# 8.扫雷完结版（2021.10.15）

该版本改进了:
1.点击按钮还是蓝色【已解决：地图应该再开始前初始化】
2.添加了扫雷图标
3.简化了代码
mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H


#include <QMainWindow>
#include <QPushButton>
#include <QLabel>
#include <QMouseEvent> //鼠标点击事件
#include <QTimer> //定时器对象
#include "Mine.h"
```

```
QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
 Q_OBJECT

public:
 QPushButton *MainButton[30][30]; //雷按钮
 Mine *mine; //地雷
 int retainx,retainy; //鼠标点击的位置
 int isStart; //是否开始游戏
 int uigrade; //难度
 int seconds; //用时

```

```
//定义需要用到的定时器（第一个用来计时，第二个用来鼠标点击）
int Timer1; //用来计算时间
int Timer2; //用来判断鼠标点击是单击还是双击
int ClickedState; //点击状态，常态不点击为0，单击对应1，双击对应2

MainWindow(QWidget *parent = nullptr);
~MainWindow();

void newGame(int difficult); //初始化游戏
void startTime(); //开始计时
void empty(); //清空之前创建的东西
void chordingShow(int x,int y); //双击的展示
void leftShow(int x,int y); //左键单击的显示
void showtime(); //更新时钟
void failShow(int ii,int jj); //失败的时候显示所有的雷
void ButtonClick(int i,int j); //按钮点击
```

```
 void showButton(int i,int j,int model); //点击按钮
 void timerSingleClick(); //定时器事件，处理单击任务
 int* reverseij(int i,int j); //保存当前坐标
 void mousePressEvent(QMouseEvent *event); //鼠标点击事件
 void mouseDoubleClickEvent(QMouseEvent *event); //鼠标双击事件
private slots:
 void timerEvent(QTimerEvent *event);
 void on_pushButton_clicked(); //笑脸按钮
 void on_action_2_triggered(); //菜单:初级
 void on_action_4_triggered(); //菜单:中级
 void on_action_5_triggered(); //菜单:高级
 void on_action_6_triggered(); //菜单:游戏说明

private:
 Ui::MainWindow *ui;
};
```

```
#endif // MAINWINDOW_H
```

Mine.h

```


#ifndef MINE_H

#define MINE_H



#include <QMessageBox>

#include <iostream>

#include <stdlib.h>

#include <time.h>



class Mine{ //扫雷的类

public:

 int map[30][30]; //雷图，实际图（雷表示-1）

 int w_map[30][30]; //显示出来的地图
 int lei; //右键旗的次数
 int lei_all; //雷的个数
 int success; //成功全部雷即为胜利
 int y_N,x_N; //map[y_N][x_N]
 int grade; //等级(初级，中级，高级)
 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方块99个雷


 Mine(int _grade);
 ~Mine();


 void newMine(); //产生新的雷
 void calcNum(); //计算雷的数量
 void left(int x, int y); //左击
 void right(int x, int y); //右击
};
```

```
#endif // MINE_H
```

main.cpp

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
 QApplication a(argc, argv);
 MainWindow w;
 w.show();
 return a.exec();
}
```

mainwindow.cpp

```cpp
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
 : QMainWindow(parent)
 , ui(new Ui::MainWindow)
{
 ui->setupUi(this);
 uigrade = 1; //初始化等级为1
 newGame(1); //初始化游戏，生成游戏
}

MainWindow::~MainWindow()
{
 delete ui;
}



//显示游戏时间（计时）

void MainWindow::showtime(){

 // ui->textEdit_2->setText("str");

 seconds++;

 QString str;

 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html

 str = QString("%01").arg(seconds, 3, 10, QLatin1Char('0'));

 ui->lcdNumber->display(str);

}



//开始计时

void MainWindow::startTime(){

 Timer1 = startTimer(1000);

}
```

```
//清空本来生成的东西

void MainWindow::empty(){

 //killTimer(Timer1); //先把计时停了

 //qDebug()<<"12121";

 //在重新初始化之前，先删除之前的new出来的东西，否则会出现虚幻的雷之类的

 //把new出来的按钮delete掉

 for(int j = 0 ; j < mine->y_N ; j ++ ){

 for(int i = 0 ; i < mine->x_N ; i ++ ){

 delete MainButton[j][i];

 }

 }

 //把雷delete掉

 delete mine;

}
```

```
//点击笑脸按钮
void MainWindow::on_pushButton_clicked()
{
 //qDebug()<<"尝试重新开始游戏";
 if( isStart != 0 ){ //设置必须失败才能重新开始
 empty(); //清空本来的东西
 newGame(uigrade); //生成新的游戏
 }
}


//双击的展示（用递归实现最佳）
void MainWindow::chordingShow(int x,int y){
 if( x < 0 || y < 0 || x >= mine->x_N || y >= mine->y_N ) return;
 if( mine->map[y][x] == 0 && mine->w_map[y][x] == 0 ){ //只有0且还没打开才能双击
 leftShow(x,y);
 mine->left(x,y);

 //对四周（注意加上还没打开的限制，否则会报错）

 chordingShow(x+1,y);

 chordingShow(x,y+1);

 chordingShow(x,y-1);

 chordingShow(x-1,y);



 }else if( mine->map[y][x] != -1 ){ //如果是其他，就只能左键点击

 leftShow(x,y);

 mine->left(x,y);

 }

 return;

}



//扫雷失败会把所有的雷位置展示出来

void MainWindow::failShow(int ii,int jj) //参数为点击的位置（标
```

```
红）

{

 isStart = -1; //表示为游戏失败

 ui->pushButton->setText("☹");

 killTimer(Timer1);

 //if (myTimer->isActive()) myTimer->stop();

 for(int j = 0 ; j < mine->y_N ; j ++ ){

 for(int i = 0 ; i < mine->x_N ; i ++ ){

 //失败还得把按钮都变成不可点击

 if(mine->w_map[j][i] == 0 ){ //加上判断，避免失败后一些按钮会恢复

 MainButton[j][i]->setEnabled(false);

 }



 if( mine->map[j][i] == -1 ){

 MainButton[j][i]->setStyleSheet("background-
color:#c2d0ea;font-size:10px;font-weight:bold;");

 if( ii == i && jj == j ){ //点击的位置标红
 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#ff0000;font-size:10px;font-weight:bold;");

 }

 mine->w_map[j][i] = -1;

 this->MainButton[j][i]->setDown(true);

 this->MainButton[j][i]->setText("💣");

 }

 }

 }





}



//初始化游戏

void MainWindow::newGame(int difficult){

 // 添加图标

```

```
QIcon icon(":/new/prefix1/img/lei.ico");

this->setWindowIcon(icon);

seconds=0;

int move_x,move_y;

if( difficult == 1 ){

this->resize(220,260);

move_x = 20,move_y = 65;

ui->lcdNumber_2->move(20,10);

ui->pushButton->move(90,10);

ui->lcdNumber->move(130,10);



}else if( difficult == 2 ){

this->resize(360,400);

move_x = 20,move_y = 65;

ui->lcdNumber_2->move(20,10);

ui->pushButton->move(160,10);
ui->lcdNumber->move(270,10);



}else if( difficult == 3 ){

this->resize(640,400);

move_x = 20,move_y = 65;

ui->lcdNumber_2->move(20,10);

ui->pushButton->move(300,10);

ui->lcdNumber->move(550,10);




}




//qDebug()<<"重新游戏";

isStart = 0; //尚未点击开始按钮



//设置背景颜色(#dadfec;)#91a0b3;

this->setStyleSheet("background-color:#cfd8df");
```

```
 ui->lcdNumber->setStyleSheet("background-color:#000000;color:#ff0000;");

 ui->lcdNumber_2->setStyleSheet("background-color:#000000;color:#ff0000;");



 mine = new Mine(difficult); //生成地雷

 //Qt按钮设置：https://baijiahao.baidu.com/s?id=1691291899564946968&wfr=spider&for=pc



 int size = 20;



 //注意是[y][x]对应[j][i]



 for(int j = 0 ; j < mine->y_N ; j++ ){

 for(int i = 0 ; i < mine->x_N ; i++ ){

 MainButton[j][i] = new QPushButton(" ",this);
 MainButton[j][i]->setCheckable(true);

 MainButton[j][i]->setAutoExclusive(true);



 //设置按钮颜色border:2px solid #808080 rgb(192,192,192);#77a7f0 #3547b2;

 //阴影感: https://blog.csdn.net/guorong520/article/details/107126554/

 MainButton[j][i]->setStyleSheet("background-color:#6fabee;border-bottom:4px solid #619fec;border-right:4px solid #619fec;");

 MainButton[j][i]->resize(size,size);

 //MainButton[j][i]->setEnabled(false);

 MainButton[j][i]->move(move_x+size*(i),move_y+size*(j)); //将按钮b1移动到点（50，50）的位置

 // MainButton[j][i]->setParent("1",this);



 MainButton[j][i]->show();

 //给connect传参数的方法: https://blog.csdn.net/qq_38188725/article/details/80686793
```

```
//解决鼠标左键无法响应鼠标事件，和connect无法响应
connect(MainButton[j][i], &QPushButton::clicked,this,
[=]{ButtonClick(i,j);
});
}
}


ui->pushButton->setText("☺");
ui->pushButton->resize(30,30);


//设置lcd位数
ui->lcdNumber->setDigitCount(4);
ui->lcdNumber_2->setDigitCount(3);
QString tmp = QString("%1").arg(mine->lei, 3, 10, QLatin1Char('0')); //整型转字符串); //整型转字符串;
ui->lcdNumber_2->display(tmp);
```

```
ui->lcdNumber->display("000");

/*
//显示游戏时间
myTimer = new QTimer(this);
connect(myTimer,&QTimer::timeout, this, [=]{
showtime();});
*/


//解决鼠标单击还是双击的
ClickedState = 0; //初始化不点击
/*
timer = new QTimer(this);
connect(timer, &QTimer::timeout, this,[ = ]{
 timerSingleClick();

 });*/

}



//鼠标单击事件

void MainWindow::mousePressEvent(QMouseEvent *event){

 //Timer2 = startTimer(180);

 //timer->start(300);



 //鼠标事件参考链接：https://www.cnblogs.com/aiguona/p/9999507.html

 //qDebug("鼠标点击");

 // 如果是鼠标左键按下

 // 如果是鼠标右键按下

 if(event->button() == Qt::RightButton)

 {

 //qDebug("鼠标右键点击");
 killTimer(Timer2);

 //timer->stop();

 for( int j = 0 ; j < mine->y_N ; j ++ ){

 for( int i = 0 ; i < mine->x_N ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

 //qDebug("%d,%d",i,j);

 showButton(i,j,2);

 }



 }

 }



 }

}
```

```
//鼠标双击事件

void MainWindow::mouseDoubleClickEvent(QMouseEvent *event){

 // 如果是鼠标左键按下（双击）

 if(event->button() == Qt::LeftButton){

 //qDebug("鼠标左键双击");

 for( int j = 0 ; j < mine->y_N ; j ++ ){

 for( int i = 0 ; i < mine->x_N ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

 //qDebug("%d,%d",i,j);

 chordingShow(i,j);

 showButton(i,j,3);

 }



 }

 }

 }
```

```
}



//左键点击的按钮显示

void MainWindow::leftShow(int x,int y){ //左键单击的显示

 int i = x,j = y;

 if( mine->map[j][i] == 0 ){ //rgb(192,192,192)

 MainButton[j][i]->setStyleSheet("background-color:#c2d0ea;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText("");

 }else if( mine->map[j][i] == 1 ){

 //设置按钮颜色border:1px solid #999999;background-color:#c0c0c0

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 2 ){

 MainButton[j][i]->setStyleSheet("color:#008000;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");
 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 3 ){

 MainButton[j][i]->setStyleSheet("color:#fe0202;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else{

 MainButton[j][i]->setStyleSheet("color:#131386;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }

 this->MainButton[j][i]->setDown(true);



}



//按钮点击触发的事件
```

```
void MainWindow::showButton(int i, int j, int model){
 if( i < 0 && j < 0 ) return;


 if( isStart == 0 ){ //如果还没未开始，则现在开始计时
 startTime();
 }
 this->MainButton[j][i]->setEnabled(false); //已点击过的按钮无法再点击
 //注意这里的i和j对应x,y
 //qDebug("%d,%d",i,j);
 if( model == 1 ){ //左键点击
 //w_map为0说明没打开，没打开的才能点击
 if( mine->w_map[j][i] == 0 ){
 //qDebug("1111");


 //this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));
 if( mine->map[j][i] == -1 ){ //踩到雷
 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");
 //此处应该插入游戏失败的代码
 //this->MainButton[j][i]->setText("💣");
 failShow(i,j); //失败+显示所有的雷
 }else{
 if( mine->map[j][i] == 0 ){ //如果点击0相当于双击
 chordingShow(i,j);
 }else{
 leftShow(i,j);
 }
 }
 if(!this->MainButton[j][i]->isDown())
 this->MainButton[j][i]->setDown(true);
 mine->left(i,j);
 //设置按钮为按下状态:
```

```
https://blog.csdn.net/qiangzi4646/article/details/80135120



 }else{

 if(!this->MainButton[j][i]->isDown())

 this->MainButton[j][i]->setDown(true);

 }



 // this->MainButton[j][i]->setText("");

 }else if( model == 2 ){ //右键点击（标记）

 //w_map为0说明没打开，没打开的才能点击

 if( mine->w_map[j][i] == 0 ){

 //qDebug("2222");

 if( mine->lei > 0 ){

 mine->right(i,j);

 this->MainButton[j][i]->setText("⚐");

 }
 //前位补0：https://www.cnblogs.com/zzzsj/p/14566157.html

 QString L = QString("%1").arg(mine->lei, 3, 10,
QLatin1Char('0')); //整型转字符串

 ui->lcdNumber_2->display(L);

 }else{

 if(!this->MainButton[j][i]->isDown())

 this->MainButton[j][i]->setDown(true);

 }

 }else if( model == 3 ){

 //qDebug("3333");

 //mine->chording(i,j); //双击

 this->MainButton[j][i]->setText(" ");

 }

}



//定时器事件

void MainWindow::timerEvent(QTimerEvent *event){
 //QTimerEvent:https://www.cnblogs.com/Yu-nan/p/12091264.html

 //用两个QTimer不对劲

 if(event->timerId() == Timer1) {

 //do Timer1_Fuc

 showtime();

 }

 if(event->timerId() == Timer2) {

 //do Timer2_Fuc

 timerSingleClick();

 }

}



//单击双击操作

void MainWindow::ButtonClick(int i,int j){

 ClickedState ++;

 retainx = i ,retainy = j;
 if(ClickedState == 1)
 Timer2 = startTimer(180); //180ms为自己设置
 if(ClickedState == 2)
 {
 ClickedState = 0; //初始化
 //qDebug()<<Timer2<<"222";
 killTimer(Timer2);
 //按钮双击响应代码放这里
 //QMessageBox::warning(this,tr("警告"),tr("我是双击！！！"),tr("确定"));
 showButton(i,j,3);
 }


}


//单击双击：https://blog.csdn.net/wdmzslh/article/details/53301495
```

```
void MainWindow::timerSingleClick()
{
 ClickedState = 0; //初始化
 //qDebug()<<Timer2<<"111";
 killTimer(Timer2);
 //按钮单击响应代码放这里
 //QMessageBox::warning(this,tr("警告"),tr("单击"),tr("确定"));
 showButton(retainx,retainy,1);
 retainx = -1;
 retainy = -1;
}


//初级被点击
void MainWindow::on_action_2_triggered()
{
 empty();
 uigrade = 1;
 newGame(1);
}

//中级被点击
void MainWindow::on_action_4_triggered()
{
 empty();
 uigrade = 2;
 newGame(2);
}

//高级被点击
void MainWindow::on_action_5_triggered()
{
 empty();
 uigrade = 3;

 newGame(3);

}



//游戏说明被点击

void MainWindow::on_action_6_triggered()

{

 QMessageBox::about(NULL, "游戏说明", "本游戏制作 ——by：orall\n游戏的基本操作包括左键单击、右键单击、双击三种。\n其中左键用于打开安全的格子，推进游戏进度；\n右键用于标记地雷，以辅助判断，或为接下来的双击做准备；\n双击在一个数字周围的地雷标记完时，相当于对数字周围未打开的方块均进行一次左键单击操作。");

}
```

