南开大学 计算机大类

姓名：郭鑫隆

学号：2311754

班级：2-2

2024年5月12日

高级语言程序设计

实验报告

目录

[高级语言程序设计大作业实验报告 2](#_Toc21286)

[一. 作业题目 2](#_Toc29969)

[二. 开发软件 2](#_Toc18364)

[三. 课题要求 2](#_Toc742)

[四. 主要流程 2](#_Toc25240)

[1． 整体流程 2](#_Toc20452)

[2． 算法或公式 3](#_Toc25354)

[（1）基于随机化BFS算法的迷宫生成 3](#_Toc22508)

[（2） 主界面及游戏界面绘制 3](#_Toc28665)

[（3）玩家操作 3](#_Toc32428)

[（4） 开始，暂停，退出，游戏设置 4](#_Toc23860)

[（5） 基于DFS算法的迷宫问题求解及其可视化 4](#_Toc13349)

[五. 项目测试 4](#_Toc28265)

[六. 收获 5](#_Toc23116)

[1． 继承与多态的使用 5](#_Toc10799)

[2． 元对象和信号槽机制 5](#_Toc16275)

高级语言程序设计大作业实验报告

1. **作业题目**

迷宫游戏

1. **开发软件**

Qtcreator13.0.1

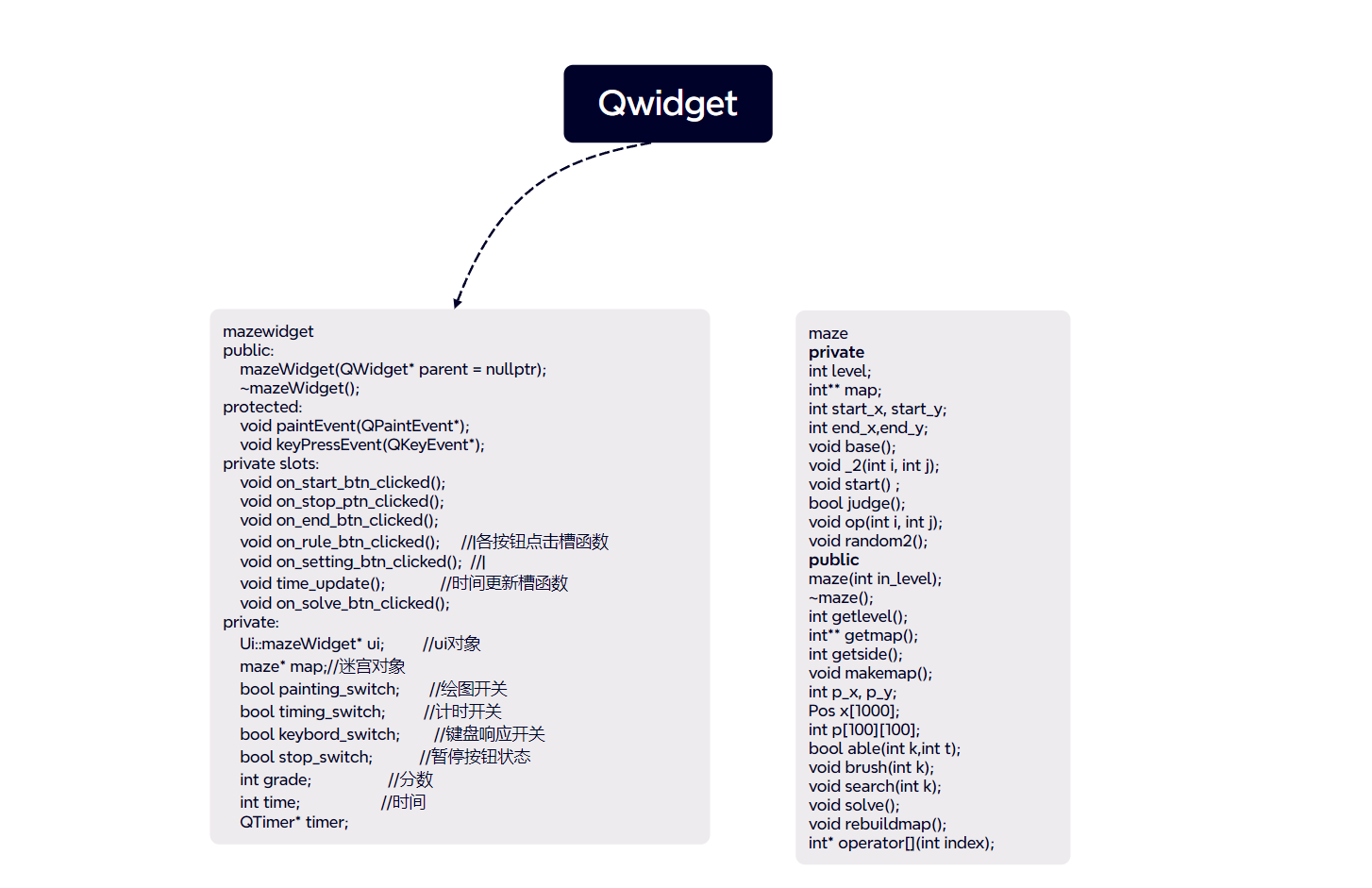
1. **课题要求**
2. 面向对象。
3. 单元测试。
4. 模型部分
5. 验证
6. **主要流程**
   1. **整体流程**

