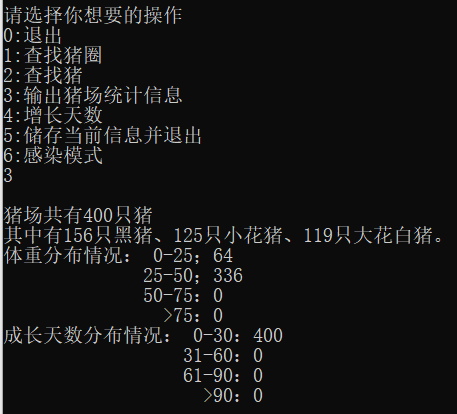
初始化

2. 

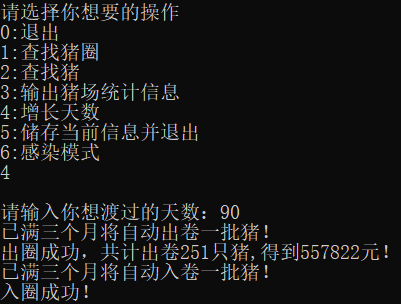
猪圈信息查询

3. 

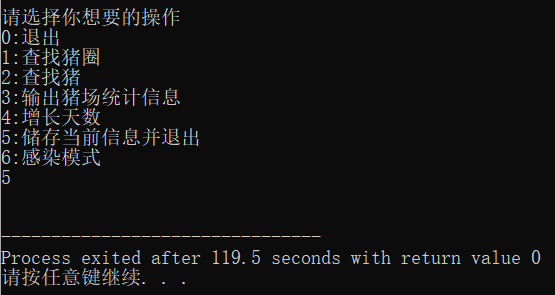
猪信息查询

4. 

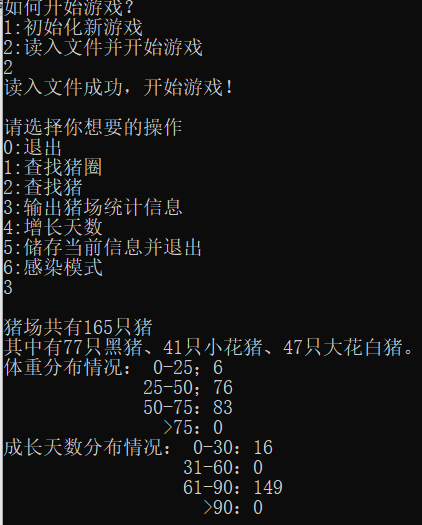
猪场信息查询

5. 

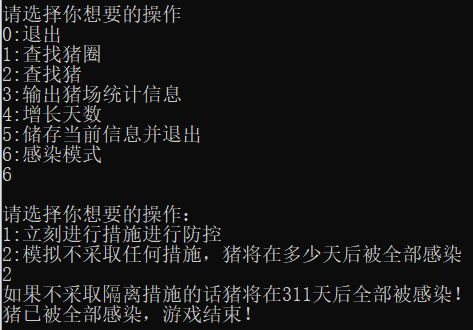
增长90天自动买卖猪

6. 

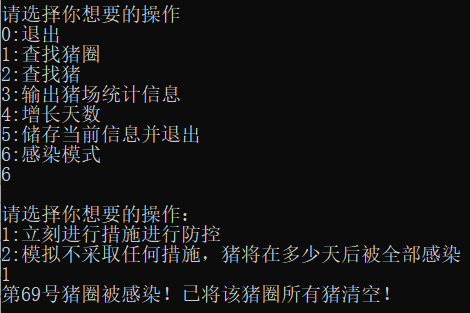
储存游戏

7. 

载入游戏

8.

模拟感染

9. 

随机感染

Qt版本：

1. 

猪信息查询

2. 

猪圈信息查询

3. 

猪场信息查询

4. 

卖猪

5. 

买猪

6.

随机感染

7. 

统计图

8. 

