中国科学 : 信息科学

SCIENTIA SINICA Informationis

高级 Java 程序设计——炸弹人项目报告

Developing a Java Game from Scratch

191220109 王珺

南京大学, 中国江苏省南京市栖霞区仙林大道 163 号
E-mail: 191220109@smail.nju.edu.cn

摘要 本报告为南京大学 2021 年秋季学期高级 Java 程序设计课程实验报告。本实验使用 Java 语言完成。通过运用一系列面向对象的方法，完成了一个具有进度保存、单机/多人联机的图形化炸弹人对战游戏。

关键词 课程设计，Java 开发，面向对象，单人游戏/多人联机，进度保存，图形化界面

1 开发目标

游戏灵感来源主要源于经典游戏泡泡堂，在原有游戏基础上，精简了道具功能并区分了人类玩家与怪物的攻击方式。

游戏简介：本游戏为基于二维平面实现的 Rougelike 炸弹人游戏，游戏支持单人游戏、多人联机对战。同时，游戏支持进度实时保存及进度恢复。

地图与玩家：人类玩家人数范围为 1-4，怪物个数为 8- 人类玩家人数。游戏开始时随机将 9 位人机玩家均匀放置在地图内部，并以所有玩家为中心扩展地块生成随机地图。

游戏玩法：人类玩家通过放置定时炸弹实现爆破墙面或攻击其它玩家或怪物。同时，处在爆炸范围内的玩家（包括安放此炸弹的玩家）也会在炸弹爆炸时受到伤害。每位玩家开局均只能同时放置 1 个攻击范围为 1 的炸弹，炸弹个数及攻击范围可以通过拾取道具提高，道具则通过击杀怪物掉落获得。怪物通过向距离自己最近的人类玩家不断移动，对人类玩家发起近身攻击。人类玩家受到怪物近身攻击时会还击（人类玩家被动还击伤害远低于炸弹爆炸威力）。人类玩家不可以主动发起近身攻击。

阵营：每名人类玩家分别自成一个阵营，所有怪物共同组成一个独立阵营。

操作方式：玩家接入服务器后，通过方向键 ↑ ↓ 选择新游戏或继续上局游戏，选定后按空格确认选择。注意：只有当本局游戏人数与上局存档的游戏人数一致时，才可选择继续游戏。进入游戏后，通过方向键 ↑ ↓ ←→控制角色移动，按空格键安放炸弹。

存档与恢复：游戏开始前，等待玩家连入服务器，在准备界面展示当前连入玩家人数。只有当连入玩家人数与上次游戏一致时，才允许继续上次游戏进度。否则只能开始新游戏。游戏过程中，服务器实时获取当前信息并存档。存档恢复时，服务器通过读取对上局游戏最后状态的存档文件，恢复线程池，进而恢复上局游戏中断前的游戏状态。

游戏道具：怪物死亡时，会有一定概率爆出游戏道具。道具仅可供人类玩家拾取，对怪物无效。同时，道具会因炸弹爆炸波及而消失。道具分三种，道具 1（红色心形）可为玩家增加 10 点生命，道具 2（黑色圆点）可使玩家允许安放的最大炸弹数量增加 1，道具 3（黄色十字）可使玩家安放炸弹的最大攻击范围增加 1。

胜利条件：人类玩家通过炸弹爆破、被动近身攻击，对其它人类玩家或怪物发动攻击，减少对方血量。当场上仅剩 1 位人类玩家且所有怪物被清理完毕时，存活的人类玩家即为胜利者。若所有怪物被清理完毕前已无人类玩家存活，则怪物阵营胜利。若人类玩家与怪物全部死亡，则最后死亡的玩家即为胜利者。

