**基于QT的麻将程序设计作业报告 (1)**

**陈子睿 2200012913**

**一、程序功能介绍**

由于之前没有接触过日本麻将，不太清楚日本麻将的规则，基于最通常的麻将规则及部分四川麻将规则实现了本程序作业。另一位组员写了一个基于日本麻将规则的程序，实现了日本麻将的部分功能。

**（一）界面设计**

游戏界面采用经典的布局设计。顶端为功能按钮，包括：游戏界面、系统设置，役种列表、使用帮助和退出游戏按钮。左侧根据不同的功能按钮显示对应的子功能按钮。界面其他空间用于麻将桌布局。



**（二）主要游戏功能**

实现了麻将的各项主要功能：

1. **抓牌和打牌**

点击洗牌按钮后，可以选择东南西北不同玩家开局，选择的玩家在右上角标识。根据对局情况进行分值的计算。结束对局后重新计分。

根据逆时针抓牌顺序，按照规则，从牌池中自动抓牌。AI玩家可根据选定的不同算法自动打牌，自动吃牌、杠牌和碰牌。手工玩家可手动选择要打出的牌。每次抓到的牌，红色遮罩突出显示，然后按顺序放置在手牌序列中。拟选打出的牌立起突出显示。打出后的牌放倒弃牌堆中。

1. **吃牌**

下家若可以吃上家的牌，自动出现吃牌提示，可吃牌，也可过牌。吃的牌放倒后放置在手牌的最左端。

****

1. **碰牌**

任何玩家可碰当前打出的牌，手工玩家可选择过牌。碰的牌放倒后放置在手牌的最左端

****

1. **杠牌**

可以支持明杠、暗杠、杠手中牌。杠的牌放倒后放置在手牌的最左端。



1. **听牌**

玩家听牌后，出现听牌标志，显示可胡的牌。如下图玩家北，听的牌为二万、五条。

****

1. **胡牌**

根据规则胡牌，胡牌后计算分值。目前简化处理，赢家得分10分。本游戏实现了四川麻将的血战到底模式，一家胡牌后，剩余玩家可继续对战。

1. **撤销操作**

可以撤销打牌的各类操作，一直恢复到初始状态，包括：各个玩家的手牌、牌墙和弃牌堆的各种状态。

****

**（三）规则设置**

该界面设置游戏的所有规则。可以设置以下规则，重新开局后，规则生效。

1. **码牌**

目前实现了两层码牌。

在麻将牌的显示上支持以下三种方式：（1）设置显示所有麻将牌，包括各个玩家的手牌，待抓的牌，主要用于演示。（2）设置显示AI玩家的手牌，用于观察AI玩家的打牌策略的智能程度.（3）只显示自己的手牌和打出的弃牌，其他牌不可见。

1. **AI玩家设置**

AI玩家实现了低智商、中智商两种。（1）低智商选择手牌中的倒数第二张排。（2）中智商根据不同牌型对每一张手牌进行赋值计算，选择打出分值最低的牌。赋值的排序为：刻子、顺子、对子、连牌。高智商可通过神经网络等实现更复杂的算法，可下一步实现。

1. **抓牌设置**

总麻将牌数可设置为136、108。玩家人数目前支持4人。跳牌方式包括：隔跳、平跳、上下跳。

1. **打牌设置**

可选择是否支持：碰牌、吃牌、杠手中牌、明杠、暗杠。

1. **胡牌设置**

可选择是否支持：只限自摸、抢杠胡、小七对、二五八将、缺一门、血战到底。

****

**（四）主题设置**

可设置选择背景音乐。可选择不同的麻将桌背景样式。



**（五）规则说明和帮助**

役种列表：这里是从网上查到的日本麻将各役种的说明，本作业没有实现和判断这些役钟。

使用帮助：本游戏的设置界面和游戏基本功能介绍。

****

****

**二、项目各模块与类设计细节**

**（一）QT自动生成的主界面类**

class Ui\_UIMainWindows

{

public:

