“谁是人上人”作业报告

队名：家瑛说的都队 队伍编号：69 队长：吕佳楠 队员：李明隆、罗珑赞

一、功能概述：

菜单界面：

菜单界面设计简洁，主要由三个按钮构成： "进入游戏"，"游戏介绍"和 "退出游戏"。这些按钮提供了直观的导航，用户通过点击不同的按钮可以轻松实现各自的功能。

游戏介绍界面：

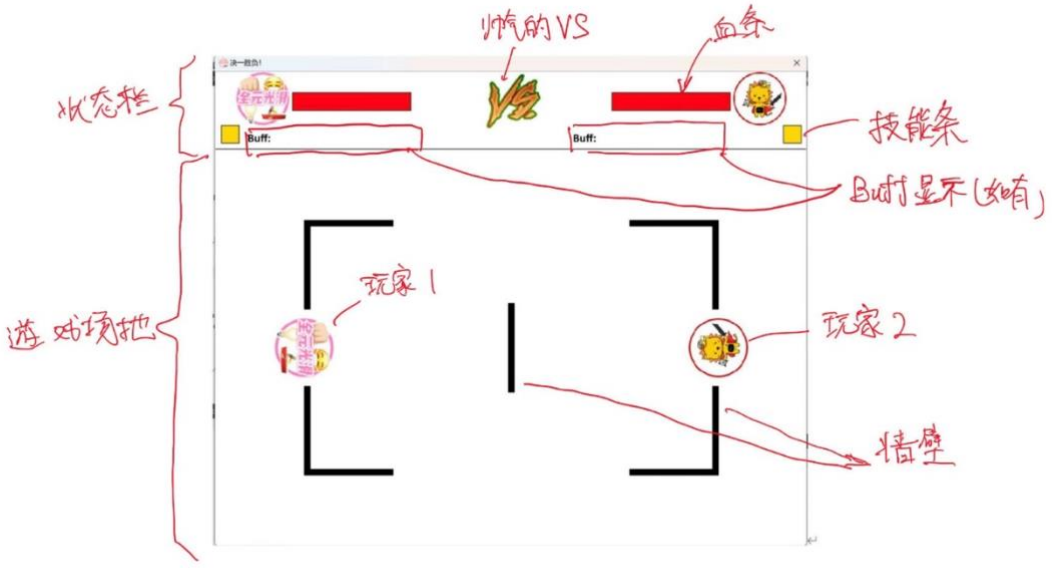
这个界面详细解释了游戏的背景故事和操作规则，帮助新玩家快速了解并掌握游戏。

角色选择界面：

这个界面在两侧展示了可供选择的角色，中间部分则给出了每个角色技能的描述。当一方玩家选择了角色后，所选角色将在已选角色栏中显示。等到双方都确定了各自的角色后，右上角将出现 "开始游戏" 按钮。

游戏主体：

在角色选择界面点击 "进入游戏" 按钮后，玩家将进入游戏的主体部分，开始进行激烈的对战。



游戏结算界面：

游戏结束后，结算界面将根据游戏的结果展示出胜利方及其代表的院系。

总体而言，本游戏从用户体验出发，设计了清晰直观的导航系统，并在游戏过程中引入多样化的元素，提供了丰富而有趣的游戏体验。

二、游戏设定详述：

玩家初始属性：

所有角色开始时均具有如下属性：

生命值：10

子弹攻击力：1

速度：3 （格每游戏刻）

子弹攻击间隔：1 秒

技能冷却时间：15 秒

角色移动：

玩家通过键盘控制角色在游戏场地内自由移动。当玩家停止前进时，角色的速度会以一定的减速度降至 0，这与现实物理相吻合。游戏有碰撞检测，因此角色不会穿过墙体或相互穿模。当玩家碰到墙体或边界时，其行为会有所不同。当角色向墙体移动时，速度将降至 0（但角色仍可旋转）；当角色碰到边界时，只会将碰到的方向上的速度设为 0。