模拟感染

## 1.6 调试记录

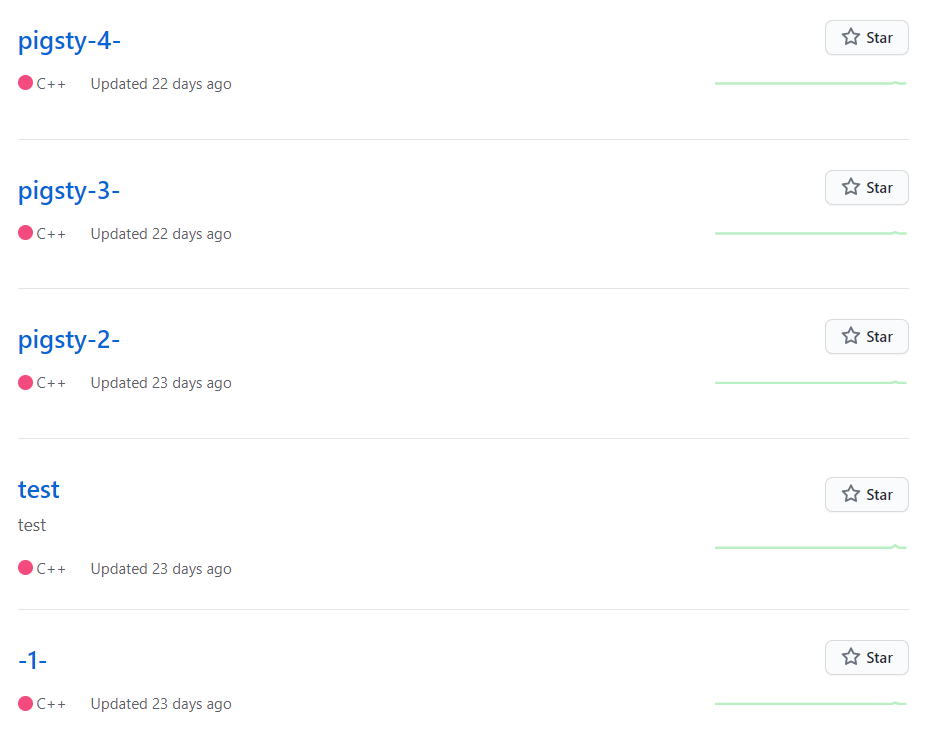
9月3日double pigsty::getprice()函数出错，当猪圈卖空时，无法成功卖猪且程序会异常退出，最后发现是因为少考虑了猪圈为空时的情况，找到问题后经修改解决了问题。

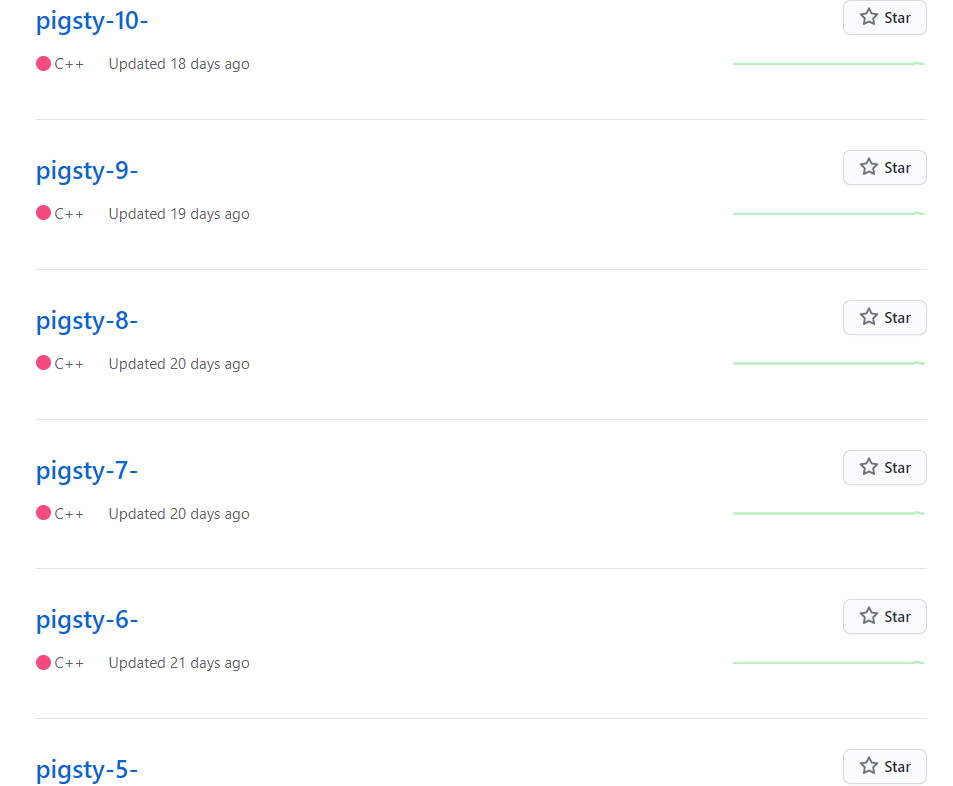
9月8日void infectday(pigsty ps[])函数出错，用该函数获取多少天猪圈会全部感染时其数据有时十分之大。原因是用rand((unsigned)time(0))获取时间算子数据的随机性无法得到保证，在网上查找各种资料后，将其改为srand(time(0) + rand())便解决了问题。