    QWidget \*widgetTitle;

    QHBoxLayout \*horizontalLayout

    QLabel \*labIco;

    QToolButton \*btnMain;

… …

}

**实现QT定义的界面类和实现功能的类关联起来：**

UIMainWindows::UIMainWindows(QWidget \*parent) :

QWidget(parent),

ui(new Ui::UIMainWindows)

{

ui->setupUi(this);

… …

}

**（二）定义麻将类，包括状态以及主要操作动作**

**class Mj{**

**public:**

**int getId() {return id;}**

**void setId(int id) {this->id = id; }**

**int getResult() const{return result;}**

**void setResult(int result) {this->result = result;}**

**int getPaizhuangtai() const {return paizhuangtai;}**

**void setPaizhuangtai(int paizhuangtai) {this->paizhuangtai = paizhuangtai;}**

**private:**

**int id; int result; int paizhuangtai;**

**… …**

**}**

其中，id和result是最重要的两个属性。id为108张牌或者136张牌的唯一ID;result为麻将牌的对应数值，同一花色和数字的牌，其result相同，对应关系如下：

**1-9，对应1万-9万；**

**11-19，对应1条-9条；**

**21-29，对应1筒-9筒；**

**31-37，对应东、南、西、北、中、发、白**

paizhuangtai标识牌的状态：

**0:正常 1:碰 2:明杠 3:暗杠 4:吃 5:胡牌**

**（三）定义系统配置类**

**class SystemConfig{**

**public:**

**//规则参数**

**bool isPrediction=false; // false 不显示预测牌，true显示**

**bool isAIshow=true; // false 不显示AI手牌，true显示**

**int AILevel; //AI智商级别**

**int AIPlayers=3; //AI玩家数量**

**int pongpai = 1; //碰牌**

**int chipai = 0; //吃牌**

**int gangshouzhongpai = 1; //杠手中牌(yong)**

**int fengpaizipaisanzhangkegang = 0;//风牌字牌三张可杠**

**int angang = 1; //暗杠(yong)**

**int minggang = 1; //明杠(yong)**

**… …**

**}**

**（四）基于QWidget生成的UIMainWindows类**

该类中定义了麻将手牌属性、各种操作的判断、界面显示、各种操作功能。

**class UIMainWindows : public QWidget{**

**private://重要变量**

**std::list<Mj> majhongTotal;//全部的麻将**

**std::list<Mj> majhongRemain;//剩余的麻将**

**std::list<Mj> majhongTaken;// 拿走的麻将**

**//手牌**

**std::list<Mj> handNorthMjs;**

**std::list<Mj> handSouthMjs;**

**std::list<Mj> handWestMjs;**

**std::list<Mj> handEastMjs;**

**//弃牌**

**std::list<Mj> discardMjs;**

**int mopaiWanjia = -1;//0南 1东 2北 3西**

**int dapaiWanjia = -1;**

**//目前打出的牌**

**std::vector<Mj> playedPai;**

**//判断吃碰杠**

**bool isSouthChow = false;**

**bool isNorthChow = false;**

**bool isEastChow = false;**

**bool isWestChow = false;**

**//碰**

**bool isSouthPong = false;**

**bool isNorthPong = false;**

**bool isEastPong = false;**

**bool isWestPong = false;**

**//杠(明杠)**

**bool isSouthKong = false;**

**bool isWestKong = false;**

**bool isNorthKong = false;**

**bool isEastKong = false;**

**//胡**