Mine.cpp

```


#include "Mine.h"

#include "ui_mainwindow.h"

//注意是map[y][x]
```

```
//产生新的雷（map中-1表示雷）
void Mine::newMine() {

 //初始化
 for(int j = 0 ; j < 30 ; j++ ){
 for(int i = 0 ; i < 30 ; i++ ){
 map[j][i] = 0;
 w_map[j][i] = 0; //注意这里w_map得初始化，否则会出错
 }
 }
 lei = lei_all;
 success = 0; //找到的雷初始化为0

 int tmp = 0; //随机生成对应雷
 srand((unsigned)time(NULL));
 while(1){

 int x = rand() % x_N; //产生0到雷数的数字

 int y = rand() % y_N;

 if (map[y][x] != -1) { //如果该位置没有雷

 map[y][x] = -1; //埋下地雷

 tmp++;

 }

 if( tmp >= lei_all ) break; //埋下对应雷数就停止

 }

}



//根据雷填充map地图

void Mine::calcNum() { //计算雷的数量(用map[j][i]更佳，不容易混)

 //根据展开方格的数字去推断其相邻九宫格内未展开方格下面是否是地雷



 //填入数字
```

```
for(int j = 0 ; j < y_N ; j ++ ){

for(int i = 0 ; i < x_N ; i++ ){

if( map[j][i] != -1 ){ //如果不是雷就计算周围的雷

int tmp = 0;

//周围一圈

if (i - 1 >= 0 && j + 1 < y_N && map[j + 1][i - 1] == -1) tmp++;

if (j + 1 < y_N && map[j + 1][i] == -1) tmp++;

if (j + 1 < y_N && i + 1 < x_N && map[j + 1][i + 1] == -1) tmp++;



if (i - 1 >= 0 && map[j][i - 1] == -1) tmp++;

if (i + 1 < x_N && map[j][i + 1] == -1) tmp++;



if (i - 1 >= 0 && j - 1 >= 0 && map[j - 1][i - 1] == -1) tmp++;

if (j - 1 >= 0 && map[j - 1][i] == -1) tmp++;
```

```
 if (i + 1 < x_N && j - 1 >= 0 && map[j - 1][i + 1] == -1)
tmp++;

 map[j][i] = tmp;

 }

 }

 }

}



void Mine::left(int x, int y) {

 //☺☹

 //左键用于打开安全的格子，推进游戏进度；

 if (map[y][x] == -1) {

 //信息框（https://www.cnblogs.com/zhoug2020/p/10094855.html，注
意加上头文件）

 QMessageBox::information(NULL, "游戏结束", "很遗憾，你已经踩到雷
了，游戏结束！");

 }

 else {
```

```
 if (map[y][x] == 0) {

 w_map[y][x] = -3; //-3在表地图表示为空

 }

 else {

 w_map[y][x] = map[y][x];

 }

 }

}



void Mine::right(int x, int y) {

 //后续补充再次右键可取消

 if( lei > 0 ){ //如果标记够用的话

 lei--; //标记雷的数量减1

 //右键用于标记地雷，以辅助判断，或为接下来的双击做准备；(-2表示标记)

 if( w_map[y][x] == -2 ){

 w_map[y][x] = 0;
 }else{

 w_map[y][x] = -2;

 }

 if( map[y][x] == -1 ){ //如果找到的是雷

 success++;

 }

 }else{

 if( success == lei_all ){ //成功找到所有的雷

 //此处应该贴胜利代码

 QMessageBox::information(NULL, "游戏结束", "恭喜你，游戏胜
利！");

 }



 }

}
```

```
Mine::Mine(int _grade):grade(_grade){

 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方块99个雷

 if( grade == 1 ){

 y_N = 9;

 x_N = 9;

 lei = lei_all = 10; //旗的个数和雷的个数

 }else if( grade == 2 ){

 y_N = 16;

 x_N = 16;

 lei = lei_all = 40; //旗的个数和雷的个数



 }else if( grade == 3 ){

 y_N = 16;

 x_N = 30;

 lei = lei_all = 99; //旗的个数和雷的个数

 }
```

```
newMine();
calcNum();

//显示地图
for(int j = 0 ; j < y_N ; j ++ ){
for(int i = 0 ; i < x_N ; i++ ){
std::cout<<map[j][i]<<" ";
}
std::cout<<std::endl;
}

//显示地图
for(int j = 0 ; j < y_N ; j ++ ){
for(int i = 0 ; i < x_N ; i++ ){
std::cout<<w_map[j][i]<<" ";
```

```
 }
 std::cout<<std::endl;
 }

}

Mine::~Mine(){}
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
 <property name="windowModality">
 <enum>Qt::WindowModal</enum>
 </property>
```

```
<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>259</width>

<height>352</height>

</rect>

</property>

<property name="font">

<font>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="contextMenuPolicy">

<enum>Qt::NoContextMenu</enum>
```

```
</property>

<property name="windowTitle">

<string>扫雷 ——by：orall</string>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="pushButton">

<property name="geometry">

<rect>

<x>110</x>

<y>10</y>

<width>31</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>
```

```
<family>Microsoft YaHei UI Light</family>
<weight>75</weight>
<bold>true</bold>
</font>
</property>
<property name="text">
<string/>
</property>
</widget>
<widget class="QLCDNumber" name="lcdNumber">
<property name="geometry">
<rect>
<x>150</x>
<y>10</y>
<width>64</width>
<height>31</height>
```

```
</rect>

</property>

<property name="font">

<font>

<family>Arial</family>

<pointsize>9</pointsize>

<weight>50</weight>

<bold>false</bold>

</font>

</property>

<property name="frameShape">

<enum>QFrame::Box</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Raised</enum>

</property>
```

```xml
<property name="lineWidth">

<number>1</number>

</property>

<property name="midLineWidth">

<number>0</number>

</property>

</widget>

<widget class="QLCDNumber" name="lcdNumber_2">

<property name="geometry">

<rect>

<x>30</x>

<y>10</y>

<width>61</width>

<height>31</height>

</rect>

</property>
```

```
<property name="font">
<font>
<weight>50</weight>
<bold>false</bold>
</font>
</property>
<property name="frameShape">
<enum>QFrame::Box</enum>
</property>
<property name="frameShadow">
<enum>QFrame::Raised</enum>
</property>
<property name="lineWidth">
<number>1</number>
</property>
<property name="digitCount">
```

```
<number>5</number>

</property>

<property name="intValue" stdset="0">

<number>10</number>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>259</width>

<height>25</height>

</rect>

</property>
```

```
<widget class="QMenu" name="menu">

<property name="title">

<string>游戏</string>

</property>

<addaction name="separator"/>

<addaction name="action_2"/>

<addaction name="action_4"/>

<addaction name="action_5"/>

<addaction name="action_6"/>

<addaction name="separator"/>

</widget>

<addaction name="menu"/>

</widget>

<widget class="QStatusBar" name="statusbar"/>

<action name="action_2">

<property name="text">
```

```
<string>初级</string>

</property>

</action>

<action name="action_4">

<property name="text">

<string>中级</string>

</property>

</action>

<action name="action_5">

<property name="text">

<string>高级</string>

</property>

</action>

<action name="action_6">

<property name="text">

<string>游戏说明</string>
```

```
 </property>

 </action>

 </widget>

 <resources/>

 <connections/>

</ui>
```

## 7.扫雷4.0版本（2021.10.15）

该版本改进了：
1.第二次开始时，点击按钮没反应【已解决：w_map在重开也需要初始化】
2.点击的时候按钮会恢复【解决一部分：大部分情况不会恢复，小部分情况会恢复】
mainwindow.h

```


#ifndef MAINWINDOW_H

#define MAINWINDOW_H
```

```cpp
#include <QMainWindow>
#include <QPainter>
#include <QPushButton>
#include <QLabel>
#include <QMouseEvent>
#include <QTimer> // 定时器对象
#include "Mine.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
 Q_OBJECT
```

```
public:
 QPushButton *MainButton[30][30];
 Mine *mine; //地雷
 int retainx,retainy; //鼠标点击的位置
 int isStart; //是否开始游戏
 int uigrade; //难度
 int seconds; //用时

 //定义需要用到的定时器（第一个用来计时，第二个用来鼠标点击）
 int Timer1; //用来计算时间
 int Timer2; //用来判断鼠标点击是单击还是双击
 int ClickedState; //点击状态，常态不点击为0，单击对应1，双击对应2

 MainWindow(QWidget *parent = nullptr);
 ~MainWindow();
 void chordingShow(int x,int y); //双击的展示

 void leftShow(int x,int y); //左键单击的显示

 void showtime(); //更新时钟

 void showButton(int i,int j,int model);

 void startTime(); //开始计时

 void empty(); //清空之前创建的东西

 void mousePressEvent(QMouseEvent *event); //鼠标点击事件

 void mouseDoubleClickEvent(QMouseEvent *event); //鼠标双击事件

 void newGame(int difficult); //初始化游戏

 void failShow(int ii,int jj); //失败的时候显示所有的雷

 void ButtonClick(int i,int j); //按钮点击

 void timerSingleClick(); //定时器事件，处理单击任务

 int* reverseij(int i,int j); //保存当前坐标

private slots:

 void timerEvent(QTimerEvent *event);

 void on_pushButton_clicked();
 void on_action_2_triggered();

 void on_action_4_triggered();

 void on_action_5_triggered();



private:

 Ui::MainWindow *ui;


};



#endif // MAINWINDOW_H
```

Mine.h

```


#ifndef MINE_H

#define MINE_H



#include<iostream>
```

```
#include<time.h>

#include<stdlib.h>

#include <QMessageBox>



class Mine{ //扫雷的类

public:

 int map[30][30]; //初级（初级为9*9个方块10个雷） //雷表示-1

 int w_map[30][30]; //显示出来的地图

 int lei; //右键旗的次数

 int lei_all; //雷的个数

 int success; //成功全部雷即为胜利

 int y_N,x_N; //map[y_N][x_N]

 int grade; //等级(初级，中级，高级)

 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方块99个雷



 Mine(int _grade);
 ~Mine();

 void newMine(); //产生新的雷
 void calcNum(); //计算雷的数量
 void left(int x, int y); //左击
 void right(int x, int y); //右击
// void chording(int x, int y); //双击
};


#endif // MINE_H
```

main.cpp

```
#include "mainwindow.h"

#include <QApplication>
```

```
int main(int argc, char *argv[])
{
 QApplication a(argc, argv);
 MainWindow w;
 w.show();
 return a.exec();
}
```

mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
 : QMainWindow(parent)
 , ui(new Ui::MainWindow)
```

```
{
 ui->setupUi(this);
 uigrade = 1; //初始化等级为1
 newGame(1); //初始化游戏
}

MainWindow::~MainWindow()
{
 delete ui;
}

//显示游戏时间（计时）
void MainWindow::showtime(){
 // ui->textEdit_2->setText("str");
 seconds++;
 QString str;
 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html

 str = QString("%01").arg(seconds, 3, 10, QLatin1Char('0'));

 ui->lcdNumber->display(str);

}



//开始计时

void MainWindow::startTime(){

 Timer1 = startTimer(1000);

 /*

 myTimer->start(1000);

 //启动定时器

 //时间间隔为100ms

 //每隔100ms, 定时器myTimer自动触发timeout()

 //如果定时器没有激活，才启动,防止多次点击start按钮出现错误

 if (myTimer->isActive() == false)

 {
```

```
 isStart = 1; //标记为开始游戏

 }*/

}



//清空本来生成的东西

void MainWindow::empty(){

 killTimer(Timer1); //先把计时停了



 //qDebug()<<"12121";

 //在重新初始化之前，先删除之前的new出来的东西，否则会出现虚幻的雷之类的

 //把new出来的按钮delete掉

 for(int j = 0 ; j < mine->y_N ; j ++ ){

 for(int i = 0 ; i < mine->x_N ; i ++ ){

 delete MainButton[j][i];

 }
 }

 //把雷delete掉

 delete mine;

}



//点击笑脸按钮

void MainWindow::on_pushButton_clicked()

{

 //qDebug()<<"尝试重新开始游戏";

 if( isStart != 0 ){ //设置必须失败才能重新开始

 empty(); //清空本来的东西

 newGame(uigrade); //生成新的

 }

}



//双击的展示（用递归实现最佳）

```

```
void MainWindow::chordingShow(int x,int y){

 if( x < 0 || y < 0 || x >= mine->x_N || y >= mine->y_N ) return;

 if( mine->map[y][x] == 0 && mine->w_map[y][x] == 0 ){ //只有0且还没打开才能双击

 leftShow(x,y);

 mine->left(x,y);

 //对四周（注意加上还没打开的限制，否则会报错）

 chordingShow(x+1,y);

 chordingShow(x,y+1);

 chordingShow(x,y-1);

 chordingShow(x-1,y);



 }else if( mine->map[y][x] != -1 ){ //如果是其他，就只能左键点击

 leftShow(x,y);

 mine->left(x,y);

 }
 return;


}


//扫雷失败会把所有的雷位置展示出来

void MainWindow::failShow(int ii,int jj) //参数为点击的位置（标红）

{


 isStart = -1; //表示为游戏失败

 ui->pushButton->setText("🙁");

 killTimer(Timer1);

 //if (myTimer->isActive()) myTimer->stop();

 for(int j = 0 ; j < mine->y_N ; j ++ ){

 for(int i = 0 ; i < mine->x_N ; i ++ ){

 //失败还得把按钮都变成不可点击

 if(mine->w_map[j][i] == 0 ){ //加上判断，避免失败后一些按钮会恢复
 MainButton[j][i]->setEnabled(false);
 }

 if( mine->map[j][i] == -1 ){
 MainButton[j][i]->setStyleSheet("background-color:#c2d0ea;font-size:10px;font-weight:bold;");
 if( ii == i && jj == j ){ //点击的位置标红
 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#ff0000;font-size:10px;font-weight:bold;");
 }
 mine->w_map[j][i] = -1;
 this->MainButton[j][i]->setDown(true);
 this->MainButton[j][i]->setText("💣");
 }
 }
 }
```

```
}


//初始化游戏
void MainWindow::newGame(int difficult){
 seconds=0;
 int move_x,move_y;
 if( difficult == 1 ){
 this->resize(220,260);
 move_x = 20,move_y = 65;
 ui->lcdNumber_2->move(20,10);
 ui->pushButton->move(90,10);
 ui->lcdNumber->move(130,10);

 }else if( difficult == 2 ){
```

```
this->resize(360,400);
move_x = 20,move_y = 65;
ui->lcdNumber_2->move(20,10);
ui->pushButton->move(160,10);
ui->lcdNumber->move(270,10);


}else if( difficult == 3 ){
this->resize(640,400);
move_x = 20,move_y = 65;
ui->lcdNumber_2->move(20,10);
ui->pushButton->move(300,10);
ui->lcdNumber->move(550,10);


}


//qDebug()<<"重新游戏";
```

```
 isStart = 0; //尚未点击开始按钮



 //设置背景颜色(#dadfec;)#91a0b3;

 this->setStyleSheet("background-color:#cfd8df");



 ui->lcdNumber->setStyleSheet("background-color:#000000;color:#ff0000;");

 ui->lcdNumber_2->setStyleSheet("background-color:#000000;color:#ff0000;");



 mine = new Mine(difficult); //生成地雷

 //Qt按钮设置：https://baijiahao.baidu.com/s?id=1691291899564946968&wfr=spider&for=pc



 int size = 20;



 //注意是[y][x]对应[j][i]
 for(int j = 0 ; j < mine->y_N ; j++ ){

 for(int i = 0 ; i < mine->x_N ; i++ ){

 MainButton[j][i] = new QPushButton(" ",this);



 MainButton[j][i]->setCheckable(true);

 MainButton[j][i]->setAutoExclusive(true);



 //设置按钮颜色border:2px solid #808080 rgb(192,192,192);#77a7f0 #3547b2;

 //阴影感：https://blog.csdn.net/guorong520/article/details/107126554/

 MainButton[j][i]->setStyleSheet("background-color:#6fabee;border-bottom:4px solid #619fec;border-right:4px solid #619fec;");

 MainButton[j][i]->resize(size,size);

 //MainButton[j][i]->setEnabled(false);

 MainButton[j][i]->move(move_x+size*(i),move_y+size*(j)); //将按钮b1移动到点（50，50）的位置
```

```
// MainButton[j][i]->setParent("1",this);




MainButton[j][i]->show();

//给connect传参数的方法:https://blog.csdn.net/qq_38188725/article/details/80686793

//解决鼠标左键无法响应鼠标事件，和connect无法响应

connect(MainButton[j][i], &QPushButton::clicked,this,

[=]{ButtonClick(i,j);

});




}

}
```

```
 ui->pushButton->setText("🙂");

 ui->pushButton->resize(30,30);



 //设置lcd位数

 ui->lcdNumber->setDigitCount(4);

 ui->lcdNumber_2->setDigitCount(3);