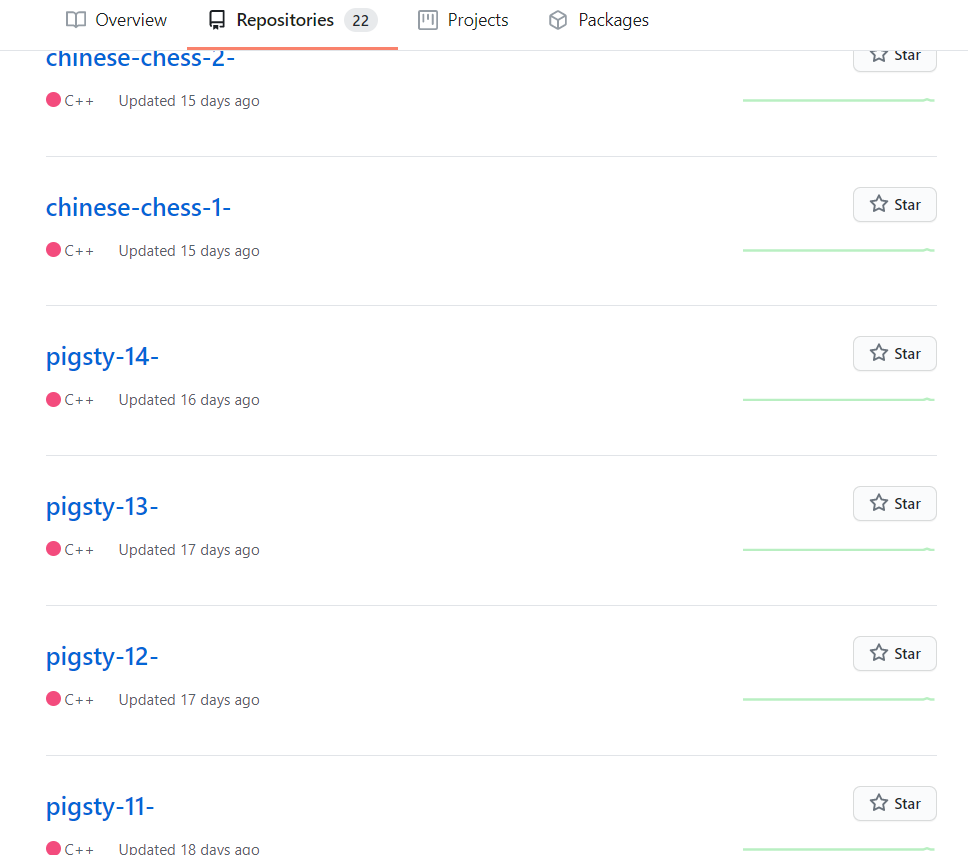
9月12日使用Qt绘制饼图发现图标太小，原因是图标layout没有指定当前窗口为父对象而导致错误，指定父对象后图标大小可以保证与窗口大小一致。

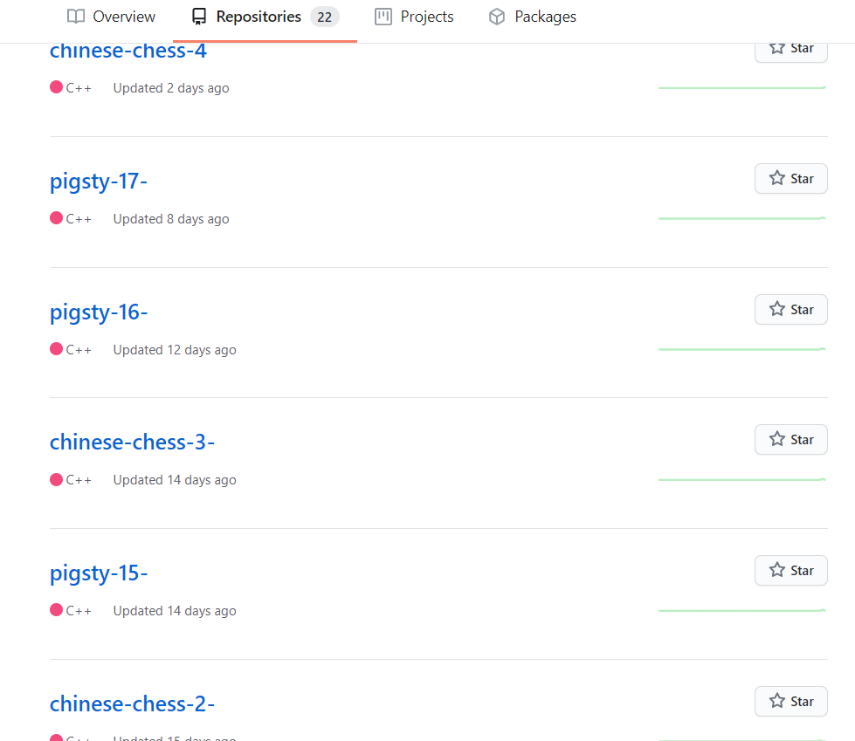
## 1.7 版本记录

网址: <https://github.com/twilight-mu?tab=repositories>









# 2 课题二：名称

## 2.1课题功能及来源

课题功能:本项目是基于Qt所创作的一款中国象棋游戏，并且能够同时实现单人对战和双人对战，并配备了悔棋功能。项目保证每个棋子都只能按照中国象棋的棋子走步规则来移动。且运用最大最小值算法和剪枝算法编写了一个较为简单的人工智能来保证单人对战游戏的进行。

来源: 项目来源于bilibili中的视频《C++课设 QT安装 [中国象棋]》(网址:https://www.bilibili.com/video/BV14t411V77f )

