**斗地主-----基于Qt6框架**

1. **项目分工**

****

1. **界面ui及游戏控制**

a.界面的初始化：设置背景图片和音乐、定义各种图片和按钮形象和位置，牌堆的显示、抢地主不抢的动画、发牌动画

b.总控制：控制玩家轮流出牌、实现点击开始按钮后(抢地主、出牌、超时不出、菜单项)后一系列操作，实现卡牌初始化、洗牌、发牌和排序算法，卡片和图片绑定

**2、牌型算法**

列举所有牌型，判断牌型、设计权值、设计拆牌算法、判断当前卡牌是否能打出

1. **出牌逻辑**

当前玩家选择出的牌是否大于上家出的牌，ai出牌、判断输赢、ai为地主时出牌、根据牌型出牌

**小组成员：**

王瑶瑶：牌型算法、出牌逻辑

伍静茹：对界面初始化的初步定义，Ai算法

刘予：优化ui界面和游戏总控制

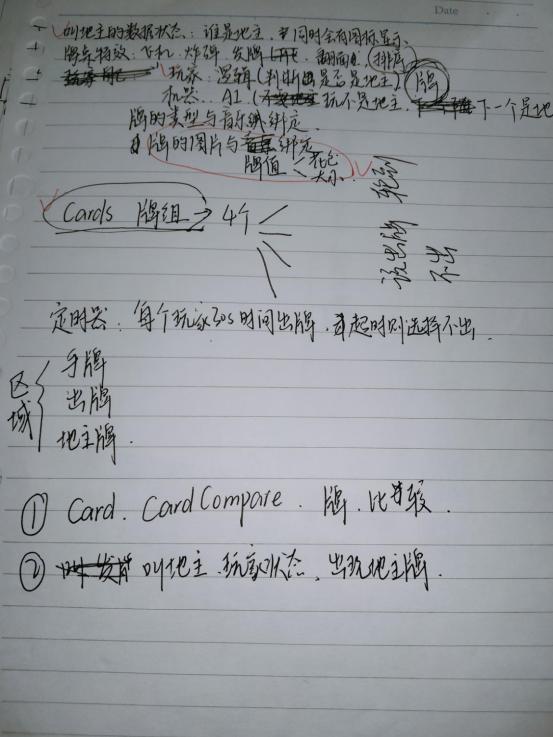
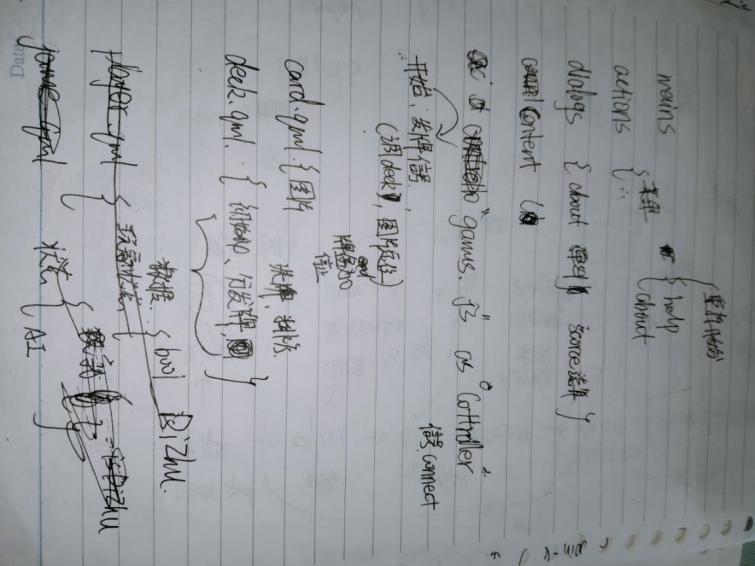
1. **项目总体设计与开发过程**
2. **总体设计**