图 1　准备界面

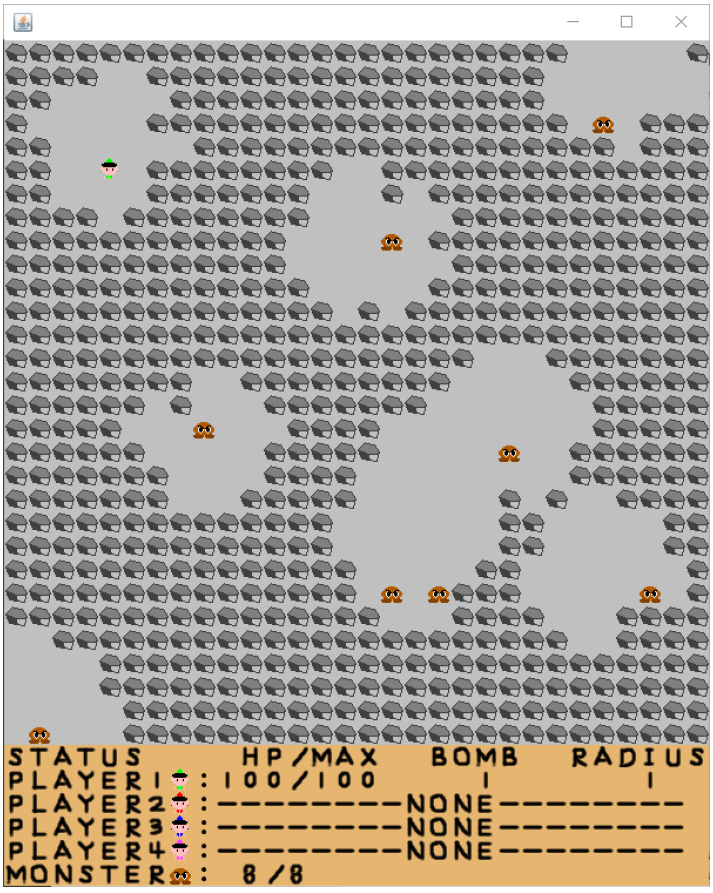

图 2　游戏界面

2　游戏流程

玩家连入：界面展示当前连入的玩家人数，最多连入 4 人，最少 1 人。

准备开始：所有玩家共同选择游戏模式，只有当连入人数与上次游戏人数相等时，才可选择继续游戏。否则只能选择新游戏。

游戏开始：在界面底部展示当前所有人类玩家的状态（血量、可同时安放的最大炸弹个数、炸弹爆炸范围）及怪物的剩余数量。

新游戏：在地图上均匀放置 9 名人机玩家，其中人类玩家的位置按编号唯一确定。以玩家位置为中心扩展地图范围，以达到随机生成地图的目的。

继续游戏：恢复上次游戏中止时的地图状态（墙壁位置、玩家位置、炸弹位置）及玩家状态（包括血量、可同时安放的最大炸弹个数、爆炸范围及怪物的剩余数量）等。

游戏过程：人类玩家通过上述游戏玩法中的说明进行移动和攻击操作，怪物在无法接触到任何玩家时采取随机游走策略。当视野内有玩家出现时，选择距离最近的玩家接近。

游戏规则：同上述游戏玩法说明。

玩家操作：使用方向键移动，空格键安放炸弹。当血量归零时，无法继续操作，需等待至游戏结束。当最终胜者产生时，所有玩家停止移动，界面底部展示胜利者信息（x 号玩家胜利或怪物胜利）。

游戏存档：游戏中，服务器自动对存档文件进行实时刷新，直至服务器停止工作。继续游戏时，服务器通过读取存档文件内容，恢复游戏中止前的地图状态与玩家状态，以恢复中止前的游戏状态。

3 面向对象设计

总体上采用了面向对象设计方法，从现实角度出发，尽可能做到贴近实际的对象交互模式。

实验框架源于实验 4 迷宫游戏，在其基础上，将原有的单人迷宫游戏改造为多人（含电脑玩家）炸弹人对战游戏。

游戏界面：源于对实验 4 迷宫游戏中 Screen 与 WorldScreen 类的重用与修改。界面中字符的显示方式复用 AsciiPanel 中提供的方法，在此基础上，使用像素画图软件将对应字符重置，实现界面的图形化。