 QString tmp = QString("%1").arg(mine->lei, 3, 10, QLatin1Char('0')); //整型转字符串); //整型转字符串;

 ui->lcdNumber_2->display(tmp);



 ui->lcdNumber->display("000");



 /*

 //显示游戏时间

 myTimer = new QTimer(this);

 connect(myTimer,&QTimer::timeout, this, [=]{

 showtime();});
 */



 //解决鼠标单击还是双击的
 ClickedState = 0; //初始化不点击
 /*
 timer = new QTimer(this);
 connect(timer, &QTimer::timeout, this,[ = ]{
 timerSingleClick();
 });*/
}


void MainWindow::mousePressEvent(QMouseEvent *event){


 Timer2 = startTimer(180);
 //timer->start(300);
 //鼠标事件参考链接:
https://www.cnblogs.com/aiguona/p/9999507.html

 qDebug("鼠标点击");

 // 如果是鼠标左键按下

 if(event->button() == Qt::LeftButton)

 {

 /*

 qDebug("鼠标左键点击");



 for( int j = 0 ; j < mine->y_N ; j ++ ){

 for( int i = 0 ; i < mine->x_N ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this-
>mapFromGlobal(QCursor::pos()))){

 qDebug("%d,%d",i,j);

 showButton(i,j,1);

 }
```

```
}
}*/
}
// 如果是鼠标右键按下
else if(event->button() == Qt::RightButton)
{
qDebug("鼠标右键点击");
killTimer(Timer2);
//timer->stop();
for( int j = 0 ; j < mine->y_N ; j ++ ){
for( int i = 0 ; i < mine->x_N ; i ++ ){
if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){
qDebug("%d,%d",i,j);
showButton(i,j,2);
```

```
 }



 }

 }



 }

}

//鼠标双击事件

void MainWindow::mouseDoubleClickEvent(QMouseEvent *event){

 // 如果是鼠标左键按下（双击）

 if(event->button() == Qt::LeftButton){

 //qDebug("鼠标左键双击");

 for( int j = 0 ; j < mine->y_N ; j ++ ){

 for( int i = 0 ; i < mine->x_N ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this-
>mapFromGlobal(QCursor::pos()))){

 //qDebug("%d,%d",i,j);
 chordingShow(i,j);

 showButton(i,j,3);

 }



 }

 }

 }

}



void MainWindow::leftShow(int x,int y){ //左键单击的显示

 int i = x,j = y;

 if( mine->map[j][i] == 0 ){ //rgb(192,192,192)

 MainButton[j][i]->setStyleSheet("background-
color:#c2d0ea;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText("");

 }else if( mine->map[j][i] == 1 ){
```

```
 //设置按钮颜色border:1px solid #999999;background-color:#c0c0c0

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 2 ){

 MainButton[j][i]->setStyleSheet("color:#008000;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 3 ){

 MainButton[j][i]->setStyleSheet("color:#fe0202;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else{

 MainButton[j][i]->setStyleSheet("color:#131386;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));
 }

 this->MainButton[j][i]->setDown(true);



}

//按钮点击触发的事件

void MainWindow::showButton(int i, int j, int model){

 // if( isStart == 1 ){



 if( isStart == 0 ){ //如果还没未开始，则现在开始计时

 startTime();

 }



 this->MainButton[j][i]->setEnabled(false); //已点击过的按钮无法再点击

 //注意这里的i和j对应x,y

 qDebug("%d,%d",i,j);

 if( model == 1 ){ //左键点击
 //w_map为0说明没打开，没打开的才能点击

 if( mine->w_map[j][i] == 0 ){

 qDebug("1111");



 //this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 if( mine->map[j][i] == -1 ){ //踩到雷

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

 //此处应该插入游戏失败的代码

 //this->MainButton[j][i]->setText("💣");

 failShow(i,j); //失败+显示所有的雷

 }else{

 if( mine->map[j][i] == 0 ){ //如果点击0相当于双击

 chordingShow(i,j);

 }else{

 leftShow(i,j);
 }

 }

 if(!this->MainButton[j][i]->isDown())

 this->MainButton[j][i]->setDown(true);

 mine->left(i,j);



 //设置按钮为按下状态：
https://blog.csdn.net/qiangzi4646/article/details/80135120



 }else{

 if(!this->MainButton[j][i]->isDown())

 this->MainButton[j][i]->setDown(true);

 }



 // this->MainButton[j][i]->setText("");

 }else if( model == 2 ){ //右键点击（标记）
```

```
//w_map为0说明没打开，没打开的才能点击
if( mine->w_map[j][i] == 0 ){
//qDebug("2222");
if( mine->lei > 0 ){
mine->right(i,j);
this->MainButton[j][i]->setText("🚩");
}
//前位补0：https://www.cnblogs.com/zzzsj/p/14566157.html
QString L = QString("%1").arg(mine->lei, 3, 10, QLatin1Char('0')); //整型转字符串
ui->lcdNumber_2->display(L);
}else{
if(!this->MainButton[j][i]->isDown())
this->MainButton[j][i]->setDown(true);
}
//应该是设置了不可点击，所以没法撤销
/*
```

```
 else if( mine->w_map[j][i] == -2 ){ //再点一次恢复

 this->MainButton[j][i]->setEnabled(true); //按钮再点击

 if( mine->lei <= 10 ){

 mine->right(i,j);

 this->MainButton[j][i]->setText(" ");

 }

 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html

 QString L = QString("%1").arg(mine->lei, 3, 10, QLatin1Char('0')); //整型转字符串

 ui->lcdNumber_2->display(L);



 }*/

 // ui->textEdit->setText(mine->);

 }else if( model == 3 ){

 qDebug("3333");

 //mine->chording(i,j); //双击
 this->MainButton[j][i]->setText(" ");

 }

 // }



}



void MainWindow::timerEvent(QTimerEvent *event){

 //QTimerEvent:https://www.cnblogs.com/Yu-nan/p/12091264.html

 //用两个QTimer不对劲



 if(event->timerId() == Timer1) {

 //do Timer1_Fuc

 showtime();

 }

 if(event->timerId() == Timer2) {

 //do Timer2_Fuc
 timerSingleClick();

 }



}




void MainWindow::ButtonClick(int i,int j){

 ClickedState ++;

 retainx = i ,retainy = j;

 if(ClickedState == 1)

 Timer2 = startTimer(180);

 //timer->start(180); //430ms为自己设置


 if(ClickedState == 2)

 {
```

```
 ClickedState = 0; //初始化

 killTimer(Timer2);

 //timer->stop(); //停止定时器继续计时

 //按钮双击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("我是双
击！！！"),tr("确定"));

 showButton(i,j,3);

 }



}



//https://blog.csdn.net/wdmzslh/article/details/53301495

void MainWindow::timerSingleClick()

{

 ClickedState = 0; //初始化

 //timer->stop(); //定时器停止计时

 killTimer(Timer2);
 //按钮单击响应代码放这里
 //QMessageBox::warning(this,tr("警告"),tr("单击"),tr("确定"));
 showButton(retainx,retainy,1);
}


//初级被点击
void MainWindow::on_action_2_triggered()
{
 empty();
 uigrade = 1;
 newGame(1);
}

//中级被点击
```

```
void MainWindow::on_action_4_triggered()
{
 empty();
 uigrade = 2;
 newGame(2);
}



//高级被点击
void MainWindow::on_action_5_triggered()
{
 empty();
 uigrade = 3;
 newGame(3);
}
```

Mine.cpp

```
//注意是map[y][x]
#include "Mine.h"
#include "ui_mainwindow.h"

//产生新的雷（map中-1表示雷）
void Mine::newMine() {
 //lei = 10;
 success = 0;
 int tmp = 0; //随机生成对应雷
 srand((unsigned)time(NULL));
 while(1){
 int x = rand() % x_N; //产生0到8的数字
 int y = rand() % y_N;
 if (map[y][x] != -1) {
```

```
 map[y][x] = -1; //埋下地雷

 tmp++;

 }

 if( tmp >= lei_all ) break;

 }





}



//根据雷补充map地图

void Mine::calcNum() { //计算雷的数量(实际用map[j][i]更佳，不容易
混)

 //根据展开方格的数字去推断其相邻九宫格内未展开方格下面是否是地雷



 //填入数字

 for(int j = 0 ; j < y_N ; j ++ ){

 for(int i = 0 ; i < x_N ; i++ ){
```

```
 if( map[j][i] != -1 ){ //如果不是雷就计算周围的雷

 int tmp = 0;

 //周围一圈

 if (i - 1 >= 0 && j + 1 < y_N && map[j + 1][i - 1] == -1)
tmp++;

 if (j + 1 < y_N && map[j + 1][i] == -1) tmp++;

 if (j + 1 < y_N && i + 1 < x_N && map[j + 1][i + 1] == -1)
tmp++;



 if (i - 1 >= 0 && map[j][i - 1] == -1) tmp++;

 if (i + 1 < x_N && map[j][i + 1] == -1) tmp++;



 if (i - 1 >= 0 && j - 1 >= 0 && map[j - 1][i - 1] == -1)
tmp++;

 if (j - 1 >= 0 && map[j - 1][i] == -1) tmp++;

 if (i + 1 < x_N && j - 1 >= 0 && map[j - 1][i + 1] == -1)
tmp++;

 map[j][i] = tmp;
 }

 }

 }

}



void Mine::left(int x, int y) {



 //☺🙁☹

 //左键用于打开安全的格子，推进游戏进度；

 if (map[y][x] == -1) {

 //信息框（https://www.cnblogs.com/zhoug2020/p/10094855.html，注意加上头文件）



 QMessageBox::information(NULL, "游戏结束", "很遗憾，你已经踩到雷了，游戏结束！");

 //cout << "你已经踩雷，游戏失败" << endl;

 }
 else {

 if (map[y][x] == 0) {

 w_map[y][x] = -3; //-3表示为空

 }

 else {

 w_map[y][x] = map[y][x];

 }

 }



}



void Mine::right(int x, int y) {

 //后续补充再次右键可取消

 if( lei > 0 ){ //如果标记够用的话

 lei--; //标记雷的数量减1

 //右键用于标记地雷，以辅助判断，或为接下来的双击做准备；(-2表示标记)
```

```
if( w_map[y][x] == -2 ){
w_map[y][x] = 0;
}else{
w_map[y][x] = -2;
}

if( map[y][x] == -1 ){ //如果找到的是雷
success++;
}


}else{
if( success == lei_all ){ //成功找到所有的雷
//此处应该贴胜利
QMessageBox::information(NULL, "游戏结束", "恭喜你，游戏胜利！");
}
```

```
 }

}

/*

void Mine::chording(int x, int y) {

 //双击只能双击没有数字的

 // 双击在一个数字周围的地雷标记完时， 相当于对数字周围未打开的方块均进行一次左键单击操作：

 //标记的雷数一定要等于X，否则双击是打不开的。

 //把周围没标记的都打开（8个）

 if (x + 1 < 9 && w_map[y][x + 1] != -2) left(x + 1, y);

 if (x + 1 < 9 && y - 1 >= 0 && w_map[y-1][x + 1] != -2) left(x + 1, y-1);

 if (x + 1 < 9 && y + 1 < 9 && w_map[y+1][x + 1] != -2) left(x + 1, y+1);




 if ( y - 1 >= 0 && w_map[y - 1][x] != -2) left(x, y - 1);
 if ( y + 1 < 9 && w_map[y + 1][x] != -2) left(x, y + 1);



 if (x - 1 >= 0 && w_map[y][x - 1] != -2) left(x + 1, y);

 if (x - 1 >= 0 && y - 1 >= 0 && w_map[y - 1][x - 1] != -2)
left(x - 1, y - 1);

 if (x - 1 >= 0 && y + 1 < 9 && w_map[y + 1][x - 1] != -2)
left(x - 1, y + 1);

}

*/

Mine::Mine(int _grade):grade(_grade){

 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方
块99个雷

 if( grade == 1 ){

 y_N = 9;

 x_N = 9;

 lei = lei_all = 10; //旗的个数和雷的个数

 }else if( grade == 2 ){
```

```
y_N = 16;
x_N = 16;
lei = lei_all = 40; //旗的个数和雷的个数

}else if( grade == 3 ){
y_N = 16;
x_N = 30;
lei = lei_all = 99; //旗的个数和雷的个数

}

newMine();
calcNum();
for(int j = 0 ; j < y_N ; j ++ ){
for(int i = 0 ; i < x_N ; i++ ){
std::cout<<map[j][i]<<" ";
```

```
 }

 std::cout<<std::endl;

 }





}



Mine::~Mine(){

 for(int j = 0 ; j < 30 ; j++ ){

 for(int i = 0 ; i < 30 ; i++ ){

 map[j][i] = 0;

 w_map[j][i] = 0;

 }

 }

 lei = lei_all;

 success = 0;
```

```
}
```

mainwindow.ui

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
 <property name="windowModality">
 <enum>Qt::WindowModal</enum>
 </property>
 <property name="geometry">
 <rect>
 <x>0</x>
 <y>0</y>
 <width>259</width>
```

```
<height>352</height>

</rect>

</property>

<property name="font">

<font>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="contextMenuPolicy">

<enum>Qt::NoContextMenu</enum>

</property>

<property name="windowTitle">

<string>扫雷 ——by: orall</string>

</property>

<widget class="QWidget" name="centralwidget">
```

```xml
<widget class="QPushButton" name="pushButton">
<property name="geometry">
<rect>
<x>110</x>
<y>10</y>
<width>31</width>
<height>31</height>
</rect>
</property>
<property name="font">
<font>
<family>Microsoft YaHei UI Light</family>
<weight>75</weight>
<bold>true</bold>
</font>
</property>
```

```
<property name="text">

<string/>

</property>

</widget>

<widget class="QLCDNumber" name="lcdNumber">

<property name="geometry">

<rect>

<x>150</x>

<y>10</y>

<width>64</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<family>Arial</family>
```

```
<pointsize>9</pointsize>

<weight>50</weight>

<bold>false</bold>

</font>

</property>

<property name="frameShape">

<enum>QFrame::Box</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Raised</enum>

</property>

<property name="lineWidth">

<number>1</number>

</property>

<property name="midLineWidth">

<number>0</number>
```

```
</property>

</widget>

<widget class="QLCDNumber" name="lcdNumber_2">

<property name="geometry">

<rect>

<x>30</x>

<y>10</y>

<width>61</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<weight>50</weight>

<bold>false</bold>

</font>
```

```xml
</property>
<property name="frameShape">
<enum>QFrame::Box</enum>
</property>
<property name="frameShadow">
<enum>QFrame::Raised</enum>
</property>
<property name="lineWidth">
<number>1</number>
</property>
<property name="digitCount">
<number>5</number>
</property>
<property name="intValue" stdset="0">
<number>10</number>
</property>
```

```
</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>259</width>

<height>25</height>

</rect>

</property>

<widget class="QMenu" name="menu">

<property name="title">

<string>游戏</string>

</property>

<addaction name="separator"/>
```

```
<addaction name="action_2"/>

<addaction name="action_4"/>

<addaction name="action_5"/>

<addaction name="action_6"/>

<addaction name="separator"/>

</widget>

<addaction name="menu"/>

</widget>

<widget class="QStatusBar" name="statusbar"/>

<action name="action_2">

<property name="text">

<string>初级</string>

</property>

</action>

<action name="action_4">

<property name="text">
```

```xml
<string>中级</string>
</property>
</action>
<action name="action_5">
<property name="text">
<string>高级</string>
</property>
</action>
<action name="action_6">
<property name="text">
<string>游戏说明</string>
</property>
</action>
</widget>
<resources/>
<connections/>
```

```
</ui>
```

# 6.扫雷3.0版本（2021.10.14）

该版本完善了：
1.点击按钮的时候，时间不动（把两个QTimer改成其他）
bug：
1.点击的时候按钮会恢复
2.第二次开始时，点击按钮没反应
mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H


#include <QMainWindow>
#include <QPainter>
#include <QPushButton>
#include <QLabel>
#include <QMouseEvent>
```

```
#include <QTimer> // 定时器对象
#include "Mine.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
 Q_OBJECT

public:
 QPushButton *MainButton[30][30];
 Mine *mine; //地雷
 int retainx,retainy; //鼠标点击的位置
 int isStart; //是否开始游戏
 int uigrade; //难度
 int seconds; //用时


 //定义需要用到的定时器（第一个用来计时，第二个用来鼠标点击）
 int Timer1; //用来计算时间
 int Timer2; //用来判断鼠标点击是单击还是双击
 int ClickedState; //点击状态，常态不点击为0，单击对应1，双击对应2


