**扫雷游戏开发**

### 1项目目标

### “扫雷”是一款大众类的益智小游戏。游戏目标是在最短的时间内根据点击格子出现的数字找出所有非雷格子，同时避免踩雷，踩到一个雷全盘皆输。

本项目用C语言开发一款简单的扫雷游戏，在屏幕自动生成雷区，用户可以通过鼠标进行操作。

本扫雷游戏系统旨在为用户提供一个简单易用、界面友好的扫雷游戏体验。系统将支持以下功能：

**.**开始新游戏

**.**自动生成随机雷区

**.**提示周围地雷数

**.**标记地雷

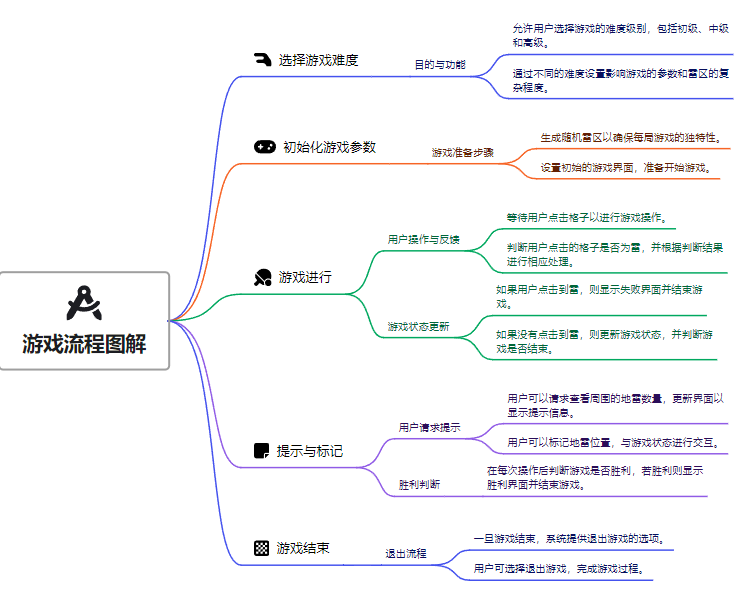
**.**游戏结束（胜利/失败）反馈

**.**时间计时功能

**.**可选择不同难度（初级、中级、高级）

本说明书旨在描述基于C语言和EasyX图形库的扫雷游戏的概要设计。该项目旨在提供一个简单而富有挑战性的游戏体验，用户可以根据需求选择游戏的难度，玩家在游玩本游戏时需要进行逻辑推理，通过排除法来确定地雷的位置。这一过程不仅考验玩家的观察力和记忆力，更重要的是锻炼了玩家的逻辑思维能力和问题解决能力。

### 2项目流程图



**3.开发环境及技术栈**

3.1**开发工具**： visual studio 2013

3.2 **操作系统**：Windows 10

3.3 **开发语言**：C语言+EasyX图形库

**4. 系统架构**

系统分为以下几个主要模块：

* **初始化模块**：负责游戏窗口的创建、EasyX库的初始化、游戏LOGO的显示以及游戏模式的选择界面。
* **雷区生成模块**：根据用户选择的游戏模式（16雷、30雷、50雷），在16x32的游戏格子上随机布置地雷。
* **游戏逻辑模块**：处理用户的鼠标点击事件，判断点击的是地雷、空白区域还是数字区域，并更新游戏状态。
* **时间记录模块**：记录并显示游戏开始到结束的时间。
* **声效处理模块**：在特定事件（如踩雷）时播放声音。
* **用户界面模块**：负责游戏界面的绘制，包括游戏格子、地雷标记、数字提示等。
* **游戏继续模块**：在踩雷或通关后，提供重新开始游戏的选项。

**5. 数据结构**

* **雷区数组**：一个二维数组，用于存储每个格子是否为地雷的信息。
* **标记数组**：一个二维数组，用于存储用户是否标记了某个格子为地雷。
* **时间变量**：用于记录游戏开始到当前的时间。
* **游戏状态变量**：用于表示游戏是否进行中、是否胜利或失败。

**6. 模块设计**

**6.1 初始化模块**

* 创建游戏窗口，设置窗口大小和标题。
* 初始化EasyX图形库。
* 显示游戏LOGO和游戏模式选择界面，等待用户选择。

**6.2 雷区生成模块**

* 根据用户选择的游戏模式，确定地雷数量。
* 在16x32的游戏格子上随机布置地雷，确保地雷数量正确且分布均匀。

**6.3 游戏逻辑模块**

* 处理用户的鼠标点击事件。
* 判断点击的格子是否为地雷、空白区域或数字区域。
* 更新雷区数组和标记数组。
* 根据点击结果更新游戏状态（进行中、胜利、失败）。

**6.4 时间记录模块**

* 在游戏开始时记录当前时间。
* 在游戏结束时计算并显示游戏时间。

**6.5 声效处理模块**

* 加载声效文件（如踩雷声、胜利声）。
* 在特定事件发生时播放相应的声效。

**6.6 用户界面模块**

* 绘制游戏格子，显示地雷标记和数字提示。
* 显示游戏时间、游戏状态等信息。
* 更新用户界面以反映游戏状态的变化。

**6.7 游戏继续模块**

* 在游戏结束（踩雷或通关）后，显示提示信息。
* 提供重新开始游戏的选项，并处理用户的重新开始游戏请求。

**7. 接口设计**

* 用户界面：提供游戏LOGO、游戏模式选择界面、游戏格子显示、时间显示、游戏状态提示等。
* 用户交互：通过鼠标点击进行游戏操作，包括标记地雷、打开格子等。
* 声效接口：提供加载和播放声效的功能。

**8. 安全性与可靠性**

* 确保游戏运行时不会出现内存泄漏或崩溃等问题。
* 处理用户输入的异常情况，如非法字符或无效操作。
* 在游戏结束时保存并清理资源，如关闭图形库和释放内存。

**9. 性能需求**

* 游戏应能够在大多数计算机上流畅运行，不会出现明显的卡顿或延迟。
* 图形界面应清晰、美观，符合用户视觉习惯。
* 声效应清晰可辨，不影响游戏体验。

**. 附录**

* 详细的系统流程图、数据流程图等图表。
* 代码中使用的第三方库（如EasyX图形库）的文档和链接。
* 声效文件的格式、来源和加载方法说明。

### 