

程序设计基础实践课程实验报告

小组：第五组

成员：刘志才，任睿哲，曲泓勃，邵辉，浦艳玲

报告编写者：刘志才，任睿哲

2024 年 9 月 10 日

目录	1
----	---

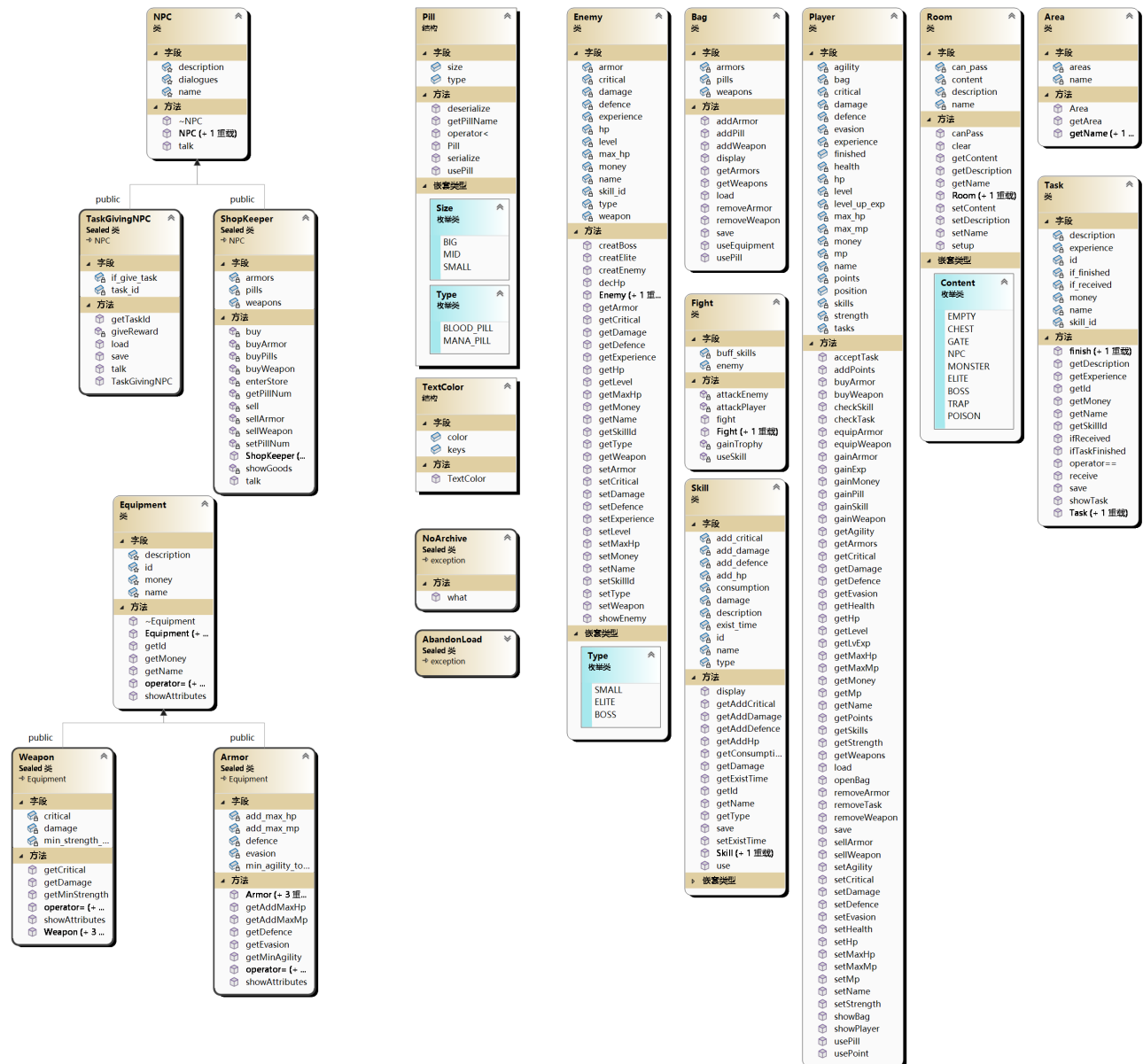
目录

1 程序设计总思路	1
2 小组分工及贡献率	2
3 项目管理和一些杂项	2
4 人物	3
5 背包	3
6 装备和道具	3
7 敌人及 npc	3
8 地图	3
9 战斗系统	4
10 存档	4
11 问题及其解决方案	4
12 不足和改进方向	5

