****

**2025 年 夏 季学期**

**程序设计基础实践**

**--MUD游戏课程报告--**

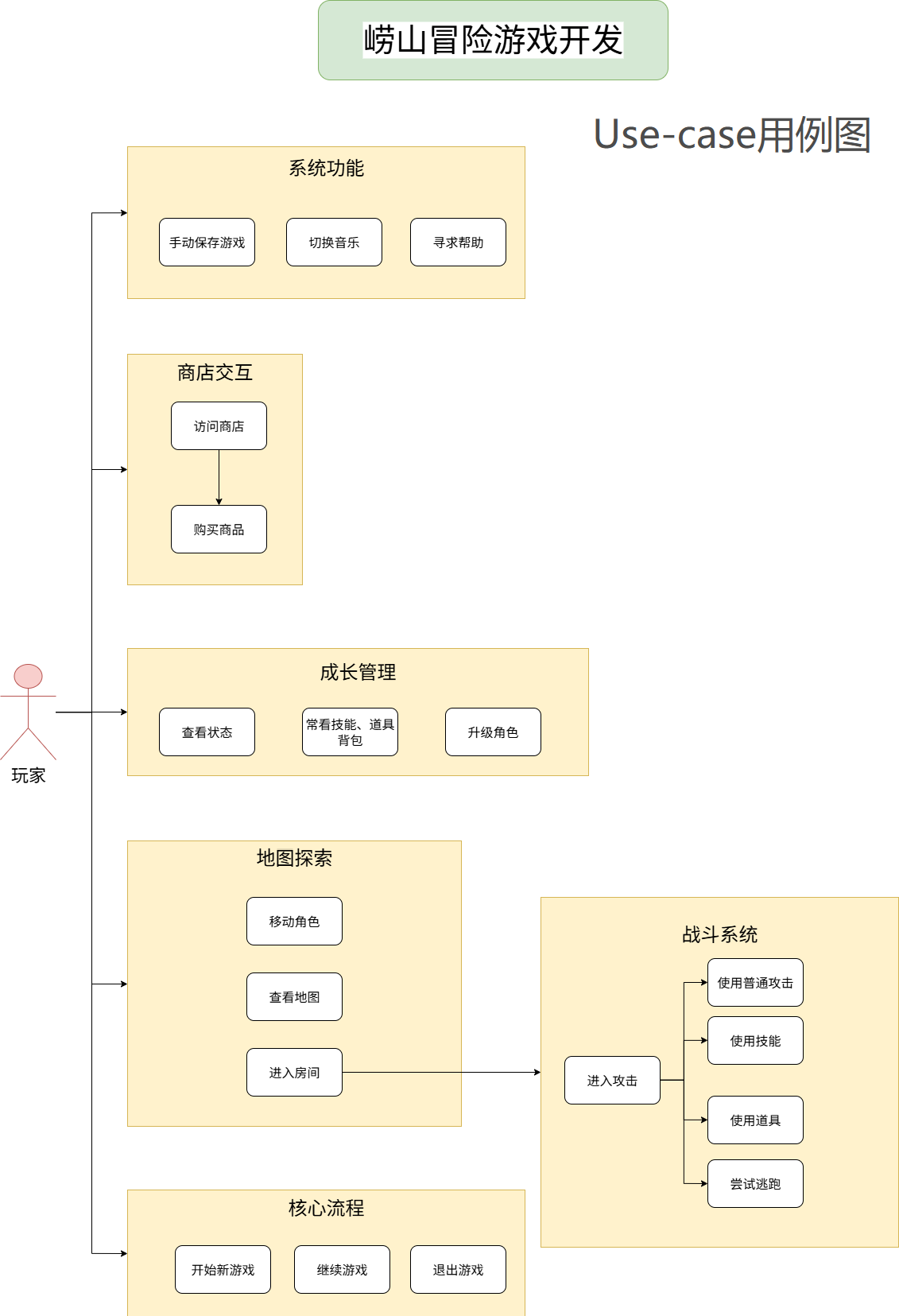
**指导老师：傅宇**

**组长：刘和雄**

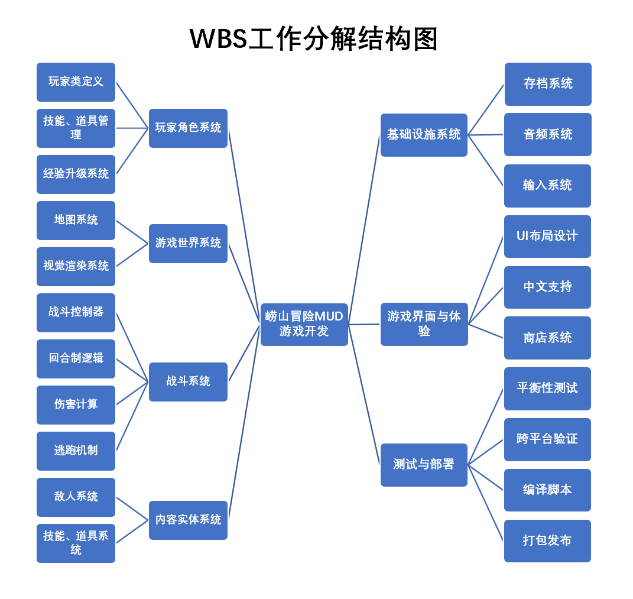
**组员：鄂思羽、韩雨希、尹传林、郑传奇**

**（一）需求分析**

* **Use - Case用例图**

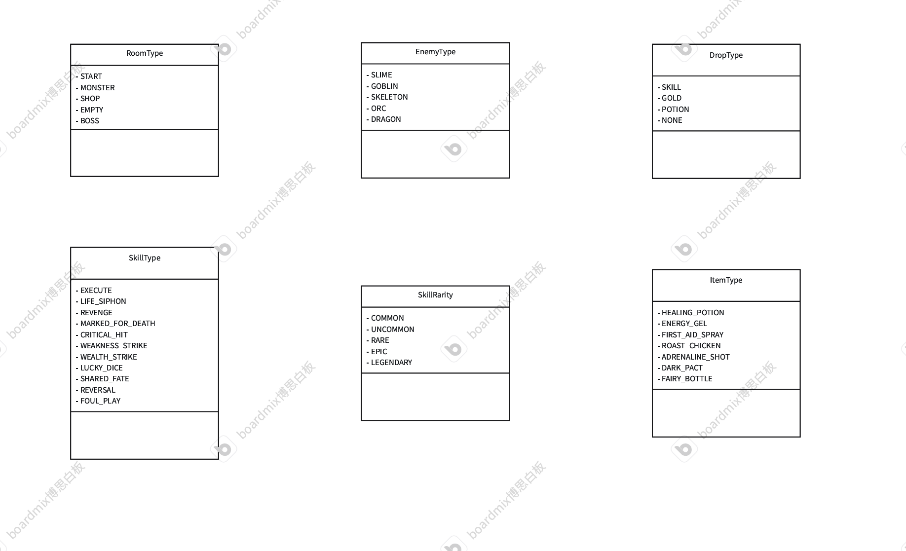
****

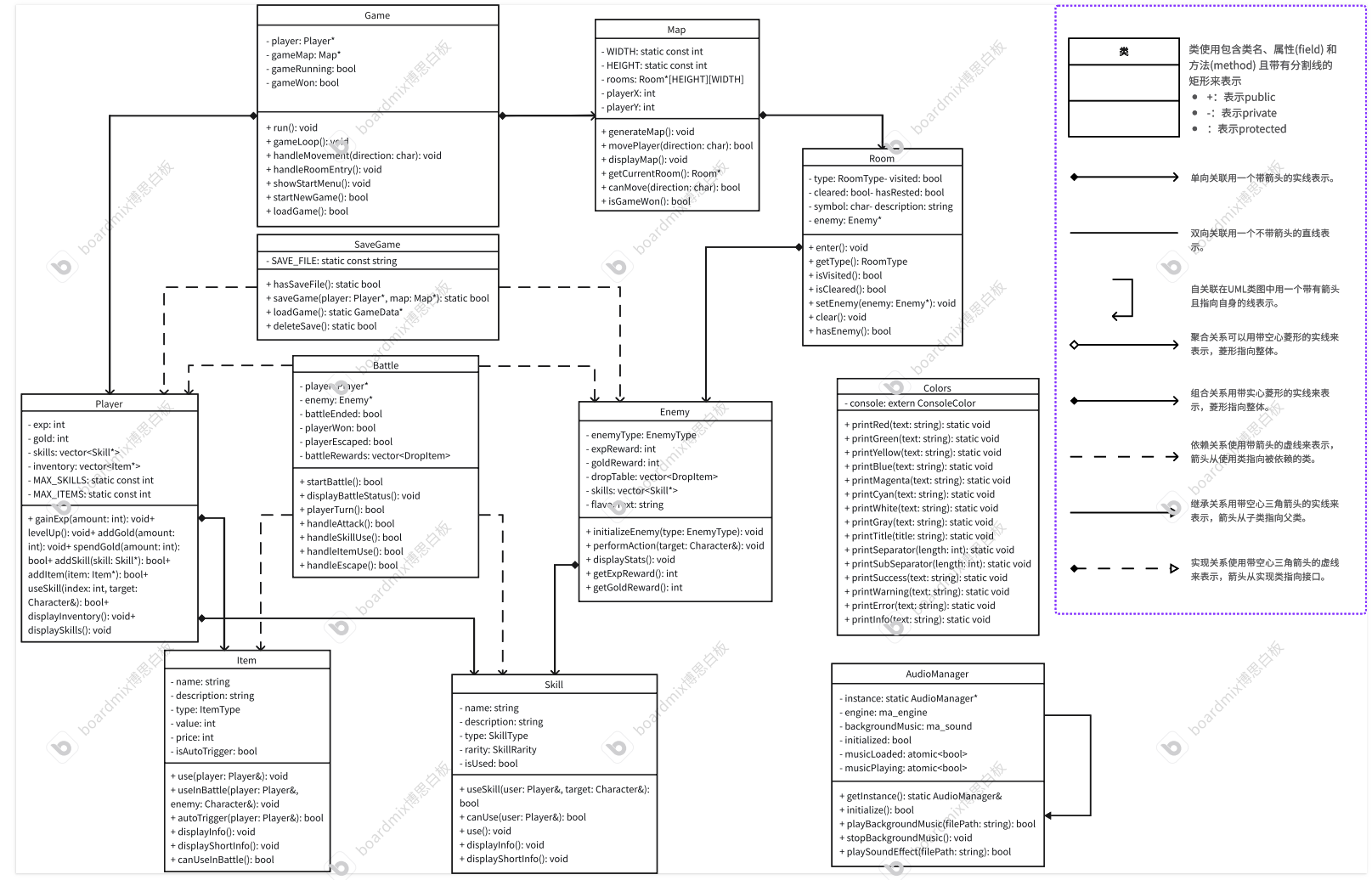
* **WBS工作分解结构图**

****

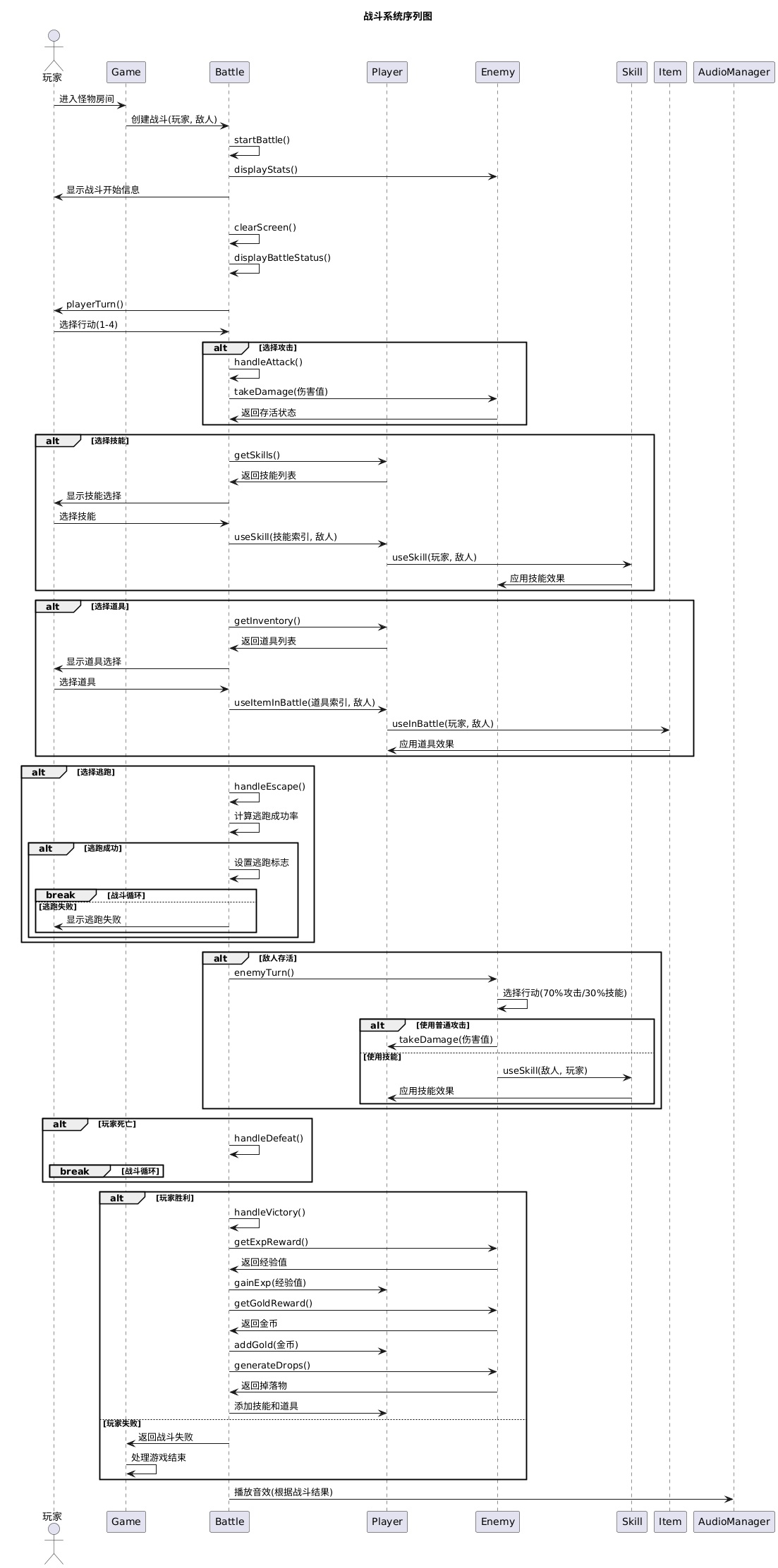
**（二）程序设计**

* **UML类图**