 MainWindow(QWidget *parent = nullptr);
 ~MainWindow();


 void chordingShow(int x,int y); //双击的展示
 void leftShow(int x,int y); //左键单击的显示
 void showtime(); //更新时钟
 void showButton(int i,int j,int model);
 void startTime(); //开始计时
```

```
 void empty(); //清空之前创建的东西
 void mousePressEvent(QMouseEvent *event); //鼠标点击事件
 void mouseDoubleClickEvent(QMouseEvent *event); //鼠标双击事件
 void newGame(int difficult); //初始化游戏
 void failShow(int ii,int jj); //失败的时候显示所有的雷
 void ButtonClick(int i,int j); //按钮点击
 void timerSingleClick(); //定时器事件，处理单击任务
 int* reverseij(int i,int j); //保存当前坐标
private slots:
 void timerEvent(QTimerEvent *event);
 void on_pushButton_clicked();
 void on_action_2_triggered();
 void on_action_4_triggered();
 void on_action_5_triggered();


private:
 Ui::MainWindow *ui;
};


#endif // MAINWINDOW_H
```

Mine.h

```


#ifndef MINE_H

#define MINE_H



#include<iostream>

#include<time.h>

#include<stdlib.h>

#include <QMessageBox>



class Mine{ //扫雷的类

public:

 int map[30][30]; //初级（初级为9*9个方块10个雷） //雷表示-1
 int w_map[30][30]; //显示出来的地图
 int lei; //右键旗的次数
 int lei_all; //雷的个数
 int success; //成功全部雷即为胜利
 int y_N,x_N; //map[y_N][x_N]
 int grade; //等级(初级，中级，高级)
 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方块99个雷

 Mine(int _grade);
 ~Mine();

 void newMine(); //产生新的雷
 void calcNum(); //计算雷的数量
 void left(int x, int y); //左击
 void right(int x, int y); //右击
```

```
// void chording(int x, int y); //双击

};




#endif // MINE_H
```

main.cpp

```


#include "mainwindow.h"



#include <QApplication>



int main(int argc, char *argv[])

{

 QApplication a(argc, argv);

 MainWindow w;

 w.show();

 return a.exec();

}
```

mainwindow.cpp

```


#include "mainwindow.h"

#include "ui_mainwindow.h"



MainWindow::MainWindow(QWidget *parent)

 : QMainWindow(parent)

 , ui(new Ui::MainWindow)

{

 ui->setupUi(this);

 uigrade = 1; //初始化等级为1

 newGame(1); //初始化游戏

}
```

```
MainWindow::~MainWindow()
{
 delete ui;
}


//显示游戏时间（计时）
void MainWindow::showtime(){
 // ui->textEdit_2->setText("str");
 seconds++;
 QString str;
 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html
 str = QString("%01").arg(seconds, 3, 10, QLatin1Char('0'));
 ui->lcdNumber->display(str);
}


//开始计时
```

```
void MainWindow::startTime(){
 Timer1 = startTimer(1000);
 /*
 myTimer->start(1000);
 //启动定时器
 //时间间隔为100ms
 //每隔100ms，定时器myTimer自动触发timeout()
 //如果定时器没有激活，才启动,防止多次点击start按钮出现错误
 if (myTimer->isActive() == false)
 {
 isStart = 1; //标记为开始游戏
 }*/
}


//清空本来生成的东西
void MainWindow::empty(){
 killTimer(Timer1); //先把计时停了



 //qDebug()<<"12121";

 //在重新初始化之前，先删除之前的new出来的东西，否则会出现虚幻的雷之类
的

 //把new出来的按钮delete掉

 for(int j = 0 ; j < mine->y_N ; j ++ ){

 for(int i = 0 ; i < mine->x_N ; i ++ ){

 delete MainButton[j][i];

 }

 }

 //把雷delete掉

 delete mine;

}



//点击笑脸按钮

void MainWindow::on_pushButton_clicked()
```

```
{

 //qDebug()<<"尝试重新开始游戏";

 if( isStart != 0 ){ //设置必须失败才能重新开始

 empty(); //清空本来的东西

 newGame(uigrade); //生成新的

 }

}



//双击的展示（用递归实现最佳）

void MainWindow::chordingShow(int x,int y){

 if( x < 0 || y < 0 || x >= mine->x_N || y >= mine->y_N ) return;

 if( mine->map[y][x] == 0 && mine->w_map[y][x] == 0 ){ //只有0且还没打开才能双击

 leftShow(x,y);

 mine->left(x,y);

 //对四周（注意加上还没打开的限制，否则会报错）
 chordingShow(x+1,y);

 chordingShow(x,y+1);

 chordingShow(x,y-1);

 chordingShow(x-1,y);



 }else if( mine->map[y][x] != -1 ){ //如果是其他，就只能左键点击

 leftShow(x,y);

 mine->left(x,y);

 }

 return;



}



//扫雷失败会把所有的雷位置展示出来

void MainWindow::failShow(int ii,int jj) //参数为点击的位置（标红）
```

```
{


 isStart = -1; //表示为游戏失败
 ui->pushButton->setText("☹");
 //if (myTimer->isActive()) myTimer->stop();
 for(int j = 0 ; j < mine->y_N ; j ++ ){
 for(int i = 0 ; i < mine->x_N ; i ++ ){
 //失败还得把按钮都变成不可点击
 if(mine->w_map[j][i] == 0 ){ //加上判断，避免失败后一些按钮会恢复
 MainButton[j][i]->setEnabled(false);
 }


 if( mine->map[j][i] == -1 ){
 MainButton[j][i]->setStyleSheet("background-
color:#c2d0ea;font-size:10px;font-weight:bold;");
 if( ii == i && jj == j ){ //点击的位置标红
 MainButton[j][i]->setStyleSheet("color:#1212f9;background-
color:#ff0000;font-size:10px;font-weight:bold;");
 }
 mine->w_map[j][i] = -1;
 this->MainButton[j][i]->setDown(true);
 this->MainButton[j][i]->setText("💣");
 }
 }
 }



}



//初始化游戏
void MainWindow::newGame(int difficult){
 seconds=0;
```

```
int move_x,move_y;
if( difficult == 1 ){
this->resize(220,260);
move_x = 20,move_y = 65;
ui->lcdNumber_2->move(20,10);
ui->pushButton->move(90,10);
ui->lcdNumber->move(130,10);

}else if( difficult == 2 ){
this->resize(360,400);
move_x = 20,move_y = 65;
ui->lcdNumber_2->move(20,10);
ui->pushButton->move(160,10);
ui->lcdNumber->move(270,10);

}else if( difficult == 3 ){
```

```
this->resize(640,400);

move_x = 20,move_y = 65;

ui->lcdNumber_2->move(20,10);

ui->pushButton->move(300,10);

ui->lcdNumber->move(550,10);



}



//qDebug()<<"重新游戏";

isStart = 0; //尚未点击开始按钮



//设置背景颜色(#dadfec;)#91a0b3;

this->setStyleSheet("background-color:#cfd8df");



ui->lcdNumber->setStyleSheet("background-color:#000000;color:#ff0000;");

ui->lcdNumber_2->setStyleSheet("background-
color:#000000;color:#ff0000;");



 mine = new Mine(difficult); //生成地雷

 //Qt按钮设置：https://baijiahao.baidu.com/s?
id=1691291899564946968&wfr=spider&for=pc



 int size = 20;



 //注意是[y][x]对应[j][i]



 for(int j = 0 ; j < mine->y_N ; j++ ){

 for(int i = 0 ; i < mine->x_N ; i++ ){

 MainButton[j][i] = new QPushButton(" ",this);

 //设置按钮颜色border:2px solid #808080 rgb(192,192,192);#77a7f0
#3547b2;

 MainButton[j][i]->setStyleSheet("background-color:#6fabee");

 MainButton[j][i]->resize(size,size);
 //MainButton[j][i]->setEnabled(false);

 MainButton[j][i]->move(move_x+size*(i),move_y+size*(j)); //将按钮b1移动到点（50，50）的位置

 // MainButton[j][i]->setParent("1",this);



 MainButton[j][i]->setCheckable(true);

 MainButton[j][i]->setAutoExclusive(true);



 MainButton[j][i]->show();

 //给connect传参数的方法：https://blog.csdn.net/qq_38188725/article/details/80686793

 //解决鼠标左键无法响应鼠标事件，和connect无法响应

 connect(MainButton[j][i], &QPushButton::clicked,this,

 [=]{ButtonClick(i,j);

 });
 }

 }



 ui->pushButton->setText("☺");

 ui->pushButton->resize(30,30);



 //设置lcd位数

 ui->lcdNumber->setDigitCount(4);

 ui->lcdNumber_2->setDigitCount(3);

 QString tmp = QString("%1").arg(mine->lei, 3, 10,
QLatin1Char('0')); //整型转字符串); //整型转字符串;

 ui->lcdNumber_2->display(tmp);



 ui->lcdNumber->display("000");



 /*

 //显示游戏时间
 myTimer = new QTimer(this);

 connect(myTimer,&QTimer::timeout, this, [=]{

 showtime();});

 */



 //解决鼠标单击还是双击的

 ClickedState = 0; //初始化不点击

 /*

 timer = new QTimer(this);

 connect(timer, &QTimer::timeout, this,[ = ]{

 timerSingleClick();

 });*/

}



void MainWindow::mousePressEvent(QMouseEvent *event){

 Timer2 = startTimer(180);

 //timer->start(300);



 //鼠标事件参考链接:https://www.cnblogs.com/aiguona/p/9999507.html

 qDebug("鼠标点击");

 // 如果是鼠标左键按下

 if(event->button() == Qt::LeftButton)

 {

 qDebug("鼠标左键点击");



 for( int j = 0 ; j < mine->y_N ; j ++ ){

 for( int i = 0 ; i < mine->x_N ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

 qDebug("%d,%d",i,j);

 showButton(i,j,1);
```

```
}



}

}

}

// 如果是鼠标右键按下

else if(event->button() == Qt::RightButton)

{

qDebug("鼠标右键点击");

//timer->stop();

for( int j = 0 ; j < mine->y_N ; j ++ ){

for( int i = 0 ; i < mine->x_N ; i ++ ){

if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

qDebug("%d,%d",i,j);

showButton(i,j,2);
```

```
 }

 }
 }

 }
}
//鼠标双击事件
void MainWindow::mouseDoubleClickEvent(QMouseEvent *event){
 // 如果是鼠标左键按下（双击）
 if(event->button() == Qt::LeftButton){
 //qDebug("鼠标左键双击");
 for( int j = 0 ; j < mine->y_N ; j ++ ){
 for( int i = 0 ; i < mine->x_N ; i ++ ){
 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){
 //qDebug("%d,%d",i,j);
 chordingShow(i,j);

 showButton(i,j,3);

 }



 }

 }

 }

}



void MainWindow::leftShow(int x,int y){ //左键单击的显示

 int i = x,j = y;

 if( mine->map[j][i] == 0 ){ //rgb(192,192,192)

 MainButton[j][i]->setStyleSheet("background-color:#c2d0ea;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText("");

 }else if( mine->map[j][i] == 1 ){
```

```
 //设置按钮颜色border:1px solid #999999;background-color:#c0c0c0

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 2 ){

 MainButton[j][i]->setStyleSheet("color:#008000;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 3 ){

 MainButton[j][i]->setStyleSheet("color:#fe0202;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else{

 MainButton[j][i]->setStyleSheet("color:#131386;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));
 }

 this->MainButton[j][i]->setDown(true);



}

//按钮点击触发的事件

void MainWindow::showButton(int i, int j, int model){

 // if( isStart == 1 ){



 if( isStart == 0 ){ //如果还没未开始，则现在开始计时

 startTime();

 }



 this->MainButton[j][i]->setEnabled(false); //已点击过的按钮无法
再点击

 //注意这里的i和j对应x,y

 qDebug("%d,%d",i,j);

 if( model == 1 ){ //左键点击
```

```
//w_map为0说明没打开，没打开的才能点击

if( mine->w_map[j][i] == 0 ){

qDebug("1111");



//this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

if( mine->map[j][i] == -1 ){ //踩到雷

MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

//此处应该插入游戏失败的代码

//this->MainButton[j][i]->setText("💣");

failShow(i,j); //失败+显示所有的雷

}else{

if( mine->map[j][i] == 0 ){ //如果点击0相当于双击

chordingShow(i,j);

}else{

leftShow(i,j);
}

}

this->MainButton[j][i]->setDown(true);

mine->left(i,j);



//设置按钮为按下状态：https://blog.csdn.net/qiangzi4646/article/details/80135120



}



// this->MainButton[j][i]->setText("");

}else if( model == 2 ){ //右键点击（标记）

//w_map为0说明没打开，没打开的才能点击

if( mine->w_map[j][i] == 0 ){

//qDebug("2222");

if( mine->lei > 0 ){
 mine->right(i,j);

 this->MainButton[j][i]->setText("🚩");

 }

 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html

 QString L = QString("%1").arg(mine->lei, 3, 10,
QLatin1Char('0')); //整型转字符串

 ui->lcdNumber_2->display(L);

 }

 //应该是设置了不可点击，所以没法撤销

 /*

 else if( mine->w_map[j][i] == -2 ){ //再点一次恢复

 this->MainButton[j][i]->setEnabled(true); //按钮再点击

 if( mine->lei <= 10 ){

 mine->right(i,j);

 this->MainButton[j][i]->setText(" ");

 }

 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html
 QString L = QString("%1").arg(mine->lei, 3, 10, QLatin1Char('0')); //整型转字符串

 ui->lcdNumber_2->display(L);



 }*/

 // ui->textEdit->setText(mine->);

 }else if( model == 3 ){

 qDebug("3333");

 //mine->chording(i,j); //双击

 this->MainButton[j][i]->setText(" ");

 }

 // }



}



void MainWindow::timerEvent(QTimerEvent *event){
 //QTimerEvent:https://www.cnblogs.com/Yu-nan/p/12091264.html
 //用两个QTimer不对劲

 if(event->timerId() == Timer1) {
 //do Timer1_Fuc
 showtime();
 }
 if(event->timerId() == Timer2) {
 //do Timer2_Fuc
 timerSingleClick();
 }


}
```

```
void MainWindow::ButtonClick(int i,int j){

 ClickedState ++;

 retainx = i ,retainy = j;

 if(ClickedState == 1)

 Timer2 = startTimer(180);

 //timer->start(180); //430ms为自己设置



 if(ClickedState == 2)

 {

 ClickedState = 0; //初始化

 killTimer(Timer2);

 //timer->stop(); //停止定时器继续计时

 //按钮双击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("我是双击！！！"),tr("确定"));

 showButton(i,j,3);

 }
}


//https://blog.csdn.net/wdmzslh/article/details/53301495
void MainWindow::timerSingleClick()
{
 ClickedState = 0; //初始化
 //timer->stop(); //定时器停止计时
 killTimer(Timer2);


 //按钮单击响应代码放这里
 //QMessageBox::warning(this,tr("警告"),tr("单击"),tr("确定"));
 showButton(retainx,retainy,1);
}
```

```
//初级被点击
void MainWindow::on_action_2_triggered()
{
 empty();
 uigrade = 1;
 newGame(1);
}

//中级被点击
void MainWindow::on_action_4_triggered()
{
 empty();
 uigrade = 2;
 newGame(2);
}
```

```
//高级被点击
void MainWindow::on_action_5_triggered()
{
 empty();
 uigrade = 3;
 newGame(3);
}
```

Mine.cpp

```
//注意是map[y][x]
#include "Mine.h"
#include "ui_mainwindow.h"

//产生新的雷（map中-1表示雷）
```

```
void Mine::newMine() {

 //lei = 10;

 success = 0;

 int tmp = 0; //随机生成对应雷

 srand((unsigned)time(NULL));

 while(1){

 int x = rand() % x_N; //产生0到8的数字

 int y = rand() % y_N;

 if (map[y][x] != -1) {

 map[y][x] = -1; //埋下地雷

 tmp++;

 }

 if( tmp >= lei_all ) break;

 }
```

```
}



//根据雷补充map地图

void Mine::calcNum() { //计算雷的数量(实际用map[j][i]更佳，不容易混)

 //根据展开方格的数字去推断其相邻九宫格内未展开方格下面是否是地雷



 //填入数字

 for(int j = 0 ; j < y_N ; j ++ ){

 for(int i = 0 ; i < x_N ; i++ ){

 if( map[j][i] != -1 ){ //如果不是雷就计算周围的雷

 int tmp = 0;