地图类：通过地图生成器类（MazeGenerator）加载随机地图。除玩家与炸弹外，地图上的每个地块有地板和墙壁两种状态。生成地图时，首先默认地图上所有地块状态均为墙壁。随后获取玩家列表及玩家位置，以每个玩家为中心，向四周随机扩展一定范围区间的地块，将包括玩家位置在内的扩展范围地块状态修改为地板。

玩家类：分为人类玩家（Calabash）与怪物（Monster），两类对象均继承自 Creature 类。两类玩家均具有血量、攻击力等基本属性，同时可以在地块内进行移动。在此基础上，人类玩家还具有玩家序号、炸弹个数、攻击范围等属性。人类玩家同时安放的炸弹个数不能超过限制。怪物死亡时，有一定概率生成游戏道具。由于人类玩家可以通过安放炸弹攻击其它玩家或爆破墙面，故需要定义炸弹类。炸弹类具有攻击力、攻击范围、爆炸倒计时、所属玩家等属性。怪物攻击、炸弹爆炸均需要通过 BFS 算法获取移动路径或杀伤范围。

炸弹类：人类玩家可安放定时炸弹（Bomb），安放后，启动炸弹定时器。当倒计时结束时，炸弹爆炸，根据爆炸范围及 BFS 算法，同时考虑墙壁的阻断效应：即炸弹冲击波遇墙壁后停止扩散，进而得到最终的爆炸范围。爆炸范围内，所有炸弹即刻停止计时并与已爆炸弹产生连爆效果。爆炸后，范围内所有墙壁变为地板，所有道具消失，所有玩家血量按炸弹伤害减少。随后炸弹消亡。

寻路算法类：采用广度优先搜索算法（BFS）对怪物移动、炸弹爆炸算法加以优化。怪物移动时，通过 BFS 搜索视野范围内最近的人类玩家。若有，则根据算法得出的路径并不断接近人类玩家。若

无，则随机在可移动的方向中任选一个进行游走。炸弹爆炸时，通过 BFS 搜索爆炸范围对应的所有地块。若遇墙壁，则爆炸最多波及一层墙壁。除墙壁外，其它物体（如玩家、道具等）不可阻止炸弹爆炸范围扩展。炸弹爆炸范围内的玩家会被扣除相应血量，爆炸范围内的道具会消失。

物品类：包括物品类（Thing）自身及在其基础上派生的多个类，如字符类（Character），地板类（Floor），墙壁类（Wall）等，用于在图形化游戏界面上展示相应内容，包括墙壁、地板、各类玩家、炸弹、爆炸效果、游戏道具及字符等。

文件类：通过文件类（File）执行一系列文件读写操作，在游戏进行过程中，服务器会在任意玩家申请刷新屏幕时，将当前每位人类玩家状态（包括位置、血量、炸弹数、最大炸弹数、攻击范围），所有怪物状态（包括位置、血量），所有炸弹状态（包括位置、攻击范围、所属玩家的玩家号、剩余爆炸时间）存入文件中，以实现实时刷新记录文件的目标。当下局游戏玩家选择继续游戏时，可通过读取记录文件恢复线程池状态，以达到继续上次被中断游戏的目的。

网络类：包括服务器（Server）与客户端（Client），两者通过网络接口实现数据交互，以完成数据传输与回显。其中，客户端可以向服务器发出三种请求，玩家刚接入时服务器时会发出 0 号请求，向服务器申请玩家编号号。服务器收到 0 号请求后，会增加 1 名人类玩家，并将分配到的玩家编号传回客户端。游戏开始后，客户端不断向服务器发出 1 号请求，服务器收到 1 号请求后，会将当前界面处理为像素及颜色信息并回传给客户端，由客户端交付客户端显示类（PlayerScreen）执行打印操作。当玩家执行键盘按键操作时，客户端发出 2 号请求，并在请求号后附上玩家编号及按键信息，由服务器交付服务器显示类（WorldScreen）进行合法性判断。若合法，则会执行按键对应的用户操作。