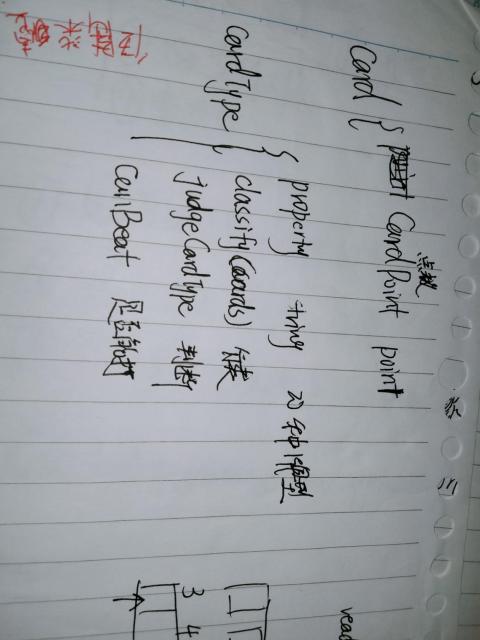
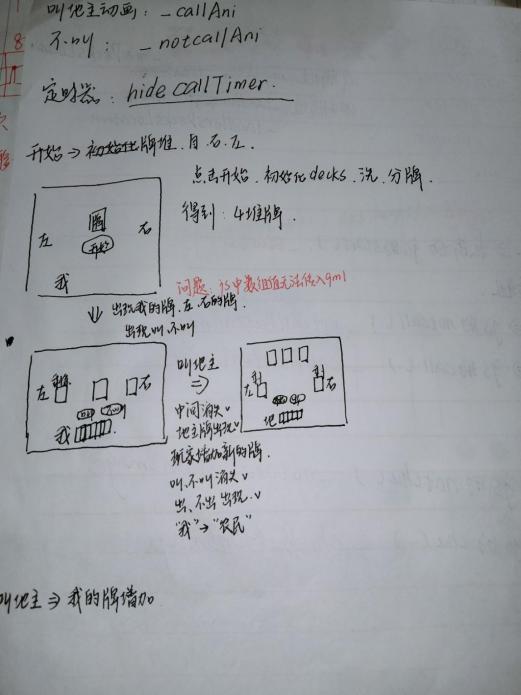
定义单独牌的属性，将54张牌添加到牌堆里，实现洗、发牌，每个玩家拿到手牌。开始检测谁是地主，如果玩家不叫地主，下一个电脑玩家就是地主，地主状态发生改变。

游戏控制：获取玩家的手牌、地主牌，监测堂子是否有人打出牌，监测用户玩家要出的牌，获取到玩家要出的牌，判断当前牌是否符合常规牌型，如果不符合就不允许用户玩家出牌。

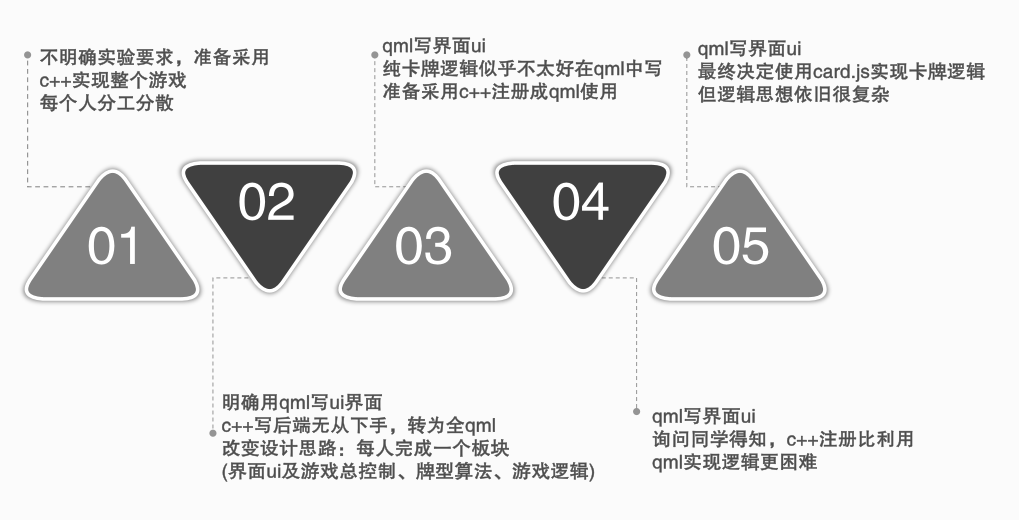
AI策略：给电脑玩家制定游戏策略，先检测手牌并归为某一类牌型，尽量让顺子、连子等特殊牌型留着等到电脑玩家主动出牌时出出去，再查看当前手牌是否有能压上家的牌。