 //周围一圈

 if (i - 1 >= 0 && j + 1 < y_N && map[j + 1][i - 1] == -1) tmp++;

 if (j + 1 < y_N && map[j + 1][i] == -1) tmp++;

 if (j + 1 < y_N && i + 1 < x_N && map[j + 1][i + 1] == -1)
tmp++;



 if (i - 1 >= 0 && map[j][i - 1] == -1) tmp++;

 if (i + 1 < x_N && map[j][i + 1] == -1) tmp++;



 if (i - 1 >= 0 && j - 1 >= 0 && map[j - 1][i - 1] == -1)
tmp++;

 if (j - 1 >= 0 && map[j - 1][i] == -1) tmp++;

 if (i + 1 < x_N && j - 1 >= 0 && map[j - 1][i + 1] == -1)
tmp++;

 map[j][i] = tmp;

 }

 }

 }

}



void Mine::left(int x, int y) {
```

```
//☺🙁☹

//左键用于打开安全的格子，推进游戏进度；

if (map[y][x] == -1) {

//信息框（https://www.cnblogs.com/zhoug2020/p/10094855.html，注意加上头文件）



QMessageBox::information(NULL, "游戏结束", "很遗憾，你已经踩到雷了，游戏结束！");

//cout << "你已经踩雷，游戏失败" << endl;

}

else {

if (map[y][x] == 0) {

w_map[y][x] = -3; //-3表示为空

}

else {

w_map[y][x] = map[y][x];
```

```
 }

 }



}



void Mine::right(int x, int y) {

 //后续补充再次右键可取消

 if( lei > 0 ){ //如果标记够用的话

 lei--; //标记雷的数量减1

 //右键用于标记地雷，以辅助判断，或为接下来的双击做准备；(-2表示标记)

 if( w_map[y][x] == -2 ){

 w_map[y][x] = 0;

 }else{

 w_map[y][x] = -2;

 }

 if( map[y][x] == -1 ){ //如果找到的是雷

 success++;

 }



 }else{

 if( success == lei_all ){ //成功找到所有的雷

 //此处应该贴胜利

 QMessageBox::information(NULL, "游戏结束", "恭喜你，游戏胜利！");

 }



 }

}

/*

void Mine::chording(int x, int y) {

 //双击只能双击没有数字的

 // 双击在一个数字周围的地雷标记完时， 相当于对数字周围未打开的方块均
```

```
进行一次左键单击操作：

 //标记的雷数一定要等于X，否则双击是打不开的。

 //把周围没标记的都打开（8个）

 if (x + 1 < 9 && w_map[y][x + 1] != -2) left(x + 1, y);

 if (x + 1 < 9 && y - 1 >= 0 && w_map[y-1][x + 1] != -2) left(x + 1, y-1);

 if (x + 1 < 9 && y + 1 < 9 && w_map[y+1][x + 1] != -2) left(x + 1, y+1);





 if ( y - 1 >= 0 && w_map[y - 1][x] != -2) left(x, y - 1);

 if ( y + 1 < 9 && w_map[y + 1][x] != -2) left(x, y + 1);



 if (x - 1 >= 0 && w_map[y][x - 1] != -2) left(x + 1, y);

 if (x - 1 >= 0 && y - 1 >= 0 && w_map[y - 1][x - 1] != -2) left(x - 1, y - 1);

 if (x - 1 >= 0 && y + 1 < 9 && w_map[y + 1][x - 1] != -2) left(x - 1, y + 1);
```

```
}

*/

Mine::Mine(int _grade):grade(_grade){

 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方块99个雷

 if( grade == 1 ){

 y_N = 9;

 x_N = 9;

 lei = lei_all = 10; //旗的个数和雷的个数

 }else if( grade == 2 ){

 y_N = 16;

 x_N = 16;

 lei = lei_all = 40; //旗的个数和雷的个数



 }else if( grade == 3 ){

 y_N = 16;

 x_N = 30;
 lei = lei_all = 99; //旗的个数和雷的个数


 }


 newMine();

 calcNum();

}


Mine::~Mine(){

 for(int j = 0 ; j < 30 ; j++ ){

 for(int i = 0 ; i < 30 ; i++ ){

 map[j][i] = 0;

 }

 }

 lei = lei_all;

 success = 0;
```

```
}
```

# 5.扫雷2.0版本（2021.10.13）

该版本增加了：
1.界面的显示
2.难度的增加
待修复
1.右键再点一次可以撤销
bug：
1.点击按钮的时候，时间不动

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H


#include <QMainWindow>
#include <QPainter>
#include <QPushButton>
```

```
#include <QLabel>
#include <QMouseEvent>
#include <QTimer> // 定时器对象
#include "Mine.h"


QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE


class MainWindow : public QMainWindow
{
 Q_OBJECT


public:
 //QLabel *MainLabel[9][9];
 QPushButton *MainButton[30][30];
```

```
Mine *mine; //地雷
int retainx,retainy; //鼠标点击的位置
int isStart; //是否开始游戏
int uigrade; //难度

MainWindow(QWidget *parent = nullptr);
~MainWindow();
void chordingShow(int x,int y); //双击的展示
void leftShow(int x,int y); //左键单击的显示
void showtime(); //更新时钟
void showButton(int i,int j,int model);
void startTime(); //开始计时
void empty();
void mousePressEvent(QMouseEvent *event);
void mouseDoubleClickEvent(QMouseEvent *event);
void newGame(int difficult); //初始化游戏
```

```
 void failShow(int ii,int jj); //失败的时候显示所有的雷
 void ButtonClick(int i,int j);
 void timerSingleClick(); //定时器事件，处理单击任务
 int* reverseij(int i,int j); //保存当前坐标
private slots:
 void on_pushButton_clicked();
 void on_action_2_triggered();
 void on_action_4_triggered();
 void on_action_5_triggered();

private:
 Ui::MainWindow *ui;



 QTimer *timer; //判断双击还是单击
 int ClickedState; //点击状态，常态不点击为0，单击对应1，双击对应2
 QTimer *myTimer;// 定时器对象
};

#endif // MAINWINDOW_H
```

Mine.h

```
#ifndef MINE_H
#define MINE_H

#include<iostream>
#include<time.h>
#include<stdlib.h>
#include <QMessageBox>

class Mine{ //扫雷的类
```

```
public:
 int map[30][30]; //初级（初级为9*9个方块10个雷） //雷表示-1
 int w_map[30][30]; //显示出来的地图
 int lei; //右键旗的次数
 int lei_all; //雷的个数
 int success; //成功全部雷即为胜利
 int y_N,x_N; //map[y_N][x_N]
 int grade; //等级(初级，中级，高级)
 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方块99个雷


 Mine(int _grade);
 ~Mine();


 void newMine(); //产生新的雷
 void calcNum(); //计算雷的数量
 void left(int x, int y); //左击

 void right(int x, int y); //右击

// void chording(int x, int y); //双击

};




#endif // MINE_H
```

main.cpp

```


#include "mainwindow.h"



#include <QApplication>



int main(int argc, char *argv[])

{

 QApplication a(argc, argv);
 MainWindow w;

 w.show();

 return a.exec();

}
```

mainwindow.cpp

```


#include "mainwindow.h"

#include "ui_mainwindow.h"



MainWindow::MainWindow(QWidget *parent)

 : QMainWindow(parent)

 , ui(new Ui::MainWindow)

{




 ui->setupUi(this);
 uigrade = 1;
 newGame(1); //初始化游戏
}

MainWindow::~MainWindow()
{
 delete ui;
}

//显示游戏时间
void MainWindow::showtime(){
 // ui->textEdit_2->setText("str");
 static int seconds=0;
 seconds++;

 QString str;
 //前位补0：https://www.cnblogs.com/zzzsj/p/14566157.html
 str = QString("%01").arg(seconds, 3, 10, QLatin1Char('0'));
 ui->lcdNumber->display(str);
}


//开始计时
void MainWindow::startTime(){
 myTimer->start(1000);
 //启动定时器
 //时间间隔为100ms
 //每隔100ms，定时器myTimer自动触发timeout()
 //如果定时器没有激活，才启动,防止多次点击start按钮出现错误
 if (myTimer->isActive() == false)
 {
 isStart = 1; //标记为开始游戏
 }
```

```
}



//清空本来生成的东西

void MainWindow::empty(){



 if (myTimer->isActive()) myTimer->stop();

 delete myTimer;



 qDebug()<<"12121";

 //在重新初始化之前，先删除之前的new出来的东西，否则会出现虚幻的雷之类的

 for(int j = 0 ; j < mine->y_N ; j ++ ){

 for(int i = 0 ; i < mine->x_N ; i ++ ){

 delete MainButton[j][i];

 }

 }

 delete mine;
```

```
}


//点击笑脸按钮
void MainWindow::on_pushButton_clicked()
{
 qDebug()<<"尝试重新开始游戏";
 if( isStart != 0 ){ //先清空本来的东西在生成新的
 empty();
 newGame(uigrade);
 }

}


//双击的展示
void MainWindow::chordingShow(int x,int y){
 if( x < 0 || y < 0 || x >= mine->x_N || y >= mine->y_N )
```

```
return;

 if( mine->map[y][x] == 0 && mine->w_map[y][x] == 0 ){ //只有0且还没打开才能双击

 leftShow(x,y);

 mine->left(x,y);

 //对四周（注意加上还没打开的限制，否则会报错）

 chordingShow(x+1,y);

 chordingShow(x,y+1);

 chordingShow(x,y-1);

 chordingShow(x-1,y);



 }else if( mine->map[y][x] != -1 ){

 leftShow(x,y);

 mine->left(x,y);

 }

 return;
```

```
}

//扫雷失败会把所有的雷位置展示出来

void MainWindow::failShow(int ii,int jj) //参数为点击的位置（标红）

{

 isStart = -1; //表示为游戏失败

 ui->pushButton->setText("🙁");

 if (myTimer->isActive()) myTimer->stop();

 for(int j = 0 ; j < mine->y_N ; j ++ ){

 for(int i = 0 ; i < mine->x_N ; i ++ ){

 //失败还得把按钮都变成不可点击

 if(mine->w_map[j][i] == 0 ){ //加上判断，避免失败后一些按钮会恢复

 MainButton[j][i]->setEnabled(false);
 }



 if( mine->map[j][i] == -1 ){

 if( ii == i && jj == j ){ //点击的位置标红

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#ff0000;font-size:10px;font-weight:bold;");

 }

 mine->w_map[j][i] = -1;

 this->MainButton[j][i]->setDown(true);

 this->MainButton[j][i]->setText("💣");

 }

 }

 }




}
```

```
//初始化游戏

void MainWindow::newGame(int difficult){


 int move_x,move_y;

 if( difficult == 1 ){

 this->resize(220,260);

 move_x = 20,move_y = 65;

 ui->lcdNumber_2->move(20,10);

 ui->pushButton->move(90,10);

 ui->lcdNumber->move(130,10);


 }else if( difficult == 2 ){

 this->resize(360,400);

 move_x = 20,move_y = 65;

 ui->lcdNumber_2->move(20,10);
 ui->pushButton->move(160,10);
 ui->lcdNumber->move(270,10);


 }else if( difficult == 3 ){
 this->resize(640,400);
 move_x = 20,move_y = 65;
 ui->lcdNumber_2->move(20,10);
 ui->pushButton->move(300,10);
 ui->lcdNumber->move(550,10);


 }


 qDebug()<<"重新游戏";
 isStart = 0; //尚未点击开始按钮


 //设置背景颜色(#dadfec;)#91a0b3;
```

```
 this->setStyleSheet("background-color:#cfd8df");



 ui->lcdNumber->setStyleSheet("background-color:#000000;color:#ff0000;");

 ui->lcdNumber_2->setStyleSheet("background-color:#000000;color:#ff0000;");



 mine = new Mine(difficult); //生成地雷

 //Qt按钮设置：https://baijiahao.baidu.com/s?id=1691291899564946968&wfr=spider&for=pc



 int size = 20;



 //注意是[y][x]对应[j][i]



 for(int j = 0 ; j < mine->y_N ; j++ ){

 for(int i = 0 ; i < mine->x_N ; i++ ){
```

```
 MainButton[j][i] = new QPushButton(" ",this);

 //设置按钮颜色border:2px solid #808080 rgb(192,192,192);#77a7f0 #3547b2;

 MainButton[j][i]->setStyleSheet("background-color:#6fabee");

 MainButton[j][i]->resize(size,size);

 //MainButton[j][i]->setEnabled(false);

 MainButton[j][i]->move(move_x+size*(i),move_y+size*(j)); //将按钮b1移动到点（50，50）的位置

 // MainButton[j][i]->setParent("1",this);



 MainButton[j][i]->setCheckable(true);

 MainButton[j][i]->setAutoExclusive(true);



 MainButton[j][i]->show();

 //给connect传参数的方法：https://blog.csdn.net/qq_38188725/article/details/80686793

 //解决鼠标左键无法响应鼠标事件，和connect无法响应

 connect(MainButton[j][i], &QPushButton::clicked,this,
 [=]{ButtonClick(i,j);

 });





 }

 }



 ui->pushButton->setText("☺");

 ui->pushButton->resize(30,30);



 //设置lcd位数

 ui->lcdNumber->setDigitCount(4);

 ui->lcdNumber_2->setDigitCount(3);

 QString tmp = QString("%1").arg(mine->lei, 3, 10,
QLatin1Char('0')); //整型转字符串); //整型转字符串;

 ui->lcdNumber_2->display(tmp);
 ui->lcdNumber->display("000");



 //显示游戏时间

 myTimer = new QTimer(this);

 connect(myTimer,&QTimer::timeout, this, [=]{

 showtime();});





 //解决鼠标单击还是双击的

 ClickedState = 0; //初始化不点击

 timer = new QTimer(this);

 connect(timer, &QTimer::timeout, this,[ = ]{

 timerSingleClick();

 });

}
```

```
void MainWindow::mousePressEvent(QMouseEvent *event){



 timer->start(300);



 //鼠标事件参考链接:
https://www.cnblogs.com/aiguona/p/9999507.html

 qDebug("鼠标点击");

 // 如果是鼠标左键按下

 if(event->button() == Qt::LeftButton)

 {

 qDebug("鼠标左键点击");



 for( int j = 0 ; j < mine->y_N ; j ++ ){

 for( int i = 0 ; i < mine->x_N ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this-
>mapFromGlobal(QCursor::pos()))){
```

```
qDebug("%d,%d",i,j);

showButton(i,j,1);

}



}

}

}

// 如果是鼠标右键按下

else if(event->button() == Qt::RightButton)

{

qDebug("鼠标右键点击");

timer->stop();

for( int j = 0 ; j < mine->y_N ; j ++ ){

for( int i = 0 ; i < mine->x_N ; i ++ ){

if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

qDebug("%d,%d",i,j);
```

```
 showButton(i,j,2);

 }


 }

 }


 }

}

//鼠标双击事件

/*

void MainWindow::mouseDoubleClickEvent(QMouseEvent *event)

{

 //QString str="("+QString::number(e->x())+","+QString::number(e->y())+")";

 qDebug("鼠标双击");