显示类：以接口类屏幕（Screen）为基础，在其上派生出服务器显示类（WorldScreen）与客户端显示类（PlayerScreen）。其中，WorldScreen 类负责执行服务器端的操作，如处理服务器接收的用户按键信息、选择游戏模式、统计接入玩家人数、显示提示信息、显示玩家状态、显示游戏状态、创建/恢复地图、生成玩家/恢复玩家状态、恢复炸弹与道具状态等。PlayerScreen 类负责收集客户端接收到的屏幕像素及颜色信息，将其解码为图形界面并打印在用户界面上。

图形化界面类：使用作图软件画图，将原有 AsciiPanel 中的符号表图替换为图形化的图片。完成作图后，将 Main 类及 Client 类中原本引用的 talryth_square_15x15.png 图片资源替换为自制像素图（My_UI.png）。随后，修改 AsciiPanel 类中的 paint() 函数，将涂色函数注释，保留图片原色，以实现图形化界面的显示。

4 设计理念

游戏开始前，所有玩家共同选择游戏模式。只有当当前玩家人数与存档一致时，才可选择继续游戏。否则，只能开始新游戏。

游戏开始后，玩家选择将由 WorldScreen 进行处理，选择新建地图与玩家，或按存档恢复地图与

线程池。随后游戏开始，玩法与规则见上述说明。

当出现以下状态之一时，游戏结束：
1、人类玩家全部死亡，怪物方获胜；
2、怪物全部死亡，人类玩家仅剩一名，该名玩家获胜；
3、怪物与人类玩家均全部死亡，最后死亡的一名人类玩家获胜。

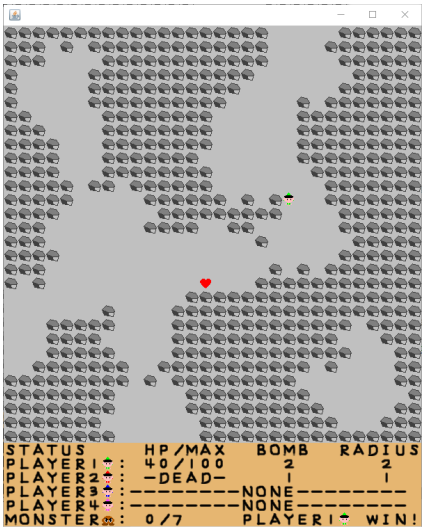

图 3　玩家 1 胜利

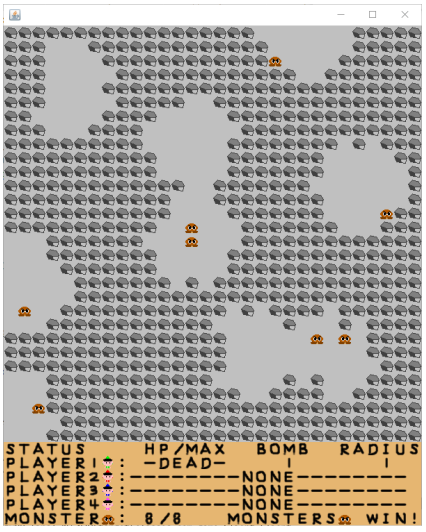

图 4　怪物胜利

程序的主要流程由 Main 类进行调度，包括服务器、游戏线程在内的主要类均定义在 Main.java 文件中。故在本质上，Main 类相当于一个服务器与调度器的集合，其主要通过 Channel 与 Client 类

产生信息交互。另外，Main 类也可以向 WorldScreen 发送命令或将用户的请求交付执行。

WorldScreen 是仅次于 Main 的主要部件，负责存储世界信息及屏幕上每个地块对应的内容。同时，在 Main 的控制下，WorldScreen 类同时负责线程的创建与恢复，一些访问控制也由 WorldScreen 类完成，如判断用户键盘操作的合法性等。另外，由于所有的线程均由 WorldScreen 直接调度与控制，故 WorldScreen 类也同时需要对地图资源的访问权限加以控制，避免并发冲突的产生。