****

****

* **核心算法流程图、UML序列图**

****

**（三）组件说明**

* **STL标准模板库组件说明**

**1. 容器 (Containers)**

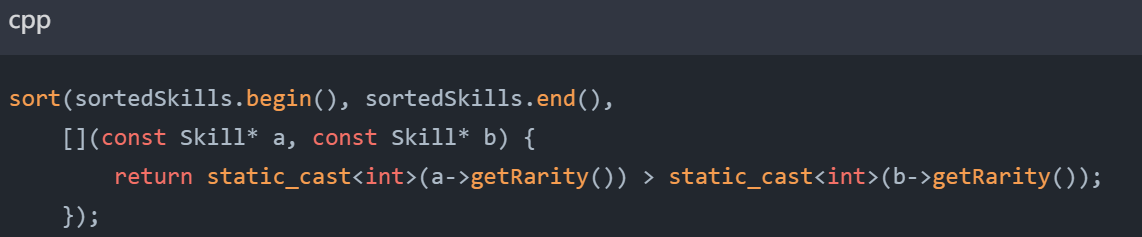
• std::vector: 用于管理动态数组。

****

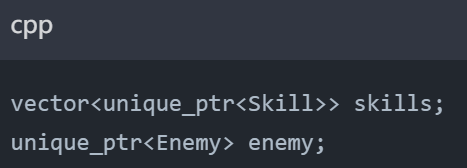
• std::string: 用于处理所有文本信息，如角色名、技能描述、物品名称等。

**2. 其他STL组件**

* 算法 (std::sort): 在 Player::displaySkillBag() 中，使用 std::sort 对技能按稀有度进行排序。