判断游戏输赢：实时检测三个玩家手上的牌数量，当某个玩家的牌数为空时，游戏结束。

****

****

1. **开发过程**

****

I、成员一：王瑶瑶

->>>6月18

使用c++实现项目，创建cardNode来实现比较同类型的大小，这个时候打算给每个牌型设置权值，用权值来比较牌面的大小并且在电脑玩家出牌时可以根据权值来判断当前最佳能够出的牌。并且初步定义当前牌类型的判断函数，将特殊的牌（比如三带一，飞机带二）设定主牌、副牌，便于后续的主牌比较。创建cardType来用于判断当前牌的类型。

->>>6月19

开始测试牌的比较，设置特殊牌的权值列表，检测到特殊牌，权值就开始添加。测试结果并不如我所想的，在添加任意牌型后进行测试，最终得到的权重值都是一样的，权重的逻辑太复杂，改变思路，利用牌自身的牌面大小进行比较，并且为了不到处调用函数，在一个cpp里面实现。

重新分工，并且搭建了一个基本的项目框架：Game类、Card类、CarType类、Cards类、AICard类、GameControl类来实现整个项目。此时只考虑到了逻辑的实现，并没有和试图结合。

->>>6月20

在测试卡牌类时（Card），意识到牌只能在终端输出不能和试图相结合，所以开始使用qml来完成卡牌的一些定义和逻辑。Card.qml实现卡牌的定义和与图片资源的绑定，比如（大王的图片资源为“5-1”）将“5”作为卡牌的花色，“1”作为牌面，尽管逻辑上实现类图片和卡牌的绑定，但是在界面输出时并没有能够让卡牌的图片显示出来，并且qml大量的写了function关于卡牌的逻辑和卡牌类型的判断，没有检查运行效果。

->>>6月21

重新搭建框架。尝试直接用纯qml来实现项目，添加Card.qml、Cards.qml、CardType.qml、Handcard.qml、Player.qml、Robot.qml、UserPlayer.qml,试图用这几个qml来实现添加牌、删除牌、清空牌、洗牌、更新牌的操作。开始实现电脑玩家的操作，此时发现电脑玩家的操作需要使用线程来处理当前状态和是否出牌，根据牌型的权重来考虑是否考虑抢地主，参照其他项目来实现的机器人策略。发现问题。

->>> 6月22

qml是用来做视图的，对于对象的创建并不能用qml表示出来，选择javascript实现逻辑

->>>6.23

改变使用javascript做项目底层逻辑开发，再次重新搭建项目逻辑，创建card.js、game.js、player.js来实现卡牌逻辑，卡牌的类型判断，实现测试函数开始测试，无太大的问题。但是在创建时，直接使用class来创建对象，此时思路开始混乱，c++语言和js混乱的使用，发现对于创建对象时c++语法上更加明确，思路卡住，决定换成c++做底层逻辑开发

->>>6.24

改变设计思路，每一个人完成一个功能板块，由于之前做的太过分散，不能结合起来实现一个完整的功能。创建playcontrol.cpp控制游戏逻辑，在实现的过程中发现需要用emit来发射信号完成交互，并不能很好的继续实现下去。

->>>>6.25

继续使用js控制整个游戏流程，思路上想得过于复杂。在与其他小组讨论交流过程中，得知对于人机而言，javascript的实现比用c++实现更小。最终确认用javascript来完成逻辑实现。

->>>>6.26

重新用javascript完成卡牌逻辑实现，在实现的过程中由于没有考虑到与界面交互的问题，就直接为每一个Card定义来suit和rank,并且检查并完成了获取牌的花色、获取牌的值、判断牌的类型、获取特殊牌型最大的一张牌（连子、连对）、查看地主权值判断是否抢地主、检查当前牌是否能够比上家的牌大能否打出（利用牌型先判断当前牌与上家的牌是否相同，再让一个数组获得相同牌出现的次数，对里面的牌值进行排序，再获取最终的最大的牌），进行测试。尝试将card.js的牌传入到主qml里面，试着给牌赋动画效果，但是失败。

->>>>7.6-7.9

初步完成一点电脑玩家需要的牌的算法Model类型，实现拆牌效果，将电脑玩家手上的牌拆分成各个特殊类型，用于在电脑玩家主动出牌时的逻辑，提升可玩性。尝试用画布将每一张牌画出来，这样利于传入卡牌的花色和牌值，在显示效果并没有得到想要的结果。后续几天都在实现如何将卡牌的数据传入试图里面。此时发现逻辑和试图的交互基本没有实现。

