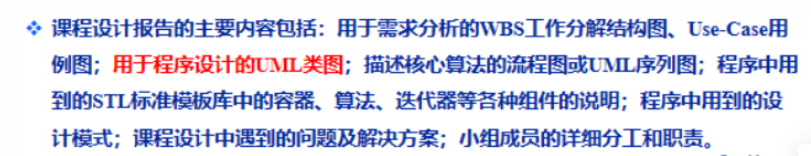
游戏设计阶段

时间线：



这是课程报告在设计阶段要求的：

**在分析设计阶段，要利用工作分解结构（WBS)、Use-Case用例图、UML类图、UML序列图（或者流程图）等方法和工具，进行充分的分析、抽象和设计表达。**



l关卡结构：每层由**随机房间**组成，房间顺序、内容每层魔域均不同。

l初始：玩家可以自主选择不同角色，（如“战士”“法师”“盗贼”），每个角色有固定初始卡牌。

l房间类型：

o战斗房：随机敌人（普通/精英/Boss），击败后获得卡牌奖励（或者其他）；

o事件房：文本叙事分支（如“遇到受伤的旅行者，选择帮助或掠夺”）（可以设计剧情之类的），结果影响后续流程（获得奖励/触发惩罚）；

o商店房：用金币购买卡牌/删除卡牌；

o休息房：恢复血量；

l卡牌获取：通过战斗胜利、事件奖励、商店购买获得新卡牌。

l卡牌管理：可通过商店、事件房等平衡卡牌数量（比如商店可删除卡牌，事件房惩罚机制可消除卡牌等）。

l卡牌效果：待确定。

l层数设计：待确定，每层结尾有Boss，击败后进入下一层。

简化：所有的战斗结果、掉落物品都是固定的，玩家的任务打怪然后获得物品，直到最后一层。我们可以先完成三到五层，保证功能完整，难度逐渐增加。

关于卡牌：我认为可以将技能、属性成长甚至生命值完全卡牌化，建立一个牌库，供玩家使用，初次进入楼层前可以有原始卡牌，待打怪成功可以获得卡牌恢复生命值等。

游戏流程：

开始界面->进入游戏->-些剧情->进入地图->玩家移动->进不同房间->战斗房间进入战斗界面以此类推->持续推进,最后进入 BOSS 战->击败 BOSS 显示胜利->剧情结束。



一、整体程序结构划分

核心模块

游戏主循环（Game Loop）

事件管理（Event Manager）

状态管理（State Manager）

游戏数据结构

卡牌系统（Card System）

玩家角色（Player）

敌人（Enemy）

牌组（Deck）

战斗系统（Battle System）

爬塔地图（Map）

辅助模块

随机数生成（Random）

用户输入（Input）

输出显示（Console/GUI）

存档/读取（Save/Load）

二、具体模块功能与任务

1. 游戏主循环（Game Loop）

负责游戏的整体流程控制。

处理游戏状态切换：地图选择、战斗、事件、商店、结算等。

2. 状态管理（State Manager）

管理不同游戏界面和状态：

主菜单

地图选择界面

战斗界面

商店界面

事件界面

游戏结束界面

3. 地图系统（Map）

实现爬塔地图结构（多节点，分支选择）。

每个节点可能是战斗、商店、事件、休息等。

玩家选择路径，推进游戏。

4. 卡牌系统（Card System）

设计卡牌基类和不同类型卡牌（攻击、防御、技能等）。

卡牌属性：名称、能量消耗、效果描述、具体效果实现。

卡牌的使用逻辑和效果触发。

5. 牌组系统（Deck）

管理玩家的牌组（抽牌堆、弃牌堆、手牌）。

实现抽牌、弃牌、洗牌等功能。

牌组构建和升级机制。

6. 玩家角色（Player）

属性：生命值、能量、护甲、防御等。

管理玩家状态和buff/debuff。

实现玩家回合逻辑：抽牌、打牌、结束回合。

7. 敌人系统（Enemy）

敌人属性和行为模式。

敌人AI：简单的攻击、防御等行为。

多种敌人类型设计。

8. 战斗系统（Battle System）

回合制战斗流程控制。

玩家回合、敌人回合切换。

处理卡牌效果、伤害计算、防御计算。

战斗胜负判定。

9. 事件系统（Event Manager）

随机事件触发。

事件选项和结果处理。

事件对玩家状态和牌组的影响。

10. 商店系统（Shop）

提供卡牌和道具购买。

货币系统管理。

购买后更新玩家牌组或状态。

11. 随机数生成（Random）

用于生成地图、卡牌奖励、敌人、事件等随机内容。

保证游戏的多样性和重玩性。

12. 输入输出模块

处理玩家输入（键盘/鼠标）。

显示游戏界面（文字界面或简单图形界面）。

提示信息和状态反馈。

13. 存档与读取

保存当前游戏进度。

读取存档恢复游戏。