实现思路：

Maze类用来创建抽象迷宫对象

Mazwidget类用于实现图形界面效果，对应ui文件

其他类继承关系

图 1shape类图

根据输入类型要求，使用相同的基类指针构造不同的子类对象。使用基类指针虚函数引用调用子类实现，达到多态效果。

* 1. **算法或公式**

1. [基于随机化BFS算法的迷宫生成](#_Toc22508)

在 **maze.cpp** 中，我们实现了一种基于随机化的 BFS（Breadth-First Search）算法来生成迷宫。该算法是一种贪心算法，通过随机选择未访问过的墙壁，并将其打通来逐步扩展迷宫路径。具体步骤如下：

**初始化地图**：在 **base()** 函数中，我们首先将地图的边界设为 **-1**，墙壁设为 **0**，空单元设为 **1**，并将起点设定为 **(1, 1)**。

**选择起点**：在 **start()** 函数中，我们选择起点，并将起点周围的墙壁标记为待定墙壁 **2**。

**随机选择待定墙壁**：在 **random2()** 函数中，我们随机选择一个待定墙壁，并进行判断操作。如果墙壁旁边有已定单元，则将其打通成迷宫路径，否则将其保留为墙壁。

**循环操作**：通过不断的随机选择待定墙壁并进行操作，直到所有的墙壁都被判断过，即地图中不存在待定墙壁。

**生成迷宫**：最终，我们就可以得到一个完整的迷宫地图。

1. [主界面及游戏界面绘制](#_Toc28665)

游戏界面由 **mazeWidget** 类实现，通过 **QWidget** 组件绘制迷宫地图和角色。在 **paintEvent()** 函数中，我们根据地图的状态绘制迷宫，并根据角色的位置绘制角色图像。游戏界面还监听键盘事件，以便玩家通过键盘操作移动角色。

1. [玩家操作](#_Toc32428)

玩家可以通过键盘操作移动角色，操作方式包括 **WASD** 或者 **IJKL** 控制方向。游戏界面监听键盘事件，根据玩家输入更新角色位置，并实时更新游戏界面。

1. [开始，暂停，退出，游戏设置](#_Toc23860)

游戏界面提供了开始、暂停、退出和游戏设置等功能按钮。玩家可以通过点击按钮来开始游戏、暂停游戏、退出游戏或者设置游戏难度。这些按钮通过 **QPushButton** 组件实现，并与相应的槽函数连接，以便在点击时触发相应的操作。

1. [基于DFS算法的迷宫问题求解及其可视化](#_Toc13349)

在迷宫问题求解部分，我们采用了深度优先搜索（DFS）算法来寻找迷宫的解，并对解进行可视化显示。具体步骤如下：

**深度优先搜索（DFS）算法：**在 **maze.cpp** 文件中，我们实现了深度优先搜索算法来解决迷宫问题。在 **search()** 函数中，我们从起点开始进行递归搜索，每次尝试向上、下、左、右四个方向移动，直到找到终点或者无法继续移动。在搜索过程中，我们使用一个数组 p[][] 来记录已经访问过的路径，并在找到终点时标记出正确的路径。搜索完成后，我们得到了迷宫的解路径。

解的可视化显示：在 **mazeWidget.cpp** 文件中，我们实现了解的可视化显示功能。在点击解迷宫按钮后，程序会调用迷宫对象的 **solve()** 函数来寻找迷宫的解。然后，我们将解路径标记在地图上，并在界面上显示出来，以便玩家清晰地看到迷宫的解路径。

**五. 项目测试**

我们对项目进行了全面的单元测试，覆盖了迷宫生成、玩家操作、游戏功能等方面。通过单元测试，我们验证了迷宫生成算法的正确性，以及玩家在游戏中的操作是否符合预期。在测试过程中，我们发现并修复了一些潜在的问题，确保了游戏的稳定性和可靠性。

**六. 收获**

**1. 面向对象设计**

通过本次项目，我们深入理解了面向对象的设计思想，并将其应用到实际项目中。通过定义抽象类和继承关系，我们实现了迷宫游戏的各个功能模块，使得代码结构清晰、易于扩展和维护。

**2. 单元测试**

学会了编写单元测试，并运用单元测试对项目进行了全面的测试。单元测试帮助我们及时发现和解决了项目中的一些问题，提高了代码质量和可靠性。

**3. 算法实现与可视化**

通过实现随机化BFS算法和DFS算法，对算法有了更深入的理解，并学会了如何将算法与图形界面相结合，实现算法的可视化展示，提高了用户体验。

通过这次项目，不仅掌握了Qt Creator等开发工具的使用，还提升了面向对象设计、算法实现和单元测试等方面的能力，为今后的软件开发项目奠定了良好的基础。