交互逻辑主要由 Main 中的 Server 与 Client 共同实现，通过使用 NIO Selector，实现服务器与客户端之间一对多的信息传输。客户端的功能为：回显服务器显示状态，同时接收用户的键盘操作并将信息处理后打包发送给服务器。除此上述的显示与人机交互功能外，游戏的其余功能均在服务器中实现。

显示框架源于实验 4 迷宫游戏中的 AsciiPanel 类，通过读取源图片并对其按块进行切割，实现图片内容在程序运行界面中的展示。在此基础上，使用画图软件编写新的源图片以实现用户界面图形化。

游戏的存档与恢复功能也是在 WorldScreen 与 Main 的共同控制下实现的。游戏过程中，服务器实时保存当前的游戏状态。待需要恢复时，服务器通过读取存档文件中的信息，将相应属性赋予对应的对象，并在线程池中启动新线程，以实现模拟原有线程的目标。

5 正确性验证

5.1 单元测试

在 AppTest.java 文件中编写单元测试函数，尽可能全面地对各类中的有关函数进行测试。编写完毕后，通过在终端中执行 mvn test 命令，完成单元测试。无误后，使用 vscode 下的 Coverage Gutters 扩展，对各个类进行覆盖率分析，最终确保了对所有代码文件（接口类除外）的测试覆盖率均不低于 50%（平均约为 80%）。

```
[INFO] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.885 s - in com.game.AppTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.4:report (post-unit-test) @ myprj ---
[INFO] Loading execution data file D:\jwork\jw08-final-KW-NJU\target\jacoco.exec
[INFO] Analyzed bundle 'myprj' with 24 classes
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.4:check (check-unit-test) @ myprj ---
[INFO] Skipping JaCoCo execution because property jacoco.skip is set.
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  7.190 s
[INFO] Finished at: 2021-12-30T00:16:54+08:00
[INFO] ------------------------------------------------------------------------
```