角色射击：

玩家从游戏开始就拥有射击能力。子弹的图案设定为小版的玩家图像，当子弹碰到墙壁、

场地边缘或其他玩家时会消失。子弹碰到对手时，会对对方造成伤害。

角色技能：

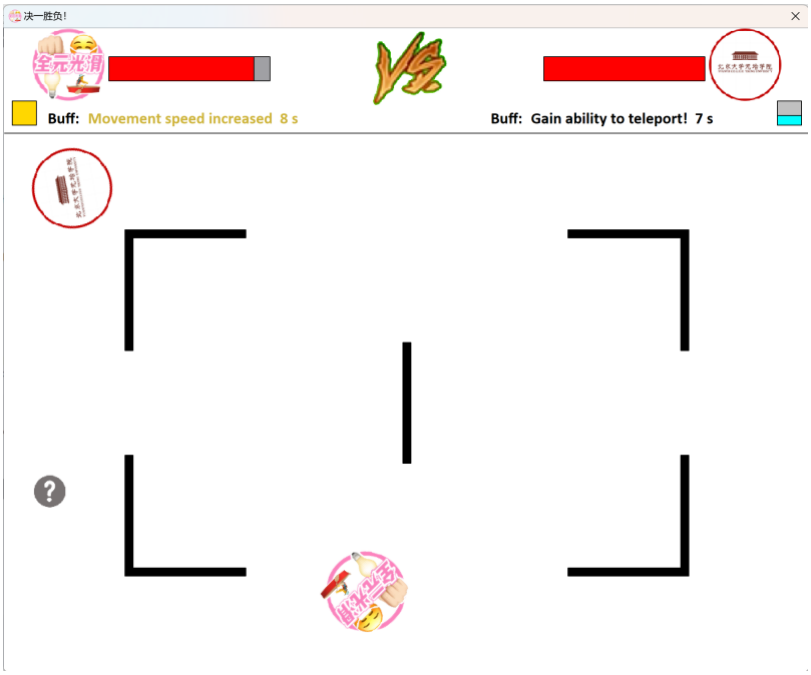
每个角色都有自己的独特技能，正确的使用时机可以改变战局。当技能冷却完成时，技能条将显示满格金色，并有"反光"动画效果提醒玩家。在技能持续时间内，技能条的金色部分会逐渐减少，技能结束后进入冷却，技能条将显示逐渐增加的水绿色，以提示冷却时间。

道具和 Buff：

在游戏中，场地上会随机生成道具，玩家接触后可获得随机的增益效果，同时状态栏会显示当前的效果。道具的图案为带有问号的灰色球体，玩家最多只能拥有一个增益，新的增益会替换旧的增益。稀有道具具有更强的增益效果，图案和状态栏的描述都会变为金色。目前有 7 种增益，包括增加移动速度、子弹射速、转速，提高子弹伤害，缩短子弹间隔，获得传送能力，以及回复生命值。

游戏结束：

当任何一方的血量降至 0 以下时，游戏结束，另一方获胜。此时会弹出结算窗口，显示胜利者。玩家可以选择退出游戏或返回主菜单。



三、类设计细节：

Game （游戏主体对象）

Game 继承了 Qt 中的 QGraphicsScene 类，是为游戏主体的核心对象。

Game 类的作用是控制键盘交互，游戏内状态栏的刷新以及所有与游戏物体（玩家、子弹、道具，下同）交互，控制添加、卸载和刷新，并实现动画。 这主要类内的槽函数 game_update() 实现。类内的 game_start() 函数保证游戏开始时变量和状态能被良好地初始化，新建墙体，并且开始一个全局计时器，以 20 毫秒为周期调用 game_update() 函数，故游戏实际上每 20 毫秒会刷新一次，下面我们将 20 毫秒称为 1 游戏刻。

为了方便控制，在 Game 类中我们创建了一个 QList<Object*> 类的指针链表

existing_objects 来存储所有游戏物体。Object 为所有游戏物体的基类，继承自 QGrpahicsPixmapItem，派生出 Player(玩家), Bullet(子弹), Prop(道具)类，下面会详细介绍。

game_update() 函数会依次完成以下操作/功能:

1. 根据键盘交互对玩家进行移动（向前，向后，旋转），调用了 Object 类提供的 change_velocity 和 change_angle 方法
2. 判断是否发射子弹，成功发射会在 existing_objects 中创建新的 Bullet 对象（Object 派生）。这里会透过 Player 类的成员变量 is_using_skill 和 faculty 判断玩家是否在使用技能且角色为元培/新雅，实现它们的技能效果。
3. 判断是否是使用增益提供的传送功能。
4. 判断是否使用技能，使用会调用 Player 类提供的 use_skill 方法
5. 更新技能条，并检测玩家的 skill_duration 和 is_using_skill 成员变量判断玩家的技能是否持续时间已过。如果已过，则调用 Player 类提供的 skill_expired 方法
6. 判断技能条是否已满，并透过用于计时的变(on_cooldown_cnt_p1 /on_cooldown_cnt_p2)，实现技能满格时的反光动画。
7. 透过键盘交互，判断玩家是否既没有向前也没有向后操作。如果是，会调用 change_velocity 给予一定减速度，实现速度逐渐减少为 0 的效果，符合物理。
8. 对两个玩家调用 Object 类提供的 object_update 方法（会传入 existing_objects），对玩家一些状态进行更行并重绘。
9. 透过 Player 类的 is_hurt 成员变量检测玩家是否受伤。如果是，则触发受击动画，并调用 Player 类的 on_hurt 方法
10. 根据 Player 类的 buff_id, buff_duration 等成员变量更新 Buff 状态栏 （每秒更新一次）
11. 对玩家外剩下的游戏物体，调用他们的 object_update 方法，并检测他们的 is_deleted 变量，如果是，则删除该物体并释放内存。
12. 更新血条
13. 透过玩家的 is_deleted 检测游戏是否结束。是则停止全局计时器，并弹出结算界面。

动画效果

游戏主体有技能满格时的“反光“动画和玩家的受攻击动画两个动画效果，以下是实现方式:

反光动画：使用 QLinearGradient 类实现一个从原色（金色）到白色再到金色的渐变效果，渐变效果发生的位置会随时间改变，从而实现动画效果。

受击动画：对玩家调用 QGraphicPixmapItem 的 setOpacity 函数，控制透明度。

血条和技能条

血量条和技能条的构成方式比较相似。每个模块实际上由两个 QGraphicsRectItem 组成：一个灰色为填充色的固定的矩形，还有覆盖在其之上的红色（血条），水绿色/金色（技能条）的矩形，且每个游戏刻会根据玩家状态变化进行更新。

Object （游戏物体）

Object 类继承自 QGraphicsPixmapItem，是所有游戏物体的基类。游戏物体的一些共有属性被储存在了基类，例如速度、半径、角度、is_deleted 等等。Object 的构造函数调用了

Game 作为一个 QGraphicsScene 的 addItem 函数，所以每当新增一个游戏物体时，它将自动被添加到游戏场景中。构造函数还设置了游戏物体的变换中心点，让旋转功能能正常运作，同时给予每个游戏物体一个唯一的 id (玩家 1 和 2 固定 id 为 0 和 1)，这样透过 id 便能判断两个 Object 对象是否一致。

Object 提供了两个方法 change_velocity 和 change_angle 对共有属性进行修改，也提供了两个可以在派生类被重写的虚函数 object_update （用于更新游戏物体状态并重绘，包括碰撞检测）以及 player_collide 函数（用于执行和玩家碰撞后执行的语句）

考虑到当前所有游戏物体都为圆形，我们选择自己实现碰撞检测，透过作为参数传递的 existing_objects 遍历所有游戏物体，检测彼此的半径之和是否大于两者中心点的距离。如果是，则代表碰撞发生

与此同时，由于游戏场地中的墙体是基于 QGraphicsRectItem 绘制的而非 QGraphicsPixmapItem，故不能作为 Object 的成员。故我们在单独实现了一个 judge_wall_collision 方法，以判断游戏物体是否与墙体发生碰撞。

Player （玩家）

Player 是玩家的类，继承自 Object 类，还派生出了四个派生类：Eecs, Guanghua, Yuanpei, Xlnya，代表着四种不同角色。

Player 拥有院系、血量、最大血量、技能冷却、Buff Id、Buff 持续时间、攻击间隔、子弹半径、子弹大小等属性，方便更新时执行各种操作。

Player 类重写了 object_update 方法，在 Object 类的基础上添加了发生碰撞时调用对方的 player_collide 方法，确保碰撞触发的事件只会发生一次。例如，当玩家和一颗子弹碰撞时，如果不重写，可能会导致玩家检测到和子弹碰撞掉一次血，子弹检测到和玩家碰撞时又扣玩家一次血的情况发生。使用虚函数保证了只会在玩家检测到与其他游戏物体碰撞时才会发生一些事件。

Player 类拥有 grant_buff， remove_buff 两个方法，用于添加/删除增益。

又拥有 use_skill, skill_expired 两个虚函数，用于执行使用技能/技能使用完毕时的语句。

on_hurt 也是一个虚函数，用于控制玩家受击执行的语句，设为虚函数使得不同类型的角色能有不同的效果（例如大信科有概率免疫伤害）

Bullet （子弹）

Bullet 是子弹的类，继承自 Object 类，添加了伤害值和 Player *parent 两个属性，parent 存放了子弹的发射人，用于碰撞检测（设定为自己的子弹不会打伤自己）。