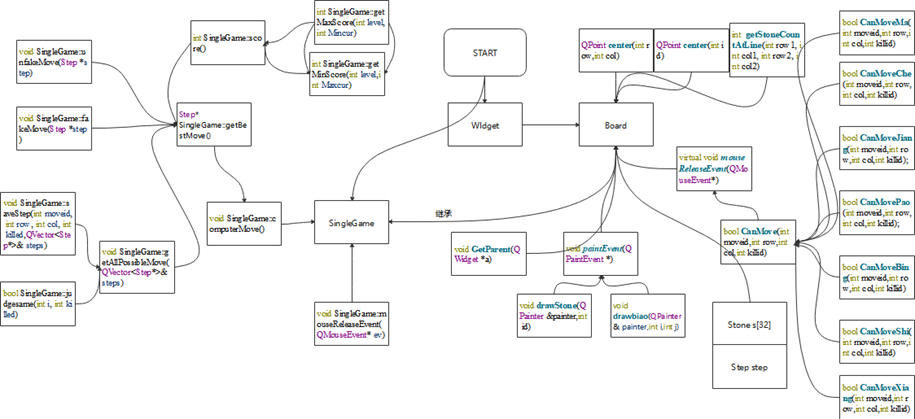
## 2.2 开发平台

操作系统：windows10

编程工具：Qt creator

## 2.3 总体方案

### 2.3.1 总体流程图



### 2.3.2 相关数据结构

Class：Widget，SinglGame，Board，Step，Stone

## 2.4 详细设计

### 2.4.1函数

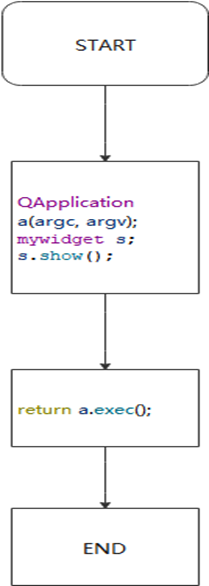
Class Board:

|  |  |
| --- | --- |
| 函数 | 功能 |
| QPoint **center**(int row,int col);//获取棋盘行列对应坐标 | //获取棋盘行列对应坐标 |
| QPoint **center**(int id) | 获取某一棋子的中心坐标 |
| bool **getRowCol** (QPoint pt,int &row,int &col); | 每个点定义一个鼠标获取区间 |
| void **drawStone**(QPainter &painter,int id) | 绘画棋子 |
| int **getStoneid**(int row,int col) | 获取某一棋子的id |
| int **getStoneCountAtLine**(int row1, int col1, int row2, int col2) | 判断选中的两点是否在同一行或同一列，且获取一行或一列上的棋子数 |
| bool **CanMove**(int moveid,int row,int col,int killid) | 棋子移动合法性判断 |
| bool **CanMoveJiang**(int moveid,int row,int col,int killid); | 将移动合法性判断 |
| bool **CanMovePao**(int moveid,int row,int col,int killid); | 炮移动合法性判断 |
| bool **CanMoveBing**(int moveid,int row,int col,int killid) | 兵移动合法性判断 |
| bool **CanMoveShi**(int moveid,int row,int col,int killid) | 士移动合法性判断 |
| bool **CanMoveXiang**(int moveid,int row,int col,int killid) | 象移动合法性判断 |
| bool **CanMoveChe**(int moveid,int row,int col,int killid) | 车移动合法性判断 |
| bool **CanMoveMa**(int moveid,int row,int col,int killid) | 马移动合法性判断 |
| void ***paintEvent***(QPaintEvent \*) | 绘制棋盘 |
| virtual void ***mouseReleaseEvent***(QMouseEvent\*) | 对鼠标释放事件作出相应的响应 |
| void **GetParent**(QWidget \*a) | this指针指向a指向的WIdget |