图 5 单元测试

6 并发测试

编写服务器与客户端的交互逻辑，通过同时展示不同客户端的窗口，检验网络传输与并发逻辑是否正常。

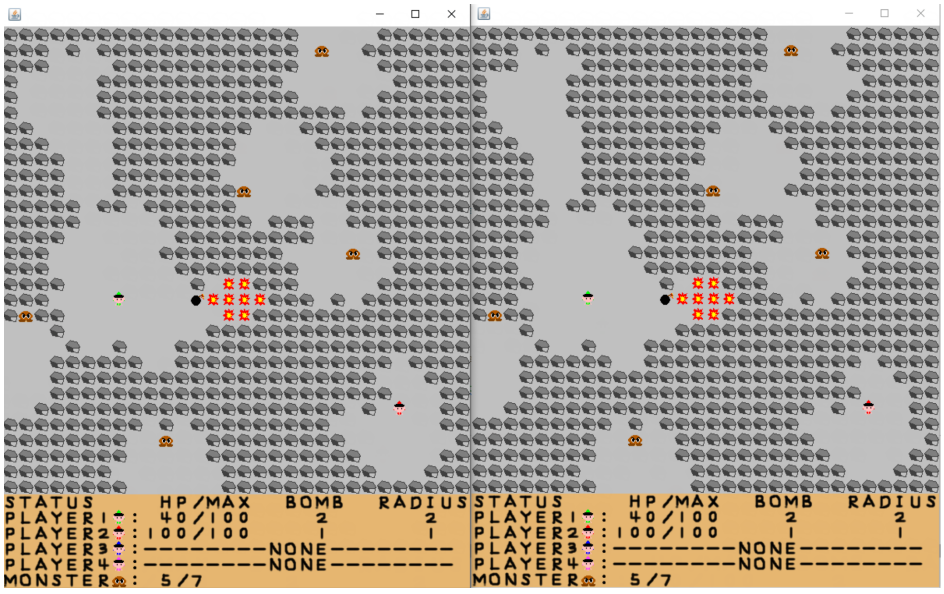

图 6 并发测试

7 技术问题 &工程问题

7.1 文件读写

起初，在尝试编写存档文件时，对文件形式产生了一系列的思考，如存储内容、存储格式等。最终决定以字符串格式来依次存储，并通过存储标记位字符来对各类型内容的条目数量加以标识。
在后续尝试通过读取文件恢复线程池时，又遇到了读取失败的原因。经断点调试与核对输出结果后，最终顺利定位到了问题。经修改代码执行顺序并对较长数据进行特殊处理后，顺利完成了通过文件读写实现游戏进度存档/恢复的功能。

图 7 加密的日志文件

7.2 线程恢复

游戏存档功能在本质上是对线程池状态的恢复。由于程序中断会导致原有线程全部被强行停止，故为完成线程恢复的任务，必须通过实时记录原有线程池状态并在需要恢复现场时，将记录的参数交由新线程来实现。由于本实验中玩家可以安放的炸弹个数有限，故若出现游戏中断时场上存在炸弹尚未爆炸或正在爆炸的情况，会导致恢复现场时玩家可安放的最大炸弹个数出现误差。为解决上述边界条件触发的问题，此处将炸弹与玩家关联。对尚未爆炸的炸弹，恢复其状态，待其正常爆炸。后续处理操作同游戏未中断时。对已经爆炸的炸弹，考虑其即将消亡且已对其爆炸范围内的一切物体产生了影响，故计算时将其视为已经消亡。通过编写上述逻辑，顺利解决了边界条件触发的逻辑问题。

7.3 网络传输

根据课程 PPT 中给出的示例编写了使用 NIO Selector 的网络对战逻辑。在实现了服务器和客户端的交互后，尝试互传信息。发现传输的内容始终含有预期以外的内容。经多次输出及修改相关代码测试，发现输出的内容始终不符合预期且始终不变。后经阅读代码，发现在编写传输逻辑时，忘记删除了原有的部分测试代码，导致 buffer 中的内容始终以某个预期之外的固定数字开头。删除该部分冗余代码后，问题解决。

7.4 单元测试

在编写网络对战部分代码之前，单元测试已经编写完毕且各文件（不含接口类）中的代码覆盖度均超过了 50%。但在网络对战部分代码编写完成后，尝试重新执行单元测试代码却无法正常结束。经检验，发现问题源自服务器与客户端的交互逻辑。由于服务器开启后默认人类玩家数量为 0，故无法执行后续操作。因此，需要通过添加代码手动模拟玩家连入的过程。随后，测试代码运行正常且结果无误。

另外，由于实现网络对战需要编写新的功能或类，导致个别文件中代码覆盖度由原有的 50% 以上降为了 40% 左右。通过添加新的测试用例，顺利使所有文件（不含接口类）中的代码覆盖度达到了原有要求。修改后的总体代码覆盖度为 75% 左右。

```
@Test
void testbomb() {
    WorldScreen w = new WorldScreen();
    w.addplayer();
    w.newgame();
    Bomb bomb = new Bomb(w.getplayer()[0], w.getworld(), 1, 0, 2000, 300);
    assertEquals(0, bomb.getstate());
    assertEquals(0, bomb.getplayernum());
    assertEquals(1, bomb.getradius());
    assertEquals(300, bomb.getmilsec(0));
    bomb.setstate(1);
    assertEquals(0, bomb.getsec(0));
    bomb.setmaze();
}
```

图 8　修改后的部分测试代码

7.5 图形化界面

完成大部分工程要求的内容后，开始对工程进行图形化。起初，对 AsciiPanel 部分代码理解存在问题，导致迟迟无法完成图形化过程。经与同学讨论并学习像素画图后，顺利编写了自己的贴图。使用贴图替换原有图片并注释 AsciiPanel 中与涂色功能有关的代码后，顺利完成了图形化过程。