**最终的功能实现**：定义卡牌和卡牌属性、列举卡牌类型、

判断卡牌的类型：（通过牌的长度检测一些特殊简单牌型：单张牌、炸弹、三带一、对子;通过数组来获取相同牌的次数来检测当前特殊复杂牌的类型：顺子、连对、三带二、飞机、四带二单、四带二对、飞机带单、飞机带对）

获取牌的最大值：计算当前牌相同牌出现的频率，并获取所有出现的牌进行排序，去掉重复出现的牌

拆牌：将手牌按照策略拆分成不同的组合，用于AI在主动行牌的时候选取哪种更适合最佳选择

检查当前牌是否能出：根据判断牌的类型检查当前的牌是否能出

II、成员二：伍静茹

->>>6.18

最初想要通过c++实现时，创建cardType来用于判断当前牌的类型。

编写了cardtype的代码，设定了牌的花色和大小，试图将牌的图片与设定的花色大小绑定。

->>>6.19

重新分工，并且搭建了一个基本的项目框架：Game类、Card类、CarType类、Cards类、AICard类、GameControl类来实现整个项目。修改了CardType类，在card类重新对卡牌的花色和点数进行定义。

->>>6.20

完善对Card类中花色和点数的定义和存储，测试比较牌面，同时发现需要与牌的逻辑相结合，将此部分转交给王瑶瑶来与游戏逻辑一同编写，转为研究ui界面，实现了叫地主按钮和不叫地主按钮鼠标悬停和点击效果，通过刘予实现的最初ui界面，定位按钮位置，以及点击按钮后动画位置

->>>6.21

重新搭建框架。尝试直接用纯qml来实现项目，添加Card.qml、Cards.qml、CardType.qml、Handcard.qml、Player.qml、Robot.qml、UserPlayer.qml。在player.qml中实现点击开始按钮，以及点击叫地主后玩家形象的变化，在点击不叫后，机器人形象的变化

->>> 6.22

不能使用Mousearea,转变为使用TapHandler来实现对叫地主等图片实现的点击效果。

->>>>7.6-7.9

初步完成电脑玩家主动出牌时的逻辑，提升可玩性。

III、成员三：刘予

->>>6.19

设置了游戏背景图、分数显示栏、中间卡牌、开始按钮悬停离开动画

将最小尺寸和最大尺寸的宽高设置为一样，保证游戏窗口界面大小不会因拖放而修改;利用列布局组件将玩家所有人得分显示在界面右上角、定义了开始按钮在鼠标悬停与离开时不同的图片效果。

->>>6.20

创建deck.qml，有三个玩家的牌堆、地主牌堆、对牌进行初始化、洗牌、发牌、排序

初始化：创建54张牌，单独创建大王、小王，利用双重循环添加其余52张牌的花色和数值;洗牌：创建一个随机数，把牌的顺序打乱;发牌：将最后三张牌作为地主牌、其余51张牌根据j%3得到的余数分别将卡牌添加到我、右边玩家、左边玩家的数组里;排序：利用冒泡排序对所有的牌堆进行排序。

->>>6.21添加了菜单项、添加了game.js作为游戏总控制

菜单项：开始、退出和关于

开始与界面上的开始按钮关联、点击它页面上开始按钮也会消失，玩家遇到不好的牌也可以点击此按钮重新发牌;退出就是退出整个程序、关于是跳出一个弹窗（但未实现）

->>>6.24对ui、js进行修改、发牌动画

运用精灵写了发牌动画、也用定时器但无法控制在开始按钮之后实现，花费时间过长;将Deck.qml里的牌的初始化内容移动到js;把前一位同学定义的按钮冗余部分删除

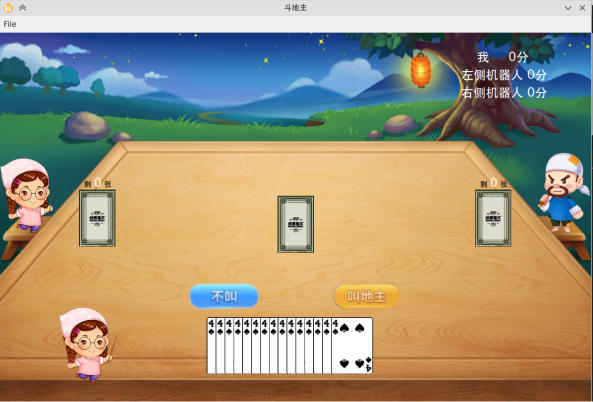
->>>6.25-6.26思考怎么利用定时器实现了不叫动画和叫动画、以及人物形象的变化（初步思路是我叫地主即为地主，不叫右边玩家为地主），以及怎么通过发送信号接受信号处理