第一部分 程序设计总思路

剧情介绍 在武侠世界中，一名青年主角踏入云霄城，与老张和城主等人交谈，得知附近的武威城城主陆洪以掠夺商队为乐，祸害四方，令无数百姓苦不堪言。主角怀揣正义之心，誓言为民除害。他披荆斩棘，目睹了无数家庭破碎与流离，最终将陆洪斩于剑下，成为云霄城的传奇英雄。然而，命运弄人，几年后，飞云商会会长段霖崛起，携手恶徒横行江湖，其武功远胜陆洪，且行踪隐秘。主角在攻打飞云商会的过程中，几度陷入暗器和毒气的险境，幸得剑客李白相助，终将段霖斩杀。自此，江湖的黑暗势力对主角的名号心生畏惧，云霄城得以长久和平，而主角的传奇旅程依旧在继续...

UML 类图



第二部分 小组分工及贡献率

姓名	负责模块	贡献率
刘志才	主函数，房间和地图，存档，帮助队友完善细节，改 bug	35%
任睿哲	战斗系统，背包，装备，技能，实现 npc 和任务，改 bug	30%
曲泓勃	敌人，保存，测试，ppt	20%
邵辉	玩家，剧情，测试	13%
浦艳玲	npc/任务/药的接口	2%

第三部分 项目管理和一些杂项

fmt 库 我们在项目中使用了开源的 fmt 库，可以在 [GitHub](#) 上找到他，使用这个库可以更加精细方便地格式化输出文本，并且它控制字体的颜色也是十分的好用。

CMake 在我们的开发过程中，我们使用了 CMake 来管理项目，CMake 可以生成不同平台（如 Windows、Linux、macOS）的构建系统文件，使项目能够在多种环境下构建。避免了组员之间因为环境，编译器等不同而造成的错误。CMake 能够自动处理项目依赖，方便了我们应用 fmt 这个库。并且 CMake 还可以方便的对程序进行打包，这样测试组就不需要手动编译再运行，大大提高了测试效率。

GitHub 我们使用了 GitHub 来方便管理我们的代码，这样我们可以踪代码变更、管理不同版本和恢复历史记录，以防修改时项目崩溃。团队成员可以并行开发、进行代码审查和轻松合并修改，提高协作效率，不必再手动的发送/接受各种文件，那样乱地要死。

C++17 标准 我们组设置了 C++17 的标准

- 使用了大量的结构化绑定来简化我们的代码，处理多返回值时十分的好用。
- 使用了 `std::optional` 来精细地处理返回值可能带来的异常，比起使用错误码更加地优雅。
- 使用了 `std::variant` 来进行类型检查，方便我们一次性处理多个类型。
- 使用了 `lambda`，方便我们随时定义一个函数，而不用跑到函数外面去定义。而且通过传 `lambda` 参数用 `std::function` 来接受可以让我们设置不同的对战效果，避免了分类判断。

代码风格 我们组的代码风格相当统一，命名方式和缩进格式均有限制：

命名方面，局部变量使用蛇形命名法，全局变量使用大驼峰命名法，全局常量使用全大写 + 下划线，函数统一用动词 + 名字的小驼峰命名法。

缩进方面，我们设置一个 `.clang-format` 利用 IDE 功能进行格式化。

在头文件中我们拒绝全局设定命名空间，而是使用前缀处理，在相应的 `.cpp` 文件中，由于我们的库只有一个，且不会造成命名冲突，所以我们允许使用了 `using namespace std;` 来简化处理。