```

```
}

*/

void MainWindow::mouseDoubleClickEvent(QMouseEvent *event){

 // 如果是鼠标左键按下（双击）

 if(event->button() == Qt::LeftButton){

 qDebug("鼠标左键双击");

 for( int j = 0 ; j < mine->y_N ; j ++ ){

 for( int i = 0 ; i < mine->x_N ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

 qDebug("%d,%d",i,j);

 chordingShow(i,j);

 showButton(i,j,3);

 }



 }

 }
 }

}



void MainWindow::leftShow(int x,int y){ //左键单击的显示

 int i = x,j = y;

 if( mine->map[j][i] == 0 ){ //rgb(192,192,192)

 MainButton[j][i]->setStyleSheet("background-color:#c2d0ea;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText("");

 }else if( mine->map[j][i] == 1 ){

 //设置按钮颜色border:1px solid #999999;background-color:#c0c0c0

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 2 ){

 MainButton[j][i]->setStyleSheet("color:#008000;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");
 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 3 ){

 MainButton[j][i]->setStyleSheet("color:#fe0202;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else{

 MainButton[j][i]->setStyleSheet("color:#131386;background-color:#c2d0ea;;font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }

 this->MainButton[j][i]->setDown(true);



}

//按钮点击触发的事件

void MainWindow::showButton(int i, int j, int model){
 // if( isStart == 1 ){



 if( isStart == 0 ){ //如果还没未开始，则现在开始计时

 startTime();

 }



 this->MainButton[j][i]->setEnabled(false); //已点击过的按钮无法再点击

 //注意这里的i和j对应x,y

 qDebug("%d,%d",i,j);

 if( model == 1 ){ //左键点击

 //w_map为0说明没打开，没打开的才能点击

 if( mine->w_map[j][i] == 0 ){

 qDebug("1111");



 //this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));
```

```
if( mine->map[j][i] == -1 ){ //踩到雷
MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");
//此处应该插入游戏失败的代码
//this->MainButton[j][i]->setText("💣");
failShow(i,j); //失败+显示所有的雷
}else{
if( mine->map[j][i] == 0 ){ //如果点击0相当于双击
chordingShow(i,j);
}else{
leftShow(i,j);
}
}
this->MainButton[j][i]->setDown(true);
mine->left(i,j);


//设置按钮为按下状态:
```

```
https://blog.csdn.net/qiangzi4646/article/details/80135120



 }



 // this->MainButton[j][i]->setText("");

 }else if( model == 2 ){ //右键点击（标记）

 //w_map为0说明没打开，没打开的才能点击

 if( mine->w_map[j][i] == 0 ){

 //qDebug("2222");

 if( mine->lei > 0 ){

 mine->right(i,j);

 this->MainButton[j][i]->setText("🚩");

 }

 //前位补0：https://www.cnblogs.com/zzzsj/p/14566157.html

 QString L = QString("%1").arg(mine->lei, 3, 10,
QLatin1Char('0')); //整型转字符串

 ui->lcdNumber_2->display(L);
 }

 // ui->textEdit->setText(mine->);

 }else if( model == 3 ){

 qDebug("3333");

 //mine->chording(i,j); //双击

 this->MainButton[j][i]->setText(" ");

 }

 // }



}



void MainWindow::ButtonClick(int i,int j){

 ClickedState ++;

 retainx = i ,retainy = j;

 if(ClickedState == 1)

 timer->start(180); //430ms为自己设置
 if(ClickedState == 2)

 {

 ClickedState = 0; //初始化

 timer->stop(); //停止定时器继续计时

 //按钮双击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("我是双击！！！"),tr("确定"));

 showButton(i,j,3);

 }



}



//https://blog.csdn.net/wdmzslh/article/details/53301495

void MainWindow::timerSingleClick()

{
 ClickedState = 0; //初始化

 timer->stop(); //定时器停止计时



 //按钮单击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("单击"),tr("确定"));

 showButton(retainx,retainy,1);

}





//初级被点击

void MainWindow::on_action_2_triggered()

{

 empty();

 uigrade = 1;

 newGame(1);

}
```

```
//中级被点击
void MainWindow::on_action_4_triggered()
{
 empty();
 uigrade = 2;
 newGame(2);
}



//高级被点击
void MainWindow::on_action_5_triggered()
{
 empty();
 uigrade = 3;
 newGame(3);

}
```

Mine.cpp

```



//注意是map[y][x]

#include "Mine.h"

#include "ui_mainwindow.h"




//产生新的雷（map中-1表示雷）

void Mine::newMine() {

 //lei = 10;

 success = 0;

 int tmp = 0; //随机生成对应雷

 srand((unsigned)time(NULL));

 while(1){

 int x = rand() % x_N; //产生0到8的数字
 int y = rand() % y_N;

 if (map[y][x] != -1) {

 map[y][x] = -1; //埋下地雷

 tmp++;

 }

 if( tmp >= lei_all ) break;

 }




}



//根据雷补充map地图

void Mine::calcNum() { //计算雷的数量(实际用map[j][i]更佳，不容易混)

 //根据展开方格的数字去推断其相邻九宫格内未展开方格下面是否是地雷



 //填入数字
 for(int j = 0 ; j < y_N ; j ++ ){

 for(int i = 0 ; i < x_N ; i++ ){

 if( map[j][i] != -1 ){ //如果不是雷就计算周围的雷

 int tmp = 0;

 //周围一圈

 if (i - 1 >= 0 && j + 1 < y_N && map[j + 1][i - 1] == -1) tmp++;

 if (j + 1 < y_N && map[j + 1][i] == -1) tmp++;

 if (j + 1 < y_N && i + 1 < x_N && map[j + 1][i + 1] == -1) tmp++;



 if (i - 1 >= 0 && map[j][i - 1] == -1) tmp++;

 if (i + 1 < x_N && map[j][i + 1] == -1) tmp++;



 if (i - 1 >= 0 && j - 1 >= 0 && map[j - 1][i - 1] == -1) tmp++;

 if (j - 1 >= 0 && map[j - 1][i] == -1) tmp++;
 if (i + 1 < x_N && j - 1 >= 0 && map[j - 1][i + 1] == -1) tmp++;

 map[j][i] = tmp;

 }

 }

 }

}



void Mine::left(int x, int y) {



 //☺🙁☹

 //左键用于打开安全的格子，推进游戏进度；

 if (map[y][x] == -1) {

 //信息框（https://www.cnblogs.com/zhoug2020/p/10094855.html，注意加上头文件）



 QMessageBox::information(NULL, "游戏结束", "很遗憾，你已经踩到雷了，游戏结束！");
 //cout << "你已经踩雷，游戏失败" << endl;
 }
 else {
 if (map[y][x] == 0) {
 w_map[y][x] = -3; //-3表示为空
 }
 else {
 w_map[y][x] = map[y][x];
 }
 }


}


void Mine::right(int x, int y) {
 //后续补充再次右键可取消
 if( lei > 0 ){ //如果标记够用的话
 lei--; //标记雷的数量减1
 //右键用于标记地雷，以辅助判断，或为接下来的双击做准备；(-2表示标记)
 w_map[y][x] = -2;
 if( map[y][x] == -1 ){ //如果找到的是雷
 success++;
 }


 }else{
 if( success == lei_all ){ //成功找到所有的雷
 //此处应该贴胜利
 QMessageBox::information(NULL, "游戏结束", "恭喜你，游戏胜
利！");
 }


 }
}
```

```
/*

void Mine::chording(int x, int y) {

 //双击只能双击没有数字的

 // 双击在一个数字周围的地雷标记完时， 相当于对数字周围未打开的方块均进行一次左键单击操作：

 //标记的雷数一定要等于X，否则双击是打不开的。

 //把周围没标记的都打开（8个）

 if (x + 1 < 9 && w_map[y][x + 1] != -2) left(x + 1, y);

 if (x + 1 < 9 && y - 1 >= 0 && w_map[y-1][x + 1] != -2) left(x + 1, y-1);

 if (x + 1 < 9 && y + 1 < 9 && w_map[y+1][x + 1] != -2) left(x + 1, y+1);




 if ( y - 1 >= 0 && w_map[y - 1][x] != -2) left(x, y - 1);

 if ( y + 1 < 9 && w_map[y + 1][x] != -2) left(x, y + 1);



 if (x - 1 >= 0 && w_map[y][x - 1] != -2) left(x + 1, y);
 if (x - 1 >= 0 && y - 1 >= 0 && w_map[y - 1][x - 1] != -2) left(x - 1, y - 1);

 if (x - 1 >= 0 && y + 1 < 9 && w_map[y + 1][x - 1] != -2) left(x - 1, y + 1);

}

*/

Mine::Mine(int _grade):grade(_grade){

 //(初级为9*9个方块10个雷，中级为16*16个方块40个雷，高级为16*30个方块99个雷

 if( grade == 1 ){

 y_N = 9;

 x_N = 9;

 lei = lei_all = 10; //旗的个数和雷的个数

 }else if( grade == 2 ){

 y_N = 16;

 x_N = 16;

 lei = lei_all = 40; //旗的个数和雷的个数
 }else if( grade == 3 ){
 y_N = 16;
 x_N = 30;
 lei = lei_all = 99; //旗的个数和雷的个数

 }

 newMine();
 calcNum();
}

Mine::~Mine(){
 for(int j = 0 ; j < 30 ; j++ ){
 for(int i = 0 ; i < 30 ; i++ ){
 map[j][i] = 0;
 }
 }
 lei = lei_all;
 success = 0;

}
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
 <property name="windowModality">
 <enum>Qt::WindowModal</enum>
 </property>
 <property name="geometry">
```

```
<rect>

<x>0</x>

<y>0</y>

<width>259</width>

<height>352</height>

</rect>

</property>

<property name="font">

<font>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="contextMenuPolicy">

<enum>Qt::NoContextMenu</enum>

</property>
```

```
<property name="windowTitle">

<string>扫雷 ——by： orall</string>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="pushButton">

<property name="geometry">

<rect>

<x>110</x>

<y>10</y>

<width>31</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<family>Microsoft YaHei UI Light</family>
```

```
<weight>75</weight>
<bold>true</bold>
</font>
</property>
<property name="text">
<string/>
</property>
</widget>
<widget class="QLCDNumber" name="lcdNumber">
<property name="geometry">
<rect>
<x>150</x>
<y>10</y>
<width>64</width>
<height>31</height>
</rect>
```

```
</property>
<property name="font">
<font>
<family>Arial</family>
<pointsize>9</pointsize>
<weight>50</weight>
<bold>false</bold>
</font>
</property>
<property name="frameShape">
<enum>QFrame::Box</enum>
</property>
<property name="frameShadow">
<enum>QFrame::Raised</enum>
</property>
<property name="lineWidth">
```

```xml
<number>1</number>
</property>
<property name="midLineWidth">
<number>0</number>
</property>
</widget>
<widget class="QLCDNumber" name="lcdNumber_2">
<property name="geometry">
<rect>
<x>30</x>
<y>10</y>
<width>61</width>
<height>31</height>
</rect>
</property>
<property name="font">
```

```xml
<font>
<weight>50</weight>
<bold>false</bold>
</font>
</property>
<property name="frameShape">
<enum>QFrame::Box</enum>
</property>
<property name="frameShadow">
<enum>QFrame::Raised</enum>
</property>
<property name="lineWidth">
<number>1</number>
</property>
<property name="digitCount">
<number>5</number>
```

```
</property>
<property name="intValue" stdset="0">
<number>10</number>
</property>
</widget>
</widget>
<widget class="QMenuBar" name="menubar">
<property name="geometry">
<rect>
<x>0</x>
<y>0</y>
<width>259</width>
<height>25</height>
</rect>
</property>
<widget class="QMenu" name="menu">
```

```
<property name="title">

<string>游戏</string>

</property>

<addaction name="separator"/>

<addaction name="action_2"/>

<addaction name="action_4"/>

<addaction name="action_5"/>

<addaction name="action_6"/>

<addaction name="separator"/>

</widget>

<addaction name="menu"/>

</widget>

<widget class="QStatusBar" name="statusbar"/>

<action name="action_2">

<property name="text">

<string>初级</string>
```

```xml
</property>

</action>

<action name="action_4">

<property name="text">

<string>中级</string>

</property>

</action>

<action name="action_5">

<property name="text">

<string>高级</string>

</property>

</action>

<action name="action_6">

<property name="text">

<string>游戏说明</string>

</property>
```

```
 </action>

 </widget>

 <resources/>

 <connections/>

</ui>
```

## 4.扫雷1.0版本（2021.10.12）

该版本增加了

1.双击的功能

2.胜利的显示

修复了：

1.地图和按钮不对应

2.失败后重新点击笑脸没有重新开始

3.点击后显示慢

4.点到会出现假雷和假旗

5.只有9个雷

待完善：

1.界面的显示

2.难度的增加

3.右键再点一次可以撤销

bug：

1.点击按钮的时候，时间不动

2.点击按钮的时候会恢复（）

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H


#include <QMainWindow>
#include <QPainter>
#include <QPushButton>
#include <QLabel>
#include <QMouseEvent>
#include <QTimer> // 定时器对象
#include "Mine.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE
```

```
class MainWindow : public QMainWindow
{
 Q_OBJECT

public:
 //QLabel *MainLabel[9][9];
 QPushButton *MainButton[9][9];
 Mine *mine; //地雷
 int retainx,retainy; //鼠标点击的位置
 int isStart; //是否开始游戏

 MainWindow(QWidget *parent = nullptr);
 ~MainWindow();
 void chordingShow(int x,int y); //双击的展示
 void leftShow(int x,int y); //左键单击的显示
 void showtime(); //更新时钟

 void showButton(int i,int j,int model);

 void startTime(); //开始计时

 void mousePressEvent(QMouseEvent *event);

 void mouseDoubleClickEvent(QMouseEvent *event);

 void newGame(); //初始化游戏

 void failShow(int ii,int jj); //失败的时候显示所有的雷

 void ButtonClick(int i,int j);

 void timerSingleClick(); //定时器事件，处理单击任务

 int* reverseij(int i,int j); //保存当前坐标

private slots:

 void on_pushButton_clicked();

private:

 Ui::MainWindow *ui;
 QTimer *timer; //判断双击还是单击

 int ClickedState; //点击状态，常态不点击为0，单击对应1，双击对应2



 QTimer *myTimer;// 定时器对象

};



#endif // MAINWINDOW_H
```

Mine.h

```


#ifndef MINE_H

#define MINE_H



#include<iostream>

#include<time.h>

#include<stdlib.h>

#include <QMessageBox>

```

```
class Mine{ //扫雷的类
public:
 int map[9][9]; //初级（初级为9*9个方块10个雷） //雷表示-1
 int w_map[9][9]; //显示出来的地图
 int lei; //右键旗的次数
 int success; //成功找到10个即为胜利

 Mine();
 ~Mine();

 void newMine(); //产生新的雷
 void calcNum(); //计算雷的数量
 void left(int x, int y); //左击
 void right(int x, int y); //右击
// void chording(int x, int y); //双击
};



#endif // MINE_H
```

main.cpp

```

#include "mainwindow.h"


#include <QApplication>


int main(int argc, char *argv[])

{

 QApplication a(argc, argv);

 MainWindow w;

 w.show();

 return a.exec();

