MUD游戏《重返崂山》测试报告

**测试信息**

被测试游戏名称：《重返崂山》

被测试小组组长：刘和雄 24070001071

测试小组组长：唐梓翔 24020007114

分数：95

**代码部分（47分）**

一、程序代码整体结构

1. 整体架构分析

游戏核心逻辑：Game.h 负责游戏主循环、用户输入处理、战斗流程调度等核心控制

地图系统：Map.h 管理地图生成、房间分布、玩家移动；Room.h 定义房间类型及行为

实体系统：Enemy.h 定义敌人类型及创建逻辑；Player（隐含）、Skill.h 处理角色技能

辅助系统：Colors.h/.cpp 负责控制台颜色输出；SaveGame.h 处理游戏存档；AudioManager.h 管理音频（代码不完整）

工具类：Item.h 定义道具类型及功能；EnemyFactory 采用工厂模式创建敌人

2. 可改进点

音频模块不完整：AudioManager.h 仅包含一行检查初始化的代码，未体现完整功能，可能导致模块缺失

二、类和类关系的抽象

Map类：聚合Room对象（二维数组rooms[HEIGHT][WIDTH]），负责地图生成与玩家位置管理

Room类：封装房间类型（RoomType枚举）、状态（是否访问 / 清理）及敌人关联，通过setEnemy/getEnemy与Enemy形成关联关系

EnemyFactory类：创建敌人实例，隐藏敌人创建细节

Game类：组合Player和Map对象，作为游戏核心控制器，协调各模块交互，体现 "包含" 的组合关系

SaveGame类：依赖Player和Map的数据接口（如getPlayerX/isRoomVisited）实现存档

三、编码格式

头文件保护：所有头文件（Colors.h、Room.h等）均使用#ifndef/#define/#endif防止重复包含，符合 C++ 编码规范

缩进与排版：代码缩进一致（函数体、循环体、分支体均有缩进），括号使用统一（{另起一行或跟随语句），可读性良好

命名风格：类名使用 PascalCase（Map、EnemyFactory），函数，变量使用 camelCase（movePlayer、playerX），常量全大写（WIDTH、HEIGHT）

四、命名与注释

命名语义化：变量名（monstersDefeated、hasRested）、函数名（incrementMonstersDefeated、canRest）均能准确反映功能

枚举清晰：RoomType（MONSTER/SHOP）、ItemType（HEALING\_POTION/ENERGY\_GEL）枚举值命名明确，直观区分类型

**功能体验（48分）**

1. 界面交互

输入时重写getinput函数，不显示用户输入的字符而直接进入执行，仅提供无效提示，在界面美观与交互清晰中选择了前者。游戏中最常见的界面之一是显示了人物状态、地图、控制面板的大型界面，对于控制台游戏来说信息略微冗杂。战斗界面常态显示对战双方状态，方便玩家即时获取信息，根据形势决策战斗。 但是在战斗过程中的技能与物品的信息显示之后缺少清屏，在思考犹豫时可能需要不时向上查找状态。

1. 核心玩法

游戏主要玩法是前往各个房间应对随机事件，最常见的是战斗事件，每当战斗胜利后都会获得金币和随机技能、道具作为奖励，从而帮助玩家继续游戏。游戏开始设置了小段剧情，途中也实时显示玩家已打败的怪物数，引导玩家逐步推进，最终到达boss关击败boss完成游戏。地图中设置了商店等其他事件帮助玩家生存。

1. 战斗设计

玩家在战斗中主要有三种决策。除去比较乏味的普通攻击，玩家可以通过使用一次性的技能和道具来使战斗难度下降，从而触发“决策-奖励”机制使玩家获得满足感。技能的设计较为多样，包括根据血量造成额外伤害、弱化敌人、攻击恢复生命、强化自身等，适当地使用技能可以扭转战局，战胜更加强大的敌人。道具主要为回复生命的药物，用来支持连续的战斗而不使玩家过于容易地死亡。

1. 体验优化

游戏中设定了技能和道具的库存上限，迫使玩家将其纳入普通战斗考量。地图设计较为狭小单一，事件不够丰富。增设了背景音乐、画面颜色等功能，提升玩家的全方位体验。