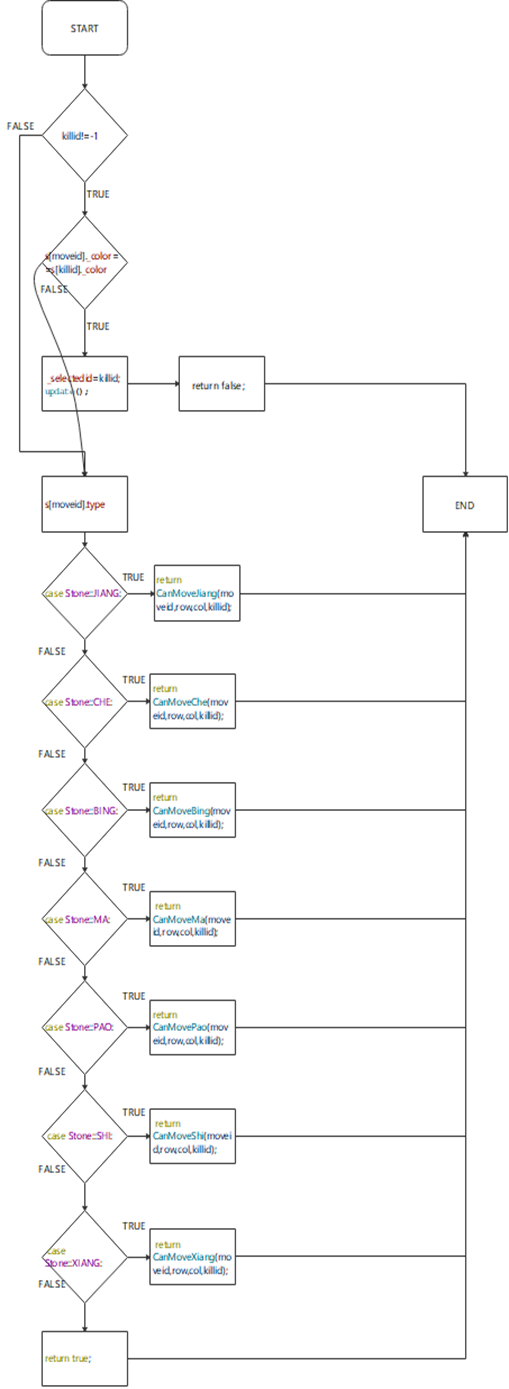
Class SingleGame

|  |  |
| --- | --- |
| 函数 | 功能 |
| void SingleGame::computerMove() | 电脑自主下棋 |
| void SingleGame::mouseReleaseEvent(QMouseEvent\* ev) | 鼠标释放事件作出满足要求的相应 |
| Step\* SingleGame::getBestMove() | 得到最优的步骤 |
| void SingleGame::saveStep(int moveid, int row, int col, int killed,QVector<Step\*>& steps) | 保存可能走的一步，并保存进入容器steps中 |
| void SingleGame::fakeMove(Step \*step) | 按照可能走的步，模拟走一步 |
| void SingleGame::unfakeMove(Step \*step) | 将模拟后的数据恢复原状 |
| int SingleGame::score() | 得到当前局面分 |
| int SingleGame::getMinScore(int level,int Maxcur) | 得到最小局面分 |
| int SingleGame::getMaxScore(int level,int Mincur) | 得到最大局面分 |
| bool SingleGame::judgesame(int i, int killed) | 判断当前棋子和死亡棋子是否为同一个棋子 |
| void SingleGame::getAllPossibleMove(QVector<Step\*>& steps) | 得到所有可能走的步，并存入容器steps中 |