****

* 智能指针的缺失: 代码中大量使用了原始指针（Skill\*, Item\*, Enemy\*等），并手动进行 delete 操作。在现代C++中，更推荐使 用 std::unique\_ptr 或 std::shared\_ptr 来自动管理资源生命周期，避免内存泄漏的风险。例如：

****

**（四）设计模式**

* **设计模式的使用**

**1. 单例模式 (Singleton Pattern)**

目的：确保一个类只有一个实例，并提供一个全局访问点。

实现：在 AudioManager 类中。

****

用途：音频管理在整个游戏中只需要一个实例，用于统一控制背景音乐和音效的播放。

**2. 工厂方法模式 (Factory Method Pattern)**

目的：定义一个创建对象的接口，但让子类决定实例化哪一个类。工厂方法使一个类的实例化延迟到其子类。  
实现：您使用了多个“工厂类”来创建对象。

* EnemyFactory: 根据 EnemyType 创建特定的敌人对象 (createRandomEnemy, createBoss)。
* SkillFactory: 创建各种技能对象 (createExecute, createRandomSkill, createSkillByName)。
* ItemFactory: 创建各种道具对象 (createHealingPotion, createRandomItem)。  
  用途：将对象的创建与使用分离。Game 和 Battle 等类不需要知道 Enemy 或 Skill 的具体实现细节，只需向工厂请求所需类型的对象即可，极大地提高了代码的灵活性和可维护性。

**3. 状态模式 (State Pattern)**

目的：允许一个对象在其内部状态改变时改变它的行为。

实现：通过 RoomType 枚举和 Room 类的 getType() 方法。

****

用途：处理不同房间类型（START, MONSTER, SHOP等）的不同行为，例如生成不同的ASCII艺术、触发不同的事件。Room 对象的行为取决于它的 RoomType 状态。

**4. 策略模式 (Strategy Pattern)**

目的：定义一系列的算法，把它们一个个封装起来，并且使它们可相互替换。该模式使得算法可独立于使用它的客户而变化。  
实现：在 Skill 类和 Item 类中非常明显。

* 每个 Skill 子类（通过工厂创建）都封装了一个不同的算法（如 useSkill 方法）。
* Battle::playerTurn() 中，玩家选择技能或道具，其实就是选择不同的策略（算法）来执行。  
  用途：技能和道具系统。每个技能/道具都有不同的使用效果（算法），但它们可以通过统一的接口（useSkill, use）来调用，使得战斗逻辑和具体的技能实现解耦。