图 9　替换 AsciiPanel 的贴图

8 课程感言

本学期的 Java 课程与实验，对我而言总体上是收获满满的。就课程实验而言，它让我在为了完成目标而付诸实践与努力的同时，也收获了满满的成就感。

一直以来，我个人喜欢偏重实验 + 实践类型的课程胜于偏理论类课程。因为将课上所学的内容付诸实践的过程，对我而言本身就是收获快乐的过程。虽然，在课程的理论学习与实验实践开发过程中，我都曾遭遇过无数瓶颈，但能够通过潜心学习钻研理论，学习开发与调试技巧，最终克服一个个“技术难关”，得到越发完善的结果，开发出可玩性越来越高、界面观感越来越强的游戏，对于我这样一个极度信奉完美主义的人而言是一种十分正向且有效的精神激励。

学期开始前，我也对这门课程的难度有所耳闻。平心而论，当初选修高级 Java 程序设计课程的目的仅仅是为了能学会运用并熟练掌握一门新的高级程序设计语言。但显然，事情绝非如我事先想象那样简单。正如曹春老师在课上所说，这门课程的内容并不仅限于学习一门新的程序设计语言。学习的目的，也不能仅仅拘泥于掌握基本的面向对象写法。课程的侧重不应在“Java”，而应在“高级”。比单纯的掌握一门语言更重要的是能够以 Java 语言为基础，通过一学期的课程与实验掌握更多的高级技能。

“从客观事实出发进行面向对象设计，模拟现实世界中事物的交互流程”是老师从课程之初一直强调的内容。自此以后，我也在有意识地尝试在本课程的实验或其它有类似需求的项目中贯彻这一理论，同时尽可能矫正以往并不规范的编码习惯。但毕竟一学期的时间十分有限，我并无可能在短时间内完全掌握这一要义的精髓。今后我也会一直努力向这一要求靠拢。

众所周知，改变往往意味着未知与风险。正因如此，我比较不愿意打破自己原有的学习习惯与生活节奏。但无疑，这门课程是促使我努力尝试走出自己舒适圈的一大关键因素。由于方向选择的缘故，我在本学期的学习压力总体较小。但这门课程的存在使我不敢懈怠。老师的督促与身边一直努力的同学们不断激励我向着更好更优秀努力。这一过程令我受益匪浅。

一学期的课程使我真实地认识到了自己与强者之间的巨大差距，比如在面向对象编程规范、熟练度、代码阅读能力及游戏引擎的使用上。正因如此，我注定无法成为本学期在这门课程中表现最优秀的学生。但经过了一学期的理论知识学习与实验，我获得了于我而言的长足进步。一学期的课程使我在学习了很多有用知识的同时，收获了更多的编程经验。

一学期的课程就要结束了。当我写下这句话的时候，我的心情是复杂的。我既为时间流逝之快、为毕业将至而感慨，也为自己能坚持到最后并按期完成了这门课程的所有实验内容而骄傲。这门课程的存在，让这个略显无聊且艰难的学期变得更有意义。同时，学习新知识、出色且按时完成实验内容的过程所带给我的成就感，也给了我不断支撑下去的动力与理由。或许当我这样想时，成绩就已经不再是最重要的东西了吧。

最后，感谢一学期以来曹春老师出色的课堂教学以及在其它方面对我的支持与包容，感谢各位同学的陪伴与帮助。这门课程将注定成为我本科四年学习生活中最难以忘记的几门课程之一。

祝愿本课程在未来的开展越来越顺利，也祝愿曹春老师工作顺利，家庭幸福。也希望我在未来能保持这份热情，不断实现自我超越。

9 致谢

感谢下列同学在本学期的课程学习及项目开发过程中给予我的学业帮助与精神支持

191220016 陈致远

191220072 卢润邦

191220083 蒙芷露

191220097 时欣

191220160 张峻

最后再次感谢曹春老师一学期以来精彩的授课与辛勤的付出