Bullet 类重写了 object_update 方法，让子弹发生碰撞后即将 is_deleted 设为 true，从而在该游戏刻中就被移除。同时也重写了 player_collide 方法，使子弹对被碰撞的玩家造成伤害，即调用该玩家的 on_hurt 方法。

Prop （道具）

Prop 也继承自 Object 类，用于道具对象。Prop 类添加了 is_rare 这一布尔值变量，表示该道具的 buff 是否为稀有。

Prop 类同样重写了 object_update 方法，让道具和玩家发生碰撞后即将 is_deleted 设为 true，从而在该游戏刻中就被移除。同时也重写了 player_collide 方法，使道具给予被碰撞的玩家一个随机 Buff，即调用该玩家的 grant_buff 方法。

ShapedWindow 类（游戏窗口）

ShapedWindow 类是一个从 QMainWindow 继承的子类，专门用于创建异形窗口，这个类通过设置窗口为无边框样式并定义样式表来实现异形窗口。同时，通过重载鼠标按下、释放和移动事件，实现了窗口的拖动功能。

MyPushButton 类（游戏按键）

MyPushButton 类是一个从 QPushButton 继承的子类。此类封装了一些不规则按钮的关键功能，例如设置遮罩、消除边框等。MyPushButton 类提供了两种按钮实现方式：用于创建具有浮动效果的按钮和没有浮动效果的按钮。也使用 QPainter 解决了图片边缘粗糙和锯齿化的问题，使得图片和按钮看起来更加清晰和光滑。此外，其中的 enterEvent 函数使得在鼠标进入按钮区域时，改变按钮的图像为 pressedImgPath，同时保持抗锯齿的效果。这增加了用户交互的视觉效果，提高了应用的友好度和趣味性。

Choose 类（角色选择）

"Choose" 是一个从 QDialog 继承的类，该类在 Qt 中用于构建对话框。此类在应用程序中定义了一个选择界面，界面中间部分展示了角色技能的介绍，用户可以在这个界面上选择角色，并在角色选择完毕后开始游戏。、它创建了一系列的按钮（用 MyPushButton 类实例化），每个按钮都与一个特定的角色关联，并在屏幕的指定位置显示。。每个按钮都连接了一个 lambda 函数，该函数在按钮被点击时执行，使相应的角色被选中，并在已选角色栏中显示选中的角色的新实例。如果两个角色都被选中，将显示开始游戏的按钮。

Dialog 类（游戏介绍）

`Dialog`是继承于 `QDialog` 的一个子类。`Dialog` 类用于展示游戏的介绍，用户可以通过点击按钮来切换不同的介绍页面。其构造函数首先设置了窗口的图标，尺寸，标题，并初始化了一些变量，包括计数器 `cnt` 用来追踪用户当前查看的是哪一页介绍 on_NextButton1_clicked 函数是 `NextButton1` 被点击时会调用的槽函数。在这个函数中，首先增加了 `cnt` 的值。然后，如果 `cnt` 的值等于 2，就将 `NextButton1` 的文本改为 "退出"。如果 `cnt` 的值等于 3，就删除当前的 `Dialog` 对象。否则，就调用 `update()` 函数，它会导致窗口的内容被重绘，从而显示新的介绍页面。

四、图像制作

游戏的图像制作主要使用 Photoshop 和 PowerPoint 来设计和布局窗口元素，如背景、个性化的按钮和标签等。利用 Photoshop 创建和修改游戏中的各种图形元素。PowerPoint 则主要用于组织和布局这些元素，通过其强大的布局和动画功能，可以快速构建和预览效果图。