**5. 观察者模式 (Observer Pattern) - 变体/简化**

目的：定义对象间的一种一对多的依赖关系，当一个对象的状态发生改变时，所有依赖于它的对象都得到通知并被自动更新。  
实现：您的代码中没有完整的观察者模式，但有一些类似的思想。例如，当玩家击败怪物时 (monstersDefeated++)，这个状态的变化会影响到 AsciiMapRenderer::displayStatus 的显示以及 Map::isGameWon() 对胜利条件的判断。  
用途：管理游戏状态和UI更新之间的松散耦合。

****

**（五）问题及解决方案**

* **中文显示/输入乱码**

现象：Windows 控制台中文标题/文案出现问号或乱码，输入姓名不稳定。

解决：启动时执行 chcp 65001，使用 SetConsoleCP/SetConsoleOutputCP 切换 UTF-8；源码文件统一 UTF-8 无 BOM；控制台 选择等宽中文字体。

* **随机刷怪种类受限**

现象：怪物房间重复度高、种类单一，体验单调。

解决：扩充 EnemyType（史莱姆·赤、哥布林斥候、幽影道士、石像鬼、雾妖·崂山、堕仙残魂）并加入工厂随机池；维持 BOSS 独立创建。 -

* **数值平衡（普攻过强）**

现象：玩家仅用普攻即可碾压，技能存在感弱。

解决：重写伤害为“百分比减伤”公式（上限 50%），上调普通/精英怪 HP/DEF 曲线，保留 BOSS 数值与三阶段；实测回合数更 匹配，技能价值提升。

* **cin 与 getline 混用**

现象：读取字符串被跳过（立即返回空行）。

解决：在使用 getline 前先清空输入缓冲（cin.ignore(numeric\_limits::max(), '\n')），或全程统一用 getline + 手 动解析。

* **无法播放自定义背景音乐**

现象：希望添加一个背景音乐，但无法实现

解决方式：使用github开源的头文件MINIAUDIO，仓库网址：[mackron/miniaudio: Audio playback and capture library written in C, in a single source file.](https://github.com/mackron/miniaudio)

阅读文档，实现自定义背景音乐的播放

**（六）小组成员分工**

* 组长：刘和雄
* 负责编写架构与主程序：规划项目结构与主循环（Game）、输入处理、固定布局UI输出、Windows UTF-8 显示/输入适配
* 负责编写战斗系统：回合流程、普通攻击与百分比减伤公式、暴击判定、逃跑机制、胜负结算与战利品生成
* 负责编写战斗系统：回合流程、普通攻击与百分比减伤公式、暴击判定、逃跑机制、胜负结算与战利品生成
* 负责编写音频与体验：miniaudio集成，背景音乐与基础音效
* 组员：韩雨希
  + 绘制 WBS工作分解结构图 和 Use-Case用例图
  + 撰写 遇到的问题及解决方案 和 成员分工职责
  + 负责编写技能系统：一次性技能框架、工厂与随机生成/按名加载映射，完成功能技能（致死打击、生命吸取、复仇、死亡标记、致命一击、虚弱打击、财富一击、运气骰子）
* 组员：尹传林
* 绘制 UML类图
* 绘制 流程图或UML序列图
* 撰写 STL 和 设计模式 的说明
* 负责编写敌人与BOSS：敌人类与工厂、随机刷怪接入；新增6类敌人落地；BOSS三阶段AI（喷火/龙息风暴/觉醒回血与强攻）
* 组员：鄂思羽
* 撰写课程报告
* 完善游戏数值和背景剧情和敌怪的类型
* 准备PPT中 "故事背景" 和 "功能介绍" 部分的文字稿
* 负责编写地图与探索：3×8 ASCII地图渲染、房间类型（起点/怪物/商店/休息/BOSS）、移动与进入事件处理
* 组员：郑传奇
* 制作课堂展示PPT
* 负责课堂上的演讲展示环节
* 负责编写道具与商店：道具工厂与背包容量校验、战斗内外使用规则、价格与购买流程