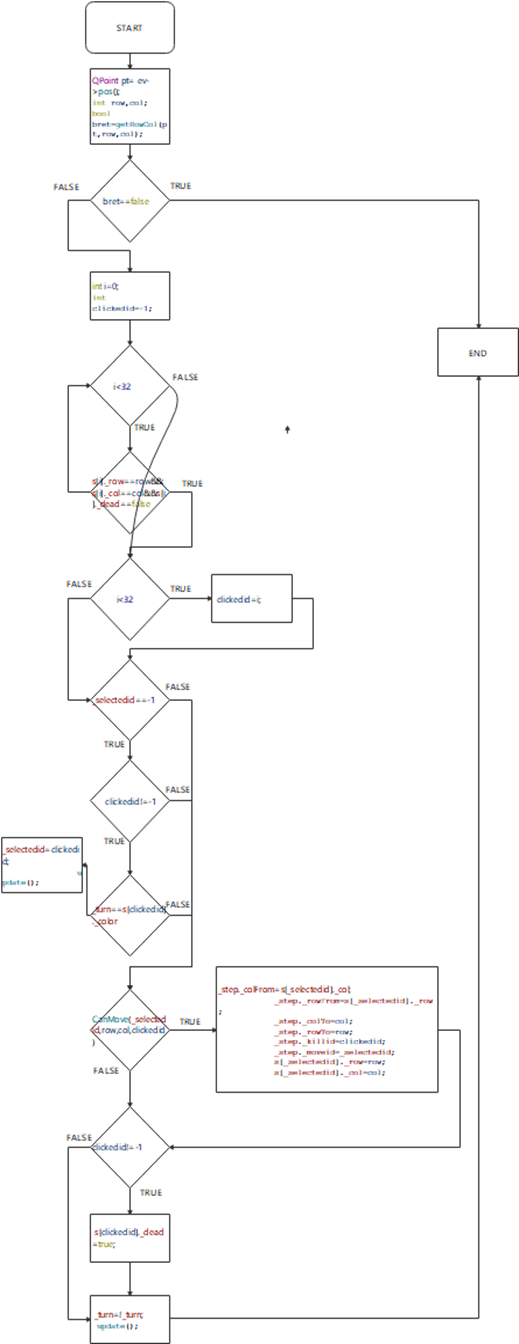
### 2.4.2 main函数



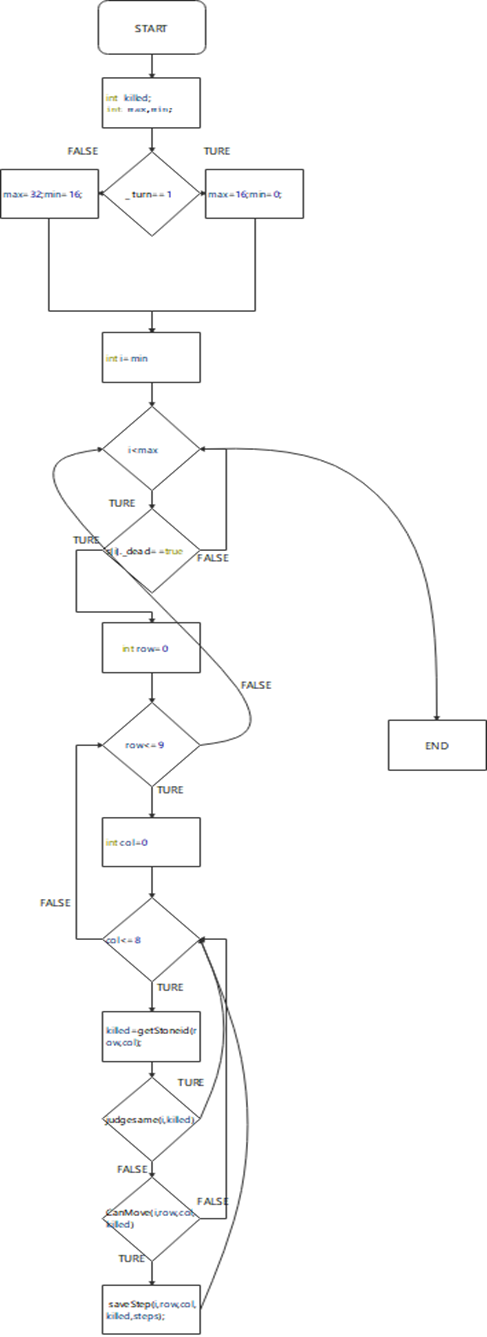
### 2.4.3 bool CanMove(int moveid,int row,int col,int killid)函数



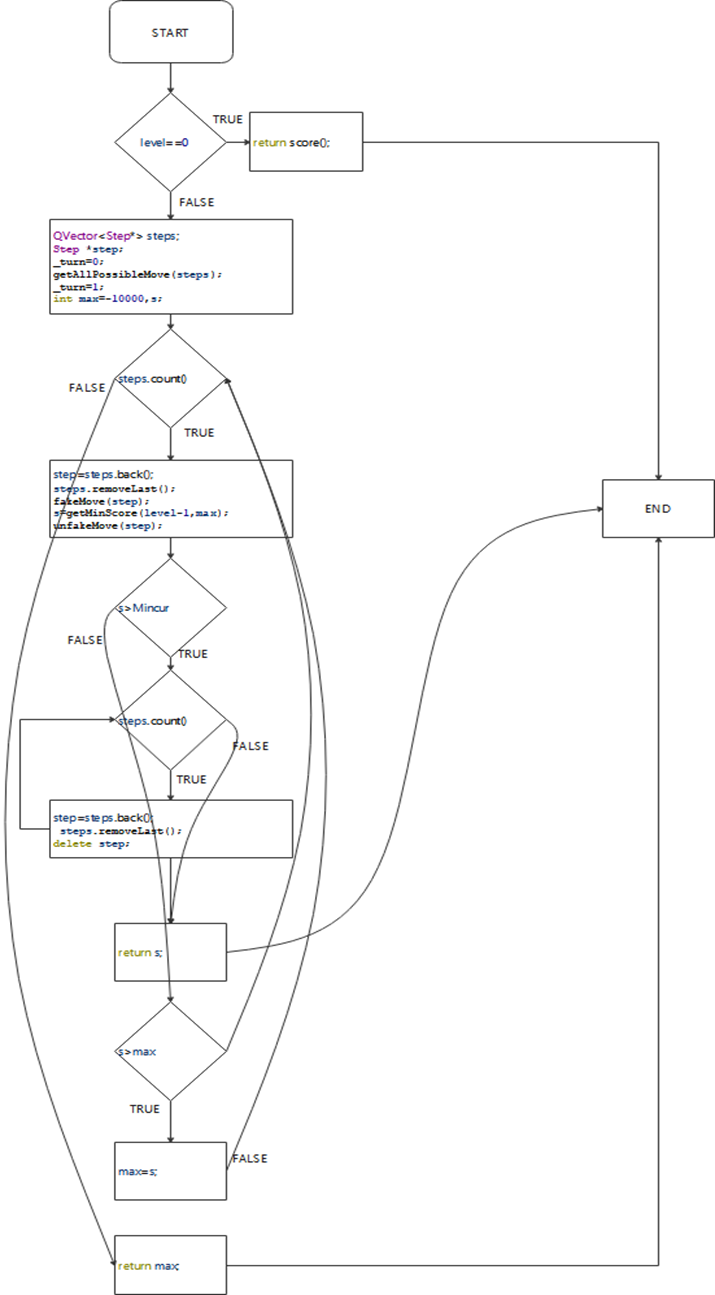
### 2.4.4 virtual void *mouseReleaseEvent*(QMouseEvent\*)函数