第四部分 人物

玩家设计为了一个类，玩家具有一些基础属性：名字，等级，经验，生命值，元气值，健康点数，攻击力，防御力，暴击率，闪避率，金钱。

同时玩家持有一个背包、任务、技能。

玩家有三种基本属性点：健康，力量和敏捷。升级和探索可以获得属性点，可以自行选择加点方式。增加健康点数可以提升角色的最大生命值和最大元气值；增加力量点可以提升攻击力、暴击率和防御力，还可以拿起更强力的武器；增加敏捷点可以提升暴击率和闪避率，还可以装备更好的装备。

第五部分 背包

背包中主要存放三种物品，玩家的武器、护甲和药品，玩家可以通过打开背包来选择查看物品，使用药品和穿脱装备三个主要接口。

第六部分 装备和道具

装备设计了一个基类 Equipment，两个子类武器 (Weapon) 和护甲 (Armor) 两类继承了基类，他们都有一个唯一 id。

武器能够增加攻击力和暴击率，有最小力量点数的要求；护甲能够增加最大生命值，最大元气，防御力和闪避率，有最小敏捷点数的要求；在达到对应的点数要求之前，无法成功装备。

药品设计为了一个简单的类，有类型（回血，回元气）和大小（小，中，大或初级，中级，高级）之分。根据不同的类别有着不同的回复效果和回复数量。

第七部分 敌人及 npc

敌人我们成了小怪，精英怪和首领三个等级，有着不同的属性和掉落物，首领还有自己的额外机制。

npc 主要有两类（商人和任务型 npc），均继承自基类。

商人主要负责两个，铁匠和药店老板；铁匠会售卖/回收装备，药店老板负责售卖药。

任务型 npc 主要给玩家发布任务，完成任务后给予奖励，引导玩家游玩，帮助玩家快速升级。本作有比较完整的任务系统，玩家可以与 NPC 对话来接取任务，每个 NPC 的任务都不同，可能是让你杀掉敌人，为民除害，可能是去找一些东西。本作共有 17 个任务，丰富了战斗和探索的体验，在完成当前 NPC 给予的任务之前，是无法向他接取下一个任务的。

第八部分 地图

我们采用了传统的 MUD 游戏的地图设计：以房间为单元，地图是若干个房间的集合。每个房间代表一个独立的地点，并通过互相连接的通道形成复杂的网络。每个房间都有详细的描述，提供关于环境、物品和敌人的信息，增强了游戏的沉浸感。

我们设计了三张地图，分别是主城、武威楼和飞云商会，玩家可以自由切换 (有等价限制哦)。云霄城是游戏的主城，在这里不会被人攻击，玩家可以购买装备，药品，恢复血量和元气，以及与 npc 对话，接取任务，还可以找城主修炼，练习战斗和刷经验。武威楼和飞云商会则是反派的地盘，需要玩家去探索。

在主城和武威楼，我们设计了地图的可视化，玩家可以清楚地知道自己所处的位置和地图全貌。而在最终关卡飞云商会，我们改而告诉玩家当前的坐标和房间的信息，玩家需要自己探索，记忆路线。这种设计支持灵活扩展，还可以能够轻松添加新内容和调整现有房间。

第九部分 战斗系统

战斗系统我们采用了经典的回合制 1v1 战斗，我们加入了类似《只狼》的弹反，《怪物猎人》的见切的闪避机制。

玩家回合，玩家可以选择查看自身或者敌人的状态，发动普通攻击，使用技能和使用药品这五种选项。发动普通攻击，使用技能和使用药品都会结束当前回合，发动普通攻击会根据玩家的暴击率，造成双倍伤害的暴击效果，而使用技能会根据技能的种类，进行回血，攻击或加 buff。