}
```

mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"


MainWindow::MainWindow(QWidget *parent)
 : QMainWindow(parent)
 , ui(new Ui::MainWindow)
{


 // ui->lcdNumber->setStyleSheet("background-color:#000000;color:#ff0000;");
 // ui->lcdNumber_2->setStyleSheet("background-color:#000000;color:#ff0000;");
 ui->setupUi(this);
 newGame(); //初始化游戏

}



MainWindow::~MainWindow()

{

 delete ui;

}



//显示游戏时间

void MainWindow::showtime(){

 // ui->textEdit_2->setText("str");

 static int seconds=0;

 seconds++;



 QString str;

 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html
 str = QString("%01").arg(seconds, 3, 10, QLatin1Char('0'));
 ui->lcdNumber->display(str);
}


//开始计时
void MainWindow::startTime(){
 myTimer->start(1000);
 //启动定时器
 //时间间隔为100ms
 //每隔100ms，定时器myTimer自动触发timeout()
 //如果定时器没有激活，才启动,防止多次点击start按钮出现错误
 if (myTimer->isActive() == false)
 {
 isStart = 1; //标记为开始游戏
 }
}
```

```
//点击笑脸按钮

void MainWindow::on_pushButton_clicked()

{

 qDebug()<<"尝试重新开始游戏";



 if( isStart != 0 ){

 if (myTimer->isActive()) myTimer->stop();

 delete myTimer;



 qDebug()<<"12121";

 //在重新初始化之前，先删除之前的new出来的东西，否则会出现虚幻的雷之类的

 for(int j = 0 ; j < 9 ; j ++ ){

 for(int i = 0 ; i < 9 ; i ++ ){
```

```
 delete MainButton[j][i];

 }

 }

 delete mine;



 newGame();

 }



}



//双击的展示

void MainWindow::chordingShow(int x,int y){

 if( x < 0 || y < 0 || x >= 9 || y >= 9 ) return;

 if( mine->map[y][x] == 0 && mine->w_map[y][x] == 0 ){ //只有0且还没打开才能双击

 leftShow(x,y);

 mine->left(x,y);
 //对四周（注意加上还没打开的限制，否则会报错）
 chordingShow(x+1,y);
 chordingShow(x,y+1);
 chordingShow(x,y-1);
 chordingShow(x-1,y);

 }else if( mine->map[y][x] != -1 ){
 leftShow(x,y);
 mine->left(x,y);
 }
 return;

}


//扫雷失败会把所有的雷位置展示出来
```

```
void MainWindow::failShow(int ii,int jj) //参数为点击的位置（标红）
{


 isStart = -1; //表示为游戏失败
 ui->pushButton->setText("🙁");
 if (myTimer->isActive()) myTimer->stop();
 for(int j = 0 ; j < 9 ; j ++ ){
 for(int i = 0 ; i < 9 ; i ++ ){
 //失败还得把按钮都变成不可点击
 if(mine->w_map[j][i] == 0 ){ //加上判断，避免失败后一些按钮会恢复
 MainButton[j][i]->setEnabled(false);
 }


 if( mine->map[j][i] == -1 ){
 if( ii == i && jj == j ){ //点击的位置标红
 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#ff0000;font-size:10px;font-weight:bold;");

 }

 mine->w_map[j][i] = -1;

 this->MainButton[j][i]->setDown(true);

 this->MainButton[j][i]->setText("💣");

 }

 }

 }





}





//初始化游戏

void MainWindow::newGame(){
```

```
qDebug()<<"重新游戏";

isStart = 0; //尚未点击开始按钮



//设置背景颜色

//this->setStyleSheet("background-color:#535552;");



mine = new Mine; //生成地雷

//Qt按钮设置：https://baijiahao.baidu.com/s?id=1691291899564946968&wfr=spider&for=pc



int size = 20;



//注意是[y][x]对应[j][i]



for(int j = 0 ; j < 9 ; j++ ){

for(int i = 0 ; i < 9 ; i++ ){
MainButton[j][i] = new QPushButton(" ",this);

//设置按钮颜色border:2px solid #808080

MainButton[j][i]->setStyleSheet("background-color:rgb(192,192,192);");

MainButton[j][i]->resize(size,size);

//MainButton[j][i]->setEnabled(false);

MainButton[j][i]->move(35+size*(i),70+size*(j)); //将按钮b1移动到点（50，50）的位置

// MainButton[j][i]->setParent("1",this);



MainButton[j][i]->setCheckable(true);

MainButton[j][i]->setAutoExclusive(true);



MainButton[j][i]->show();

//给connect传参数的方法：https://blog.csdn.net/qq_38188725/article/details/80686793

//解决鼠标左键无法响应鼠标事件，和connect无法响应

connect(MainButton[j][i], &QPushButton::clicked,this,
[=]{ButtonClick(i,j);

});




}

}


ui->pushButton->setText("☺");

ui->pushButton->resize(30,30);


//设置lcd位数

ui->lcdNumber->setDigitCount(4);

ui->lcdNumber_2->setDigitCount(3);

ui->lcdNumber_2->display("010");

ui->lcdNumber->display("000");
```

```
 //显示游戏时间
 myTimer = new QTimer(this);
 connect(myTimer,&QTimer::timeout, this, [=]{
 showtime();});



 //解决鼠标单击还是双击的
 ClickedState = 0; //初始化不点击
 timer = new QTimer(this);
 connect(timer, &QTimer::timeout, this,[ = ]{
 timerSingleClick();
 });
}

void MainWindow::mousePressEvent(QMouseEvent *event){
 timer->start(300);


 //鼠标事件参考链接: https://www.cnblogs.com/aiguona/p/9999507.html
 qDebug("鼠标点击");
 // 如果是鼠标左键按下
 if(event->button() == Qt::LeftButton)
 {
 qDebug("鼠标左键点击");


 for( int j = 0 ; j < 9 ; j ++ ){
 for( int i = 0 ; i < 9 ; i ++ ){
 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){
 qDebug("%d,%d",i,j);
 showButton(i,j,1);
 }
```

```
}
}
}
// 如果是鼠标右键按下
else if(event->button() == Qt::RightButton)
{
qDebug("鼠标右键点击");
timer->stop();
for( int j = 0 ; j < 9 ; j ++ ){
for( int i = 0 ; i < 9 ; i ++ ){
if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){
qDebug("%d,%d",i,j);
showButton(i,j,2);
}
```

```
 }

 }



 }

}

//鼠标双击事件

/*

void MainWindow::mouseDoubleClickEvent(QMouseEvent *event)

{

 //QString str="("+QString::number(e->x())+","+QString::number(e->y())+")";

 qDebug("鼠标双击");



}

*/

void MainWindow::mouseDoubleClickEvent(QMouseEvent *event){
 // 如果是鼠标左键按下（双击）

 if(event->button() == Qt::LeftButton){

 qDebug("鼠标左键双击");

 for( int j = 0 ; j < 9 ; j ++ ){

 for( int i = 0 ; i < 9 ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

 qDebug("%d,%d",i,j);

 chordingShow(i,j);

 showButton(i,j,3);

 }



 }

 }

 }

}
```

```
void MainWindow::leftShow(int x,int y){ //左键单击的显示

 int i = x,j = y;

 if( mine->map[j][i] == 0 ){

 MainButton[j][i]->setStyleSheet("background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText("");

 }else if( mine->map[j][i] == 1 ){

 //设置按钮颜色border:1px solid #999999;background-color:#c0c0c0

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 2 ){

 MainButton[j][i]->setStyleSheet("color:#008000;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else if( mine->map[j][i] == 3 ){
```

```
 MainButton[j][i]->setStyleSheet("color:#fe0202;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }else{

 MainButton[j][i]->setStyleSheet("color:#131386;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }

 this->MainButton[j][i]->setDown(true);



}

//按钮点击触发的事件

void MainWindow::showButton(int i, int j, int model){

 // if( isStart == 1 ){



 if( isStart == 0 ){ //如果还没未开始，则现在开始计时
 startTime();

 }



 this->MainButton[j][i]->setEnabled(false); //已点击过的按钮无法再点击

 //注意这里的i和j对应x,y

 qDebug("%d,%d",i,j);

 if( model == 1 ){ //左键点击

 //w_map为0说明没打开，没打开的才能点击

 if( mine->w_map[j][i] == 0 ){

 qDebug("1111");



 //this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 if( mine->map[j][i] == -1 ){ //踩到雷

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");
```

```
//此处应该插入游戏失败的代码
//this->MainButton[j][i]->setText("💣");
failShow(i,j); //失败+显示所有的雷
}else{
if( mine->map[j][i] == 0 ){ //如果点击0相当于双击
chordingShow(i,j);
}else{
leftShow(i,j);
}
}
this->MainButton[j][i]->setDown(true);
mine->left(i,j);


//设置按钮为按下状态：https://blog.csdn.net/qiangzi4646/article/details/80135120


}
```

```
// this->MainButton[j][i]->setText("");
}else if( model == 2 ){ //右键点击（标记）
//w_map为0说明没打开，没打开的才能点击
if( mine->w_map[j][i] == 0 ){
//qDebug("2222");
if( mine->lei > 0 ){
mine->right(i,j);
this->MainButton[j][i]->setText("🚩");
}
QString L = QString("%1").arg(mine->lei);
ui->lcdNumber_2->display(L);
}
// ui->textEdit->setText(mine->);
}else if( model == 3 ){
qDebug("3333");
```

```
 mine->chording(i,j); //双击

 this->MainButton[j][i]->setText(" ");

 }

 // }



}



void MainWindow::ButtonClick(int i,int j){

 ClickedState ++;

 retainx = i ,retainy = j;

 if(ClickedState == 1)

 timer->start(180); //430ms为自己设置



 if(ClickedState == 2)

 {

 ClickedState = 0; //初始化
 timer->stop(); //停止定时器继续计时

 //按钮双击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("我是双击！！！"),tr("确定"));

 showButton(i,j,3);

 }



}



//https://blog.csdn.net/wdmzslh/article/details/53301495

void MainWindow::timerSingleClick()

{

 ClickedState = 0; //初始化

 timer->stop(); //定时器停止计时



 //按钮单击响应代码放这里
 //QMessageBox::warning(this,tr("警告"),tr("单击"),tr("确定"));

 showButton(retainx,retainy,1);

}
```

Mine.cpp

```


//注意是map[y][x]

#include "Mine.h"

#include "ui_mainwindow.h"



//产生新的雷（map中-1表示雷）

void Mine::newMine() {

 lei = 10;

 success = 0;

 int tmp = 0; //随机生成10个雷

 srand((unsigned)time(NULL));

 while(1){
```

```
 int x = rand() % 10; //产生0到9的数字
 int y = rand() % 10;
 if (map[y][x] != -1) {
 map[y][x] = -1; //埋下地雷
 tmp++;
 }
 if( tmp >= 10 ) break;
 }



}


//根据雷补充map地图
void Mine::calcNum() { //计算雷的数量(实际用map[j][i]更佳，不容易混)
 //根据展开方格的数字去推断其相邻九宫格内未展开方格下面是否是地雷
```

```
//填入数字
for(int j = 0 ; j < 9 ; j ++ ){
for(int i = 0 ; i < 9 ; i++ ){
if( map[j][i] != -1 ){ //如果不是雷就计算周围的雷
int tmp = 0;
//周围一圈
if (i - 1 >= 0 && j + 1 < 9 && map[j + 1][i - 1] == -1)
tmp++;
if (j + 1 < 9 && map[j + 1][i] == -1) tmp++;
if (j + 1 < 9 && i + 1 < 9 && map[j + 1][i + 1] == -1) tmp++;


if (i - 1 >= 0 && map[j][i - 1] == -1) tmp++;
if (i + 1 < 9 && map[j][i + 1] == -1) tmp++;


if (i - 1 >= 0 && j - 1 >= 0 && map[j - 1][i - 1] == -1)
tmp++;
if (j - 1 >= 0 && map[j - 1][i] == -1) tmp++;
```

```
 if (i + 1 < 9 && j - 1 >= 0 && map[j - 1][i + 1] == -1)
tmp++;

 map[j][i] = tmp;

 }

 }

 }

}



void Mine::left(int x, int y) {



 //☺😐☹

 //左键用于打开安全的格子，推进游戏进度；

 if (map[y][x] == -1) {

 //信息框（https://www.cnblogs.com/zhoug2020/p/10094855.html，注
意加上头文件）



 QMessageBox::information(NULL, "游戏结束", "很遗憾，你已经踩到雷
了，游戏结束！");
 //cout << "你已经踩雷，游戏失败" << endl;
 }
 else {
 if (map[y][x] == 0) {
 w_map[y][x] = -3; //-3表示为空
 }
 else {
 w_map[y][x] = map[y][x];
 }
 }


}


void Mine::right(int x, int y) {
 if( lei > 0 ){ //如果标记够用的话
 lei--; //标记雷的数量减1

 //右键用于标记地雷，以辅助判断，或为接下来的双击做准备；(-2表示标记)

 w_map[y][x] = -2;

 if( map[y][x] == -1 ){ //如果找到的是雷

 success++;

 }



 }else{

 if( success == 10 ){ //成功找到所有的雷

 //此处应该贴胜利

 QMessageBox::information(NULL, "游戏结束", "恭喜你，游戏胜利！");

 }



 }

}

/*
```

```
void Mine::chording(int x, int y) {

 //双击只能双击没有数字的

 // 双击在一个数字周围的地雷标记完时， 相当于对数字周围未打开的方块均进行一次左键单击操作：

 //标记的雷数一定要等于X，否则双击是打不开的。

 //把周围没标记的都打开（8个）

 if (x + 1 < 9 && w_map[y][x + 1] != -2) left(x + 1, y);

 if (x + 1 < 9 && y - 1 >= 0 && w_map[y-1][x + 1] != -2) left(x + 1, y-1);

 if (x + 1 < 9 && y + 1 < 9 && w_map[y+1][x + 1] != -2) left(x + 1, y+1);




 if ( y - 1 >= 0 && w_map[y - 1][x] != -2) left(x, y - 1);

 if ( y + 1 < 9 && w_map[y + 1][x] != -2) left(x, y + 1);



 if (x - 1 >= 0 && w_map[y][x - 1] != -2) left(x + 1, y);
 if (x - 1 >= 0 && y - 1 >= 0 && w_map[y - 1][x - 1] != -2)
left(x - 1, y - 1);

 if (x - 1 >= 0 && y + 1 < 9 && w_map[y + 1][x - 1] != -2)
left(x - 1, y + 1);

}

*/

Mine::Mine(){

 newMine();

 calcNum();

 /*

 for( int j = 0 ; j < 9 ; j ++ ){

 for( int i = 0 ; i < 9 ; i ++ ){

 qDebug()<<map[j][i]<<" ";

 }



 }*/

}
```

```
Mine::~Mine(){

 for(int j = 0 ; j < 9 ; j++ ){

 for(int i = 0 ; i < 9 ; i++ ){

 map[j][i] = 0;

 }

 }

 lei = 10;

 success = 0;



}
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

 <class>MainWindow</class>
```

```
<widget class="QMainWindow" name="MainWindow">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>279</width>

<height>299</height>

</rect>

</property>

<property name="windowTitle">

<string>扫雷 ——by： orall</string>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="pushButton">

<property name="geometry">

<rect>
```

```
<x>110</x>

<y>10</y>

<width>31</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<family>Microsoft YaHei UI Light</family>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="text">

<string>开始游戏</string>

</property>
```

```
</widget>
<widget class="QLCDNumber" name="lcdNumber">
<property name="geometry">
<rect>
<x>150</x>
<y>10</y>
<width>64</width>
<height>31</height>
</rect>
</property>
<property name="frameShape">
<enum>QFrame::Box</enum>
</property>
<property name="frameShadow">
<enum>QFrame::Raised</enum>
</property>
```

```
<property name="lineWidth">
<number>1</number>
</property>
<property name="midLineWidth">
<number>0</number>
</property>
</widget>
<widget class="QLCDNumber" name="lcdNumber_2">
<property name="geometry">
<rect>
<x>30</x>
<y>10</y>
<width>61</width>
<height>31</height>
</rect>
</property>
```

```
<property name="font">
<font>
<weight>50</weight>
<bold>false</bold>
</font>
</property>
<property name="frameShape">
<enum>QFrame::Box</enum>
</property>
<property name="frameShadow">
<enum>QFrame::Raised</enum>
</property>
<property name="lineWidth">
<number>1</number>
</property>
<property name="digitCount">
```

```
<number>5</number>
</property>
<property name="intValue" stdset="0">
<number>10</number>
</property>
</widget>
</widget>
<widget class="QMenuBar" name="menubar">
<property name="geometry">
<rect>
<x>0</x>
<y>0</y>
<width>279</width>
<height>26</height>
</rect>
</property>
```

```
   </widget>

   <widget class="QStatusBar" name="statusbar"/>

   </widget>

   <resources/>

   <connections/>

  </ui>
```

3.扫雷Bug版本（2021.10.11）
该版本实现了：
1.扫雷基本功能
待补充：
1.胜利条件
2.双击的显示一大片
Bug:
1.地图和按钮不对应 【新版已解决：在MainButton的move这里】
2.点击了空白的按钮会出现重新弹回的情况
3.失败后重新点击笑脸没有重新开始 【新版已解决：】
4.点击后显示慢 【新版已解决：单击双击的时间判断影响了】
5.点到会出现假雷和假旗 【新版已解决：已删除，每次的new必须对应删除，否则】
6.只有9个雷 【新版已解决：mine生成雷函数那里】
7.失败后按钮会重新恢复
8.右键点击会恢复

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H


#include <QMainWindow>
#include <QPainter>
#include <QPushButton>
#include <QLabel>
#include <QMouseEvent>
#include <QTimer> // 定时器对象
#include "Mine.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE
```

```
class MainWindow : public QMainWindow
{
 Q_OBJECT

public:
 QLabel *MainLabel[9][9];
 QPushButton *MainButton[9][9];
 Mine *mine; //地雷
 int retainx,retainy; //鼠标点击的位置
 int isStart; //是否开始游戏

 MainWindow(QWidget *parent = nullptr);
 ~MainWindow();

 void showtime(); //更新时钟
 void showButton(int i,int j,int model);

 void startTime(); //开始计时

 void mousePressEvent(QMouseEvent *event);

 void mouseDoubleClickEvent(QMouseEvent *event);

 void newGame(); //初始化游戏

 void failShow(int ii,int jj); //失败的时候显示所有的雷

 void ButtonClick(int i,int j);

 void timerSingleClick(); //定时器事件，处理单击任务

 int* reverseij(int i,int j); //保存当前坐标

private slots:

 void on_pushButton_clicked();

private:

 Ui::MainWindow *ui;





 QTimer *timer; //判断双击还是单击

 int ClickedState; //点击状态，常态不点击为0，单击对应1，双击对应2



 QTimer *myTimer;// 定时器对象

};



#endif // MAINWINDOW_H
```

Mine.h

```


#ifndef MINE_H

#define MINE_H



#include<iostream>

#include<time.h>

#include<stdlib.h>

#include <QMessageBox>


```

```
class Mine{ //扫雷的类
public:
 int map[9][9]; //初级（初级为9*9个方块10个雷） //雷表示-1
 int w_map[9][9]; //显示出来的地图
 int lei;

 Mine();
 ~Mine();

 void newMine(); //产生新的雷
 void calcNum(); //计算雷的数量
 void left(int x, int y); //左击
 void right(int x, int y); //右击
 void chording(int x, int y); //双击
};
```