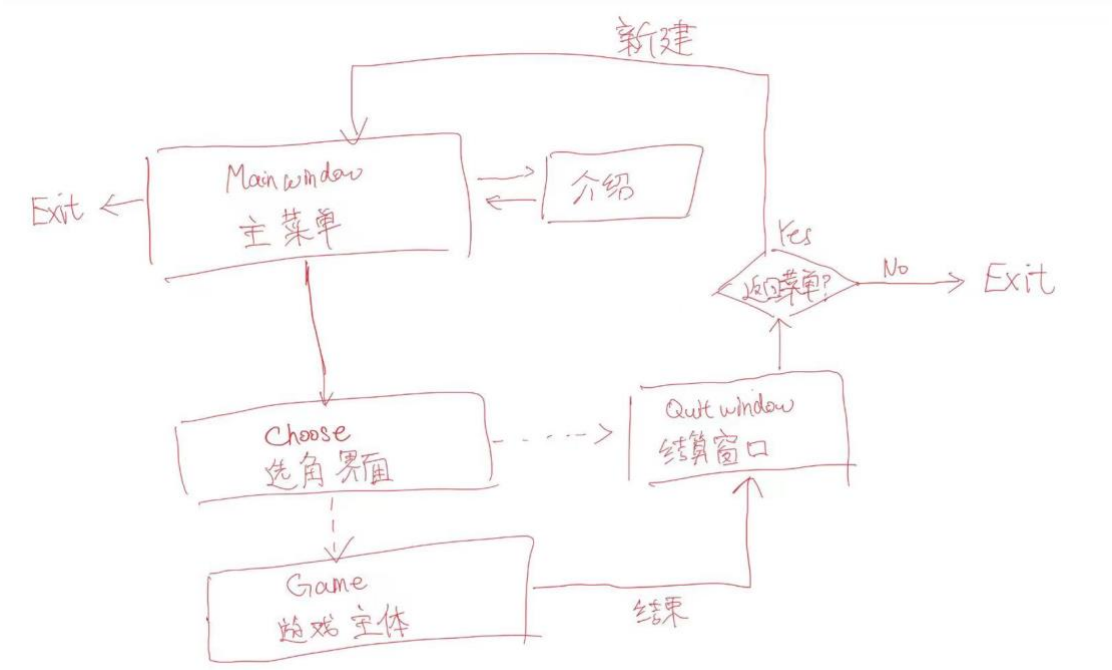
五、窗口跳转和交互

本程序运行时使用指针来动态创建和卸载窗口，在窗口对象新建的时候初始化，保证了

应用程序不会跳出多个窗口。

值得注意的是，当在 Choose 界面（选将界面）点击进入游戏时，程序会新建一个 QGraphicsView 对象赋值给一个 Choose 类内定义的 QGraphicsView* v 指针，并新建一个 Game 类（透过构造函数传入玩家的选将，和父组件，即该 Choose 类对象）对象与 v 绑定，并覆盖了 Choose 类本身的界面。所以，游戏主体实际上是在 Choose 界面上运行的。

Choose 类定义了一个 game_end 函数，在 Game 类中 game_update 的检测游戏结束（第 13 步）为真时触发，它会控制结算窗口的弹出，并且在点击返回菜单时，会卸载该对话框，Game 对象，和 Choose 对象并新建一个 Mainwindow 对象并展示，从而实现游戏重新开始。



六、小组分工

吕嘉楠：游戏核心架构搭建、游戏主体、报告撰写、视频录制
罗珑赞：对战地图与图标设计、血条设计、报告撰写、视频录制
李明隆：主菜单、开始、选将、介绍、结算界面、图标设计

七、项目总结与反思

反思：

- 一定要记得给指针初始化或者卸载后设为 nullptr。开发过程中有许多次崩溃都是因为指针指向了未知的地方，并且注意内存泄漏的问题。

- 注意头文件之间相互包含的问题，即使是加入了 ifndef 声明。善用类声明 class XXX; 解决需要不同类需要相互使用的问题。

- 不要在头文件里定义全局变量，很容易导致重定义的问题。

- 多将数值设置为可以在 cpp 文件开头修改的常量，否则调参时很容易忘记将某个地方的数值修改导致出错。

- 组员间各自电脑的操作系统，Qt 版本、配置都不尽相同。当我们将各自代码整合在一起时会产生各种问题，如提示配置文件出错、版本不一，甚至直接无法打开项目，花费了不少时间。因此，在开发刚开始的时候我们其实就应将版本、配置等环境都沟通好。并为项目整合预留更多的时间，避免开发过程中产生的各种麻烦。

- multimedia 模块的缺失使得无法使用 QSound 播放音乐。在尝试使用 C 语言的 windows.h 和 mmsystem 库的 mciSendString 函数时，由于字符串编码问题无法正常运行程序，因此只能使用绝对路径，考虑到代码的兼容性与可移植性，我们将音乐部分删除

总结：

本次作业，我们成功基于 QT6，运用面向对象编程实现了一个内容比较完整，外观比较美观的小游戏。运用多态方法，也达成了相当程度的可拓展性。这同时锻炼了组员们面向对象编程能力以及图形化界面设计能力，并给予我们进行多人工程的宝贵经验和教训。