玩家回合结束后，可以选择防御或闪避，防御功能会根据玩家的防御力减少对玩家造成的伤害；

闪避则会先对玩家的闪避率进行检测，我们用 C++ 的 random 库实现了一个有几率返回 true 或 false 的函数，检测通过则直接闪避成功，否则会进入手动闪避阶段，我们使用了 C++ 的线程 std::thread 一边加载进度条，一边通过 Windows 接口监听键盘，同时使用了 std::atomic 操作保证了线程安全，在这一阶段玩家需要在进度条出现时按下 'k' 键，反应时间大约为 100ms，如果成功则闪避成功，过早或过晚都会失败；

闪避成功会受到 0 点伤害，而失败则会收到怪物的全额伤害，不受防御力的减免，是风险与机遇并存的选项。

玩家可以自主选择攻击，技能，防御以及闪避，实现多种多样的战斗体验。在玩家或敌人血量 ≤ 0 时，战斗结束，打败精英敌人或者首领会获得额外的武器，护甲和技能奖励，小怪只会获得经验和金钱；

我们参考了绝大多数游戏的死亡机制，即玩家失败后会重新启动程序，而不是单纯地结束程序，我们从 GitHub 找到的这个好东西。

第十部分 存档

我们使用了 C++17 引入的 filesystem 库，每一个存档都创建一个独属的文件夹，将每个模块的文件存储在独立的文件夹中，可以有效组织和管理文件，减少命名冲突，简化备份和恢复过程，提升系统的扩展性和性能，同时方便清理旧数据和提高代码的可维护性。

第十一部分 问题及其解决方案

我们在开发过程中遇到了一些问题，列在下面

1. 文件编码问题，在开发过程中我们发现由于 GBK 和 UTF-8 的编码问题，导致我们遇到了乱码、中英文宽度不同的问题。
2. 尽管 C++17 下，switch 里面可以定义临时变量，但是它的生命周期得不到保证，导致在运行时有时会出现提前被释放。
3. 类的复制操作，我们遇到了复制不正确的问题。
4. 监听键盘事件不方便，难以控制。
5. 复杂逻辑处理错误。

我们对应的解决方案如下：

1. 我们统一使用了 UTF-8 并在中英文混合输出时我们手动地补足了英文较于中文缺少的字节。
2. 改用 if else 进行管理。
3. 经查阅我们遇到了 C++ 的“规则怪谈”，定义了移动构造和移动赋值操作后得以解决。
4. 使用了 Windows 提供的 API。
5. 由测试组反复测试，代码组 debug 改正。

补充 C++ “规则怪谈”：

1. 如果一个类定义了析构函数，那么你必须同时定义或删除拷贝构造和拷贝赋值函数，否则出错
2. 如果一个类定义了拷贝构造函数，那么你必须同时定义或删除拷贝赋值函数，否则出错，删除可导致低效
3. 如果一个类定义了移动构造函数，那么你必须同时定义或删除移动赋值函数，否则出错，删除可导致低效
4. 如果一个类定义了拷贝构造或拷贝赋值函数，那么你必须同时定义移动构造和移动赋值函数，否则低效

第十二部分 不足和改进方向

首先是编码问题，要注意到 UTF-8, UTF-16, UTF-32 的不同，我们在选择时应选择 UTF-8 进行存储，而使用 UTF-16 或 UTF-32 做相关的打印等操作，这样可以避免字节不同的问题而导致的出错。

部分类函数设计功能承担太多职责，应该适当地减少功能，而由调用者做判断。

对于监听键盘事件，自己写及其有可能出现 bug，可以使用已经封装好了第三方库 ncurses 做处理。

对于代码中反复出现的部分，可以利用 C++ 的模板技术和 SFINAE 技术提取封装。

由于操作过于简单，缺少了必要的新手指引，可以增添新手指引，增加玩家的目的性。

游戏没有平衡性，简单处理可以在网上查阅相关的数学公式计算。