**bool isSouthWin = false;**

**bool isEastWin = false;**

**bool isNorthWin = false;**

**bool isWestWin = false;**

**public slots:**

**//抓牌显示**

**void initSouthDrawDisplay(const vector<Mj>& ,int ) const;**

**void initWestDrawDisplay(const vector<Mj>& , int ) const;**

**void initNorthDrawDisplay(const vector<Mj>&, int ) const;**

**void initEastDrawDisplay(const vector<Mj>&, int ) const;**

**//吃、碰、 牌展示**

**void ChowDisplay();**

**void PongDisplay();**

**void KongDisplay();**

**void checkDisplay();**

**//听牌、胡牌展示**

**void readyDisplay();**

**void winDisplay();**

**private://功能函数**

**void clearVariable(); //清空变量**

**void loadWidgetList(); //控件较多，为了便于处理，将东南**

**西北相关label赋值给对应的数组**

**void draw(); //摸牌**

**void undoPai(); //回退牌**

**void endGame(); //结束游戏**

**void dapai(int pos); //打牌(正常)**

**void chowDapai(int pos); //吃 打牌**

**void kongDapai(int pos); //明杠 打牌**

**void AIPlay(); //AI打牌**

**int AIPosition(std::list<Mj>&); //AI判断打牌位置**

**void AIChow(); //AI吃牌**

**void AIPong(); //AI碰牌**

**void AIKong(); //AI杠牌**

**void AIHu(); //AI胡牌**

**//判断**

**void isChow(); //吃**

**void isKong(); //杠**

**void isPong(); //碰**

**void isWin(); //胡**

**void isReady(); //听**

**void chowOperating(std::list<Mj>&); //吃操作**

**void pongOperating(std::list<Mj>&); //碰操作**

**void isConcealedKong(std::list<Mj>,int); //暗杠**

**void kongOperating(std::list<Mj>&); //杠操作**

**void undoOperating(); //撤销操作**

**… …**

**}**

**三、小组成员分工情况**

我们小组共两人，我实现了上述所有游戏界面及对应功能。另一位组员写了一个基于日本麻将规则的程序，实现了日本麻将的部分功能。

**四、项目总结与反思**

**（一）项目总结**

**1.基于QT搭建图形界面框架，实现了主界面的菜单和功能页。**

包括主菜单栏、侧边栏功能，实现主菜单、侧边栏、展示区的互动。

**2.基于QT封装的类，实现了麻将桌的布局和游戏过程的动态展示。**

通过QWdget类的相关控件，布局游戏桌和麻将牌，通过遮罩、麻将牌的位置变化，动态显示游戏过程。

**3.定义了多个类，定义了麻将的属性和各项功能函数。**

如上所述，定义了MJ、SystemConfig、UIMainWindows等类，将类的功能函数和界面操作进行关联。

**4.实现游戏过程中的核心功能和界面的互动，包括吃、碰、杠、胡等操作。**

这是本次作业的主要工作，对打牌的动作进行判断，对每一个游戏操作更新对应的各种状态，并显示在界面上。

**5.实现电脑玩家的AI操作。**

根据麻将规则和当前状态，自动执行各类游戏操作。根据刻子、顺子、对子、连牌等不同牌型进行赋值，打出分值最低的牌。

**（二）存在的不足**

**1.类的封装需要进一步优化。**

UIMainWindows、MJ、SystemConfig等类的设计相对较单一。可以考虑定义MJ的list类，如MJList类，将相关属性和功能函数封装到MJ和MJList类中。

**2.AI算法需要进一步丰富。**

目前实现的AI算法较简单，后续可考虑基于人工智能课程相关算法设计多种打牌算法，综合考虑当前牌面已打出的牌，手中牌可胡牌型的概率，胡牌的番数等。

**3.根据麻将规则，完善各种操作的功能。**

由于对日本麻将规则不太熟悉，部分功能尚未实现。

**4．界面的自适应优化。**

当前界面中的牌主要通过QLabel控件实现，根据不同的显示分辨率，下一步可考虑通过栅格进行界面的重新布局，或者根据界面的分辨率动态设置每一个控制的位置和长宽值。同时可定义多种界面风格的QSS文件，实现界面风格的多样化。