2.4.5 void SingleGame::getAllPossibleMove(QVector<Step\*>& steps)函数



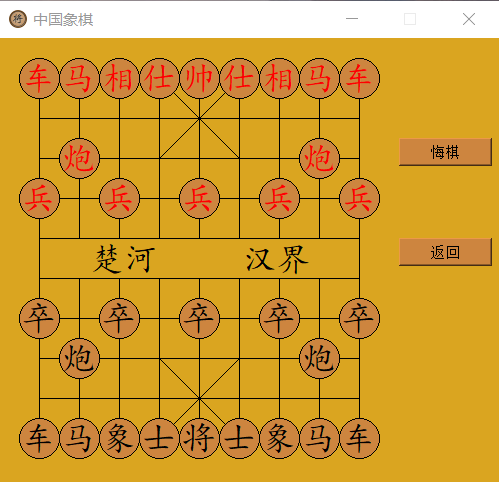
2.4.6 int SingleGame::getMaxScore(int level,int Mincur)流程图



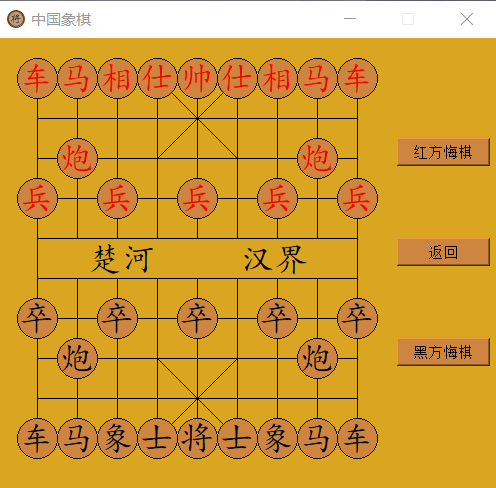
## 2.5 测试与运行

1. 

主界面

2.

单人游戏

3.