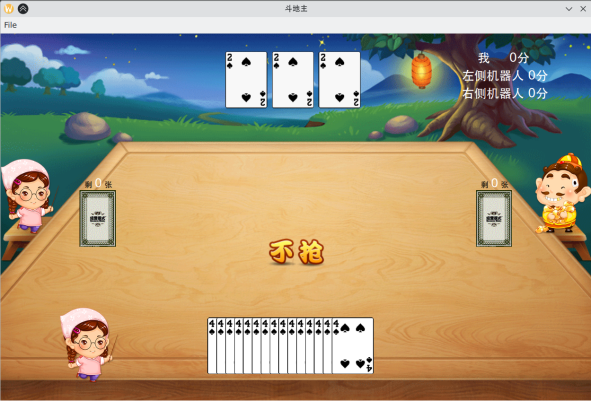
->>>7.5--7.9 将window.qml内容移到Elements.qml,增加了左右玩家的牌堆，背景音乐的播放，定义了出牌与否的组合按钮，开始游戏玩家农民形象的随机性，游戏控制玩家超时不出即自动过牌、创建了三个玩家对象，但轮次交换无法实现。

叫与不叫、出不出牌按钮的组合切换：利用堆叠顺序和图片的可视决定;叫地主不叫地主分别触发控制器的call()和notcall()，控制器再对哪些组件存在与否做出判断，实现页面交换;玩家农民形象的随机性：利用randomfarmer()返回人物的随机图片路径赋值给玩家资源;玩家超时不出：利用定时器buchuTimer，在玩家点击叫地主/不叫按钮时触发开始，且在玩家点击出牌/不出按钮之后触发停止，该定时器触发之后等待3秒不出就自动执行notcall()，实际玩家每轮思考时间应为30s,此处是为了测试方便。创建了Player.qml,实现对玩家类型的创建，但只是一个笼统的创建，具体参数传递还无法解决。

1. **项目基本实现**
2. **基本界面ui**

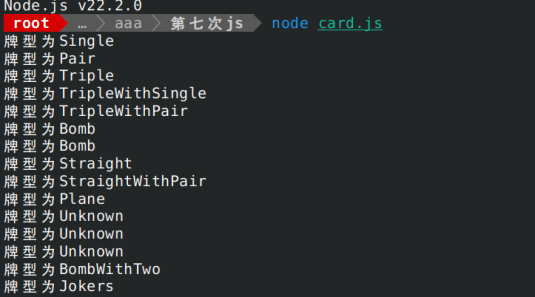
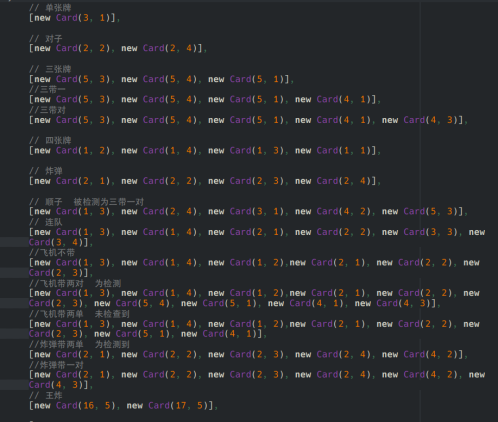
背景图片和音乐、玩家农民形象的随机、牌堆和出牌区域、开始按钮、叫与不叫地主按钮和出不出牌按钮及其响应效果