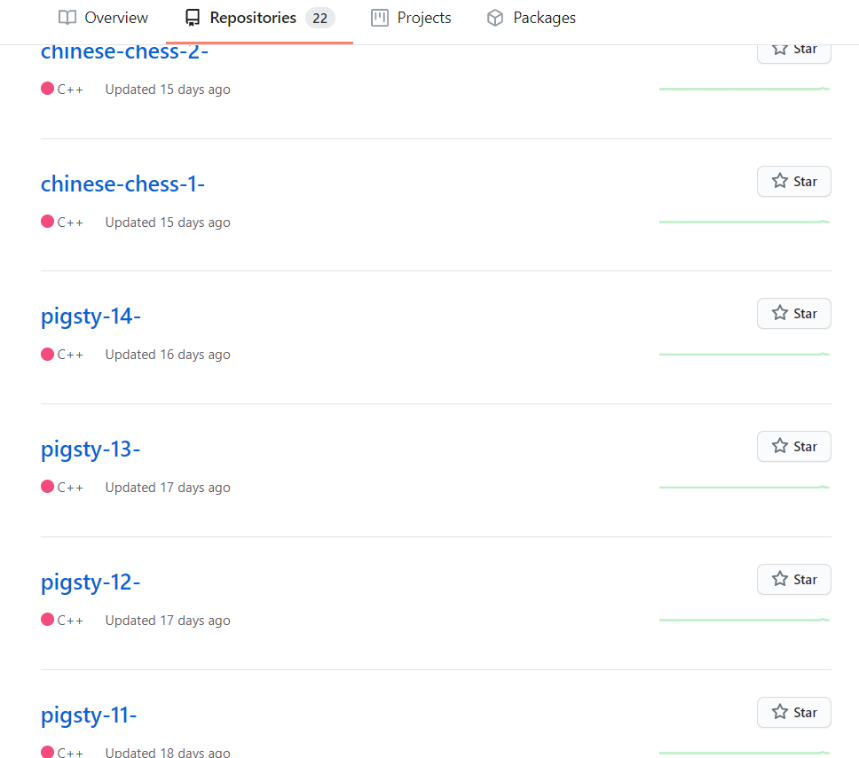
双人游戏

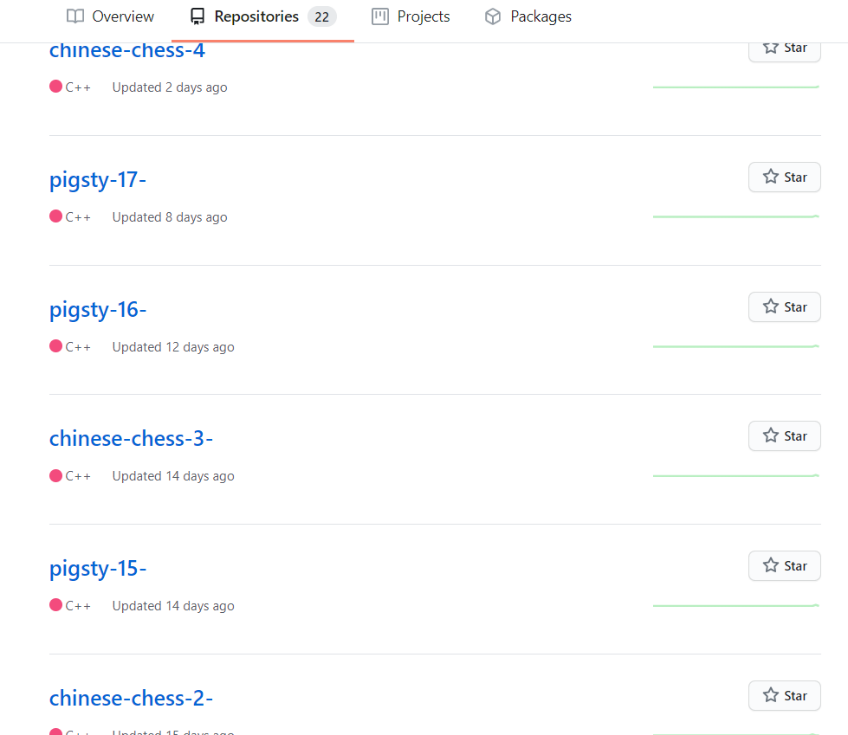
## 2.6 调试记录

|  |  |  |
| --- | --- | --- |
| 时间 | 内容 | 解决 |
| 2020.9.9 | bool Board::canMove函数出错，导致棋子在红棋走完一步后，黑棋无法再移动。 | 原因是函数前未加killed!=-1的判断，加入后程序正常运行，棋子可以正常移动了。 |
| 202.9.13 | 添加人机功能后，红棋进行第一步悔棋，黑棋棋子飞出棋盘 | 检查悔棋摁钮对应的lamda函数，修改step对s[i]的赋值，最后问题解决 |

## 2.7 版本记录

网址: <https://github.com/twilight-mu?tab=repositories>





## 2.8 分工情况

总代码量:1038行

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 班级 | 学号 | 分工情况 |
| 孙傲歆 | 9191068403 | 919106840333 | 棋盘绘制、棋子绘制、部分走棋规则(500行左右)；演示文档编写及展示 |
| 沈思宇 | 9191068403 | 919106840332 | 部分走棋规则、人工智能、悔棋功能(500行左右)；课设报告编写 |

# 3 心得体会

通过这次C++课程设计，我对“面向对象编程”这一概念有了更加深刻的了解。其次课程设计选题、构思、编码、调试、优化这一过程也让我受益匪浅，每当我解决一个问题，完成一项功能，有一种说不上来的自豪感，这或许就是编程的魅力所在吧！

而团队项目更是让我了解到了团队内部协调沟通的重要性。我与队友从讨论选题到编写代码、调试，两人一起携手合作，互相帮助。我们分工完成程序基本骨架的设计、函数设计、数据结构与算法的分析等步骤，在这些过程中总是有很多错误，但大部分是因为粗心导致的，所以设计程序一定要仔细、不容一点马虎。当然也遇到过许多大的问题，我们甚至一度遭遇到代码编写“瓶颈”。为了克服困难，我们两人一起到网上查找各类资料，一起讨论，有时甚至弄得焦头烂额。但这些困难最后通过我们的努力都被一一克服。或许程序设计的过程是枯燥无味且困难的，但当我们最后看到程序按照我们的预期运行时，心里还是美滋滋的，十分自豪且开心。

# 4毕业要求达成度分析

|  |  |  |  |
| --- | --- | --- | --- |
| **达成指标** | **具体的针对性方案** | **自评等级（1~5）** | **综合评定等级**  **（1~5）** |
| （2-3）能够针对计算机领域中的复杂工程问题进行需求分析和描述。 | 提高自己在计算机方面的知识，认真学习专业课。 | 2 |  |
| （5-1）针对复杂工程问题，了解与计算机专业相关的重要资料与信息的来源及其获取方法，能够通过图书馆、互联网及其他资源或信息检索工具，进行资料查询、文献检索，掌握运用现代信息技术和工具获取相关信息的基本方法。 | 好好利用学校的图书馆以及互联网上的各种资源。提高自己查阅资料，搜索有用信息的能力 | 5 |  |
| （8-2）能够在计算机领域工程实践中理解并遵守工程职业道德和规范，履行责任。 | 严格要求自己，提高自己的道德素质 | 5 |  |
| （9-1）具有团队意识，能够理解团队中每个角色的含义以及完成自己在团队中的充当角色的工作。 | 主动与他人交流，积极去融入团体，与团体内的成员共同进步、共同学习。 | 4 |  |
| （9-2）具备多学科背景知识，能够在多学科背景下的团队中与团队成员沟通，了解团队成员想法，并能够协调和组织。 | 好好学习基础学科以及专业课知识，提高自己的科学文化知识。 | 2 |  |
| （10-1）能够就计算机领域复杂工程问题与业界同行及社会公众进行有效沟通和交流，包括撰写报告和设计文稿、陈述发言、清晰表达或回应指令。 | 与同校同专业的同学们保持联系，以后步入工作岗位后可以进行沟通和交流。 | 3 |  |