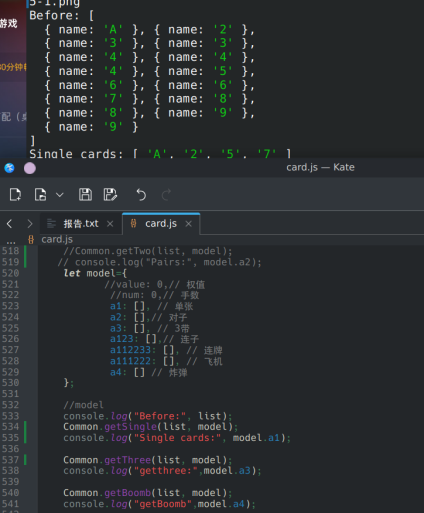
****

****

1. **牌型**

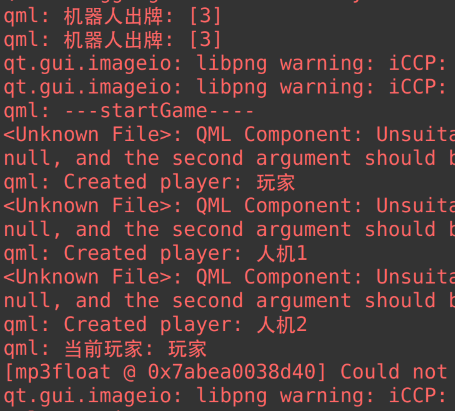
牌型的定义，判断牌型是否符合，设置权重，拆牌算法、判断牌是否能打出

****

****

1. **游戏控制器game.js**

各种按钮点击之后出发的事件和图片的可视效果、对牌的初始化、洗牌、发牌、发牌和排序算法

****

****

1. **出牌逻辑**

当前玩家选择出的牌是否大于上家出的牌，根据牌型出牌

1. **出现问题及解放方法**
2. **牌的逻辑与ui界面交互弱**

办法一：将牌的命名（name：1-4.png）直接选用图片命名，获取图片第一个字符“1”为花色，第二个数字“4”为牌面。

办法二：创建一个牌组将所有的牌列举出来，用画布将牌对应的suit和rank画出来

两种方式均未实现参数成功传递的问题，虽然检测时两种都能显示对于牌但是无法与逻辑进行连接

1. **参数无法正确传入**

在js定义的全局数组无法在qml中实时更新，qml中访问得到数据为空，运行时间上出现差异导致传递参数不是我们真正想要的参数

1. **牌的逻辑存在bug**

对于某些特殊的牌型无法检测到（炸弹带两单，飞机带两单、飞机带两对），还未找出错误

牌型的检测：首先获取当前牌的张数，牌的张数小于4张很容易获得是什么类型，主要问题在于当牌张数过长，例如飞机，并且主要的主牌存在三张牌一样的情况下，需要先用一个数组获取相同一张牌出现的次数，再通过排序获取连牌，例如连子连对的主牌最大一张，用于比较，通过出现次数可以基本获取牌的类型，比如a[3].length===1,表示相同牌出现四次的有一个

1. **调用问题**

游戏是玩家自己与人机对战，不知道怎么利用qml在js中创建的多个对象进行控制

1. **牌的点击事件处理**

玩家需要点击牌选择即将出的牌，点击牌面一次，牌面上移，再点击一次，牌面回到原始位置。起初是用taphandler处理，由于牌的显示是一张叠一张，导致玩家点击一张牌，这张牌下方的牌也会被点击起来，无法实现只点击一张牌，于是转为使用mouseArea处理该事件。

1. **心得体会**

1,整个过程对于时间把控并不合理。小组由于在选题时摇摆不定，导致实训开始后两天才开始确认选题

2,由于没有认真思考整个流程的逻辑和实现方式，导致小组成员在具体实现上很慢并且质量不高，成员在分配任务时将ui界面和打牌的逻辑两级化，在最开始并没有合理的结合ui和逻辑的交互，导致逻辑能在终端运行，ui能实现一部分但是没有逻辑

3,选题时没有根据小组具体情况选择，人机对战的斗地主的复杂性和逻辑严谨性比想象的高，超出成员的能力范围

4,对于qml、c++和javascript的语言优势把握的不够精准，在具体实现时对于语言的选择飘忽不定，小组最开始使用c++来进行开发底层逻辑发现某些交互的难度太大并且不能很好的运用，随后试图直接使用qml嵌套着逻辑，但是逻辑性和复杂程序无法满足需求，最终确认使用js，一周之后才确认用javascript实现卡牌的逻辑,在限有的时间内增添了负担

5,小组的水平和对代码的熟练度并不高，一个地方卡住导致整个进度被迫停止，在将牌的逻辑传到qml显示界面时死死卡住，没有找到合理的办法实现

6,整个设计思路存在问题，逻辑交互和页面显示一些设计思路并不能合理的实现项目需求，导致在后续开发时束手无策

7,没有合理的进行分工合作，小组成员大多时候是在完成自己的功能板块，到最后才开始进行交流发现各自的板块都存在很多的问题，并且没有及时解决，到最后已经达到了整个项目需要重构的情况

8,小组内最终知道项目无法实现，产生消极的想法，进度也始终无法推动

希望在下一次的协同合作开发时能够吸取教训成功实现一个项目