```
#endif // MINE_H
```

main.cpp

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
 QApplication a(argc, argv);
 MainWindow w;
 w.show();
 return a.exec();
}
```

mainwindow.cpp

```
#include "mainwindow.h"

#include "ui_mainwindow.h"



MainWindow::MainWindow(QWidget *parent)

 : QMainWindow(parent)

 , ui(new Ui::MainWindow)

{


 // ui->lcdNumber->setStyleSheet("background-color:#000000;color:#ff0000;");

 // ui->lcdNumber_2->setStyleSheet("background-color:#000000;color:#ff0000;");


 newGame(); //初始化游戏
 //解决鼠标单击还是双击的

 ClickedState = 0; //初始化不点击

 timer = new QTimer(this);

 connect(timer, &QTimer::timeout, this,[ = ]{

 timerSingleClick();

 });

}



MainWindow::~MainWindow()

{

 delete ui;

}



//显示游戏时间

void MainWindow::showtime(){
 // ui->textEdit_2->setText("str");
 static int seconds=0;
 seconds++;

 QString str;
 //前位补0: https://www.cnblogs.com/zzzsj/p/14566157.html
 str = QString("%01").arg(seconds, 3, 10, QLatin1Char('0'));
 ui->lcdNumber->display(str);
 //ui->textEdit_2->setText(str);
}

//开始计时
void MainWindow::startTime(){
 myTimer->start(1000);
 //启动定时器
 //时间间隔为100ms
 //每隔100ms，定时器myTimer自动触发timeout()

 //如果定时器没有激活，才启动,防止多次点击start按钮出现错误

 //if (myTimer->isActive() == false)

 //{

 isStart = 1; //标记为开始游戏

 //}

}




//点击笑脸按钮

void MainWindow::on_pushButton_clicked()

{

 myTimer->stop();

 newGame();

}
```

```
//扫雷失败会把所有的雷位置展示出来

void MainWindow::failShow(int ii,int jj) //参数为点击的位置（标红）

{

 for(int i = 0 ; i < 9 ; i ++ ){

 for(int j = 0 ; j < 9 ; j ++ ){

 //失败还得把按钮都变成不可点击

 MainButton[j][i]->setEnabled(false);

 if( mine->map[j][i] == -1 ){

 if( ii == i && jj == j ){ //标红

 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:#ff0000;font-size:10px;font-weight:bold;");

 }

 this->MainButton[j][i]->setDown(true);

 this->MainButton[j][i]->setText("💣");
 }

 }

 }




}






//初始化游戏

void MainWindow::newGame(){

 isStart = 0; //尚未点击开始按钮



 //设置背景颜色

 //this->setStyleSheet("background-color:#535552;");



 mine = new Mine; //生成地雷

 //Qt按钮设置：https://baijiahao.baidu.com/s?id=1691291899564946968&wfr=spider&for=pc

 ui->setupUi(this);

 int size = 20;



 //注意是[y][x]对应[j][i]



 for(int j = 0 ; j < 9 ; j++ ){

 for(int i = 0 ; i < 9 ; i++ ){

 MainButton[j][i] = new QPushButton(" ",this);



 //设置按钮颜色border:2px solid #808080

 MainButton[j][i]->setStyleSheet("background-color:rgb(192,192,192);");

 MainButton[j][i]->resize(size,size);

 //MainButton[j][i]->setEnabled(false);

 MainButton[j][i]->move(80+size*(j),80+size*(i)); //将按钮b1移动到点（50，50）的位置
 // MainButton[j][i]->setParent("1",this);



 MainButton[j][i]->setCheckable(true);

 MainButton[j][i]->setAutoExclusive(true);



 MainButton[j][i]->show();

 //给connect传参数的方法:
https://blog.csdn.net/qq_38188725/article/details/80686793

 //解决鼠标左键无法响应鼠标事件，和connect无法响应

 connect(MainButton[j][i], &QPushButton::clicked,this,

 [=]{ButtonClick(i,j);

 });





 }

 }
```

```
    ui->pushButton->setText("☺");

    ui->pushButton->resize(30,30);



    //设置lcd位数

    ui->lcdNumber->setDigitCount(4);

    ui->lcdNumber_2->setDigitCount(3);

    ui->lcdNumber_2->display("010");

    ui->lcdNumber->display("000");



    //显示游戏时间

    myTimer = new QTimer(this);

    connect(myTimer,&QTimer::timeout, this, [=]{

    showtime();});



}
```

```
void MainWindow::mousePressEvent(QMouseEvent *event){


 timer->start(300);


 //鼠标事件参考链接: https://www.cnblogs.com/aiguona/p/9999507.html

 qDebug("鼠标点击");

 // 如果是鼠标左键按下

 if(event->button() == Qt::LeftButton)

 {

 qDebug("鼠标左键点击");



 for( int j = 0 ; j < 9 ; j ++ ){

 for( int i = 0 ; i < 9 ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){
```

```
qDebug("%d,%d",i,j);

showButton(i,j,1);

}



}

}

}

// 如果是鼠标右键按下

else if(event->button() == Qt::RightButton)

{

qDebug("鼠标右键点击");

timer->stop();

for( int j = 0 ; j < 9 ; j ++ ){

for( int i = 0 ; i < 9 ; i ++ ){

if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

qDebug("%d,%d",i,j);
```

```
 showButton(i,j,2);
 }


 }
 }


 }
}
//鼠标双击事件
/*
void MainWindow::mouseDoubleClickEvent(QMouseEvent *event)
{
 //QString str="("+QString::number(e->x())+","+QString::number(e->y())+")";
 qDebug("鼠标双击");


```

```
}
*/
void MainWindow::mouseDoubleClickEvent(QMouseEvent *event){
 // 如果是鼠标左键按下（双击）
 if(event->button() == Qt::LeftButton){
 qDebug("鼠标左键双击");
 for( int j = 0 ; j < 9 ; j ++ ){
 for( int i = 0 ; i < 9 ; i ++ ){
 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){
 qDebug("%d,%d",i,j);
 showButton(i,j,3);
 }

 }
 }
 }
```

```
}



//按钮点击触发的事件

void MainWindow::showButton(int i, int j, int model){

 // if( isStart == 1 ){



 if( isStart == 0 ){ //如果还没未开始，则现在开始计时

 startTime();

 }

 /* if( isStart == -1 ){ //如果游戏失败，应该点击笑脸重来

 return;

 }*/



 this->MainButton[j][i]->setEnabled(false); //已点击过的按钮无法再点击

 //注意这里的i和j对应x,y

```

```
//qDebug("%d,%d",i,j);

if( model == 1 ){ //左键点击

//w_map为0说明没打开，没打开的才能点击

if( mine->w_map[j][i] == 0 ){

//qDebug("1111");



//this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

if( mine->map[j][i] == -1 ){ //踩到雷

MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

//此处应该插入游戏失败的代码

//this->MainButton[j][i]->setText("💣");

isStart = -1; //表示为游戏失败

ui->pushButton->setText("☹");

myTimer->stop(); //停止计时

failShow(i,j); //显示所有的雷
```

```
 }else if( mine->map[j][i] == 0 ){
 MainButton[j][i]->setStyleSheet("background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");
 this->MainButton[j][i]->setText("0");
 }else if( mine->map[j][i] == 1 ){
 //设置按钮颜色border:1px solid #999999;background-color:#c0c0c0
 MainButton[j][i]->setStyleSheet("color:#1212f9;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");
 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));
 }else if( mine->map[j][i] == 2 ){
 MainButton[j][i]->setStyleSheet("color:#008000;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");
 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));
 }else if( mine->map[j][i] == 3 ){
 MainButton[j][i]->setStyleSheet("color:#fe0202;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");
 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));
 }else{

 MainButton[j][i]->setStyleSheet("color:#131386;background-color:rgb(192,192,192);font-size:10px;font-weight:bold;");

 this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

 }

 this->MainButton[j][i]->setDown(true);

 mine->left(i,j);



 //设置按钮为按下状态:https://blog.csdn.net/qiangzi4646/article/details/80135120





 }



 // this->MainButton[j][i]->setText("");
```

```
}else if( model == 2 ){ //右键点击（标记）
//w_map为0说明没打开，没打开的才能点击
if( mine->w_map[j][i] == 0 ){
QString L = QString("%1").arg(mine->lei);
ui->lcdNumber_2->display(L);
//qDebug("2222");
if( mine->lei > 0 ){
mine->right(i,j);
this->MainButton[j][i]->setText("🚩");
}
}
// ui->textEdit->setText(mine->);
}else if( model == 3 ){
qDebug("3333");
mine->chording(i,j); //双击
this->MainButton[j][i]->setText(" ");
```

```
 }

 // }



}



void MainWindow::ButtonClick(int i,int j){

 ClickedState ++;

 retainx = i ,retainy = j;

 if(ClickedState == 1)

 timer->start(430); //430ms为自己设置



 if(ClickedState == 2)

 {

 ClickedState = 0; //初始化

 timer->stop(); //停止定时器继续计时

 //按钮双击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("我是双击！！！"),tr("确定"));

 showButton(i,j,3);

 }



}



//https://blog.csdn.net/wdmzslh/article/details/53301495

void MainWindow::timerSingleClick()

{

 ClickedState = 0; //初始化

 timer->stop(); //定时器停止计时



 //按钮单击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("单击"),tr("确定"));

 showButton(retainx,retainy,1);

}
```

Mine.cpp

```
//注意是map[y][x]
#include "Mine.h"
#include "ui_mainwindow.h"

//产生新的雷（map中-1表示雷）
void Mine::newMine() {
 lei = 10;

 srand((unsigned)time(NULL));
 for (int i = 0; i < 10; i++) { //随机产生10个雷
 int x = rand() % 10; //产生0到9的数字
 int y = rand() % 10;
 if (map[y][x] != -1) {
 map[y][x] = -1; //埋下地雷
 }

 else { //如果有雷则重新开始

 i--; //重新开始

 }

 }

}



//根据雷补充map地图

void Mine::calcNum() { //计算雷的数量(实际用map[j][i]更佳，不容易混)

 //根据展开方格的数字去推断其相邻九宫格内未展开方格下面是否是地雷



 //填入数字

 for(int j = 0 ; j < 9 ; j ++ ){

 for(int i = 0 ; i < 9 ; i++ ){

 if( map[j][i] != -1 ){ //如果不是雷就计算周围的雷

 int tmp = 0;
 //周围一圈

 if (i - 1 >= 0 && j + 1 < 9 && map[j + 1][i - 1] == -1)
tmp++;

 if (j + 1 < 9 && map[j + 1][i] == -1) tmp++;

 if (j + 1 < 9 && i + 1 < 9 && map[j + 1][i + 1] == -1) tmp++;



 if (i - 1 >= 0 && map[j][i - 1] == -1) tmp++;

 if (i + 1 < 9 && map[j][i + 1] == -1) tmp++;



 if (i - 1 >= 0 && j - 1 >= 0 && map[j - 1][i - 1] == -1)
tmp++;

 if (j - 1 >= 0 && map[j - 1][i] == -1) tmp++;

 if (i + 1 < 9 && j - 1 >= 0 && map[j - 1][i + 1] == -1)
tmp++;

 map[j][i] = tmp;

 }

 }
 }

}



void Mine::left(int x, int y) {


 //☺☹☹

 //左键用于打开安全的格子，推进游戏进度；

 if (map[y][x] == -1) {

 //信息框（https://www.cnblogs.com/zhoug2020/p/10094855.html，注意加上头文件）



 QMessageBox::information(NULL, "游戏结束", "很遗憾，你已经踩到雷了，游戏结束！");

 //cout << "你已经踩雷，游戏失败" << endl;

 }

 else {

 if (map[y][x] == 0) {
```

```
 w_map[y][x] = -3; //-3表示为空

 }

 else {

 w_map[y][x] = map[y][x];

 }

 }



}



void Mine::right(int x, int y) {

 if( lei > 0 ){ //如果标记够用的话

 lei--; //标记雷的数量减1

 //右键用于标记地雷，以辅助判断，或为接下来的双击做准备；(-2表示标记)

 w_map[y][x] = -2;

 }

}
```

```
void Mine::chording(int x, int y) {

 // 双击在一个数字周围的地雷标记完时， 相当于对数字周围未打开的方块均进行一次左键单击操作：

 //标记的雷数一定要等于X，否则双击是打不开的。

 //把周围没标记的都打开（8个）

 if (x + 1 < 9 && w_map[y][x + 1] != -2) left(x + 1, y);

 if (x + 1 < 9 && y - 1 >= 0 && w_map[y-1][x + 1] != -2) left(x + 1, y-1);

 if (x + 1 < 9 && y + 1 < 9 && w_map[y+1][x + 1] != -2) left(x + 1, y+1);




 if ( y - 1 >= 0 && w_map[y - 1][x] != -2) left(x, y - 1);

 if ( y + 1 < 9 && w_map[y + 1][x] != -2) left(x, y + 1);


 if (x - 1 >= 0 && w_map[y][x - 1] != -2) left(x + 1, y);

 if (x - 1 >= 0 && y - 1 >= 0 && w_map[y - 1][x - 1] != -2)
left(x - 1, y - 1);

 if (x - 1 >= 0 && y + 1 < 9 && w_map[y + 1][x - 1] != -2)
left(x - 1, y + 1);

}





Mine::Mine(){

 newMine();

 calcNum();

 for( int j = 0 ; j < 9 ; j ++ ){

 for( int i = 0 ; i < 9 ; i ++ ){

 qDebug()<<map[j][i]<<" ";

 }



 }

}
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

 <class>MainWindow</class>

 <widget class="QMainWindow" name="MainWindow">

 <property name="geometry">

 <rect>

 <x>0</x>

 <y>0</y>

 <width>366</width>

 <height>343</height>

 </rect>

 </property>

 <property name="windowTitle">

 <string>扫雷 ——by： orall</string>

 </property>
```

```
<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="pushButton">

<property name="geometry">

<rect>

<x>140</x>

<y>10</y>

<width>71</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<family>Microsoft YaHei UI Light</family>

<weight>75</weight>

<bold>true</bold>

</font>
```

```
</property>

<property name="text">

<string>开始游戏</string>

</property>

</widget>

<widget class="QLCDNumber" name="lcdNumber">

<property name="geometry">

<rect>

<x>230</x>

<y>10</y>

<width>64</width>

<height>31</height>

</rect>

</property>

<property name="frameShape">

<enum>QFrame::Box</enum>
```

```
</property>
<property name="frameShadow">
<enum>QFrame::Raised</enum>
</property>
<property name="lineWidth">
<number>1</number>
</property>
<property name="midLineWidth">
<number>0</number>
</property>
</widget>
<widget class="QLCDNumber" name="lcdNumber_2">
<property name="geometry">
<rect>
<x>50</x>
<y>10</y>
```

```
<width>61</width>

<height>31</height>

</rect>

</property>

<property name="font">

<font>

<weight>50</weight>

<bold>false</bold>

</font>

</property>

<property name="frameShape">

<enum>QFrame::Box</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Raised</enum>

</property>
```

```
<property name="lineWidth">

<number>1</number>

</property>

<property name="digitCount">

<number>5</number>

</property>

<property name="intValue" stdset="0">

<number>10</number>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>
```

```
 <width>366</width>

 <height>26</height>

 </rect>

 </property>

 </widget>

 <widget class="QStatusBar" name="statusbar"/>

 </widget>

 <resources/>

 <connections/>

</ui>
```

2.扫雷GUI基本版本(2021.10.7)

mainwindow.h

```
#ifndef MAINWINDOW_H

#define MAINWINDOW_H



#include <QMainWindow>
```

```
#include <QPainter>
#include <QPushButton>
#include <QLabel>
#include <QMouseEvent>
#include <QTimer> // 定时器对象
#include "Mine.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
 Q_OBJECT

public:
```

```
QLabel *MainLabel[9][9];
QPushButton *MainButton[9][9];
Mine *mine; //地雷
int retainx,retainy; //鼠标点击的位置

MainWindow(QWidget *parent = nullptr);
~MainWindow();

void showtime(); //更新时钟
void showButton(int i,int j,int model);
void mousePressEvent(QMouseEvent *event);
void mouseDoubleClickEvent(QMouseEvent *event);

void ButtonClick(int i,int j);
void timerSingleClick(); //定时器事件，处理单击任务
int* reverseij(int i,int j); //保存当前坐标
```

```
private slots:

 void on_pushButton_clicked();

private:

 Ui::MainWindow *ui;




 QTimer *timer; //判断双击还是单击

 int ClickedState; //点击状态，常态不点击为0，单击对应1，双击对应2


 QTimer *myTimer;// 定时器对象

};



#endif // MAINWINDOW_H
```

Mine.h

```


#ifndef MINE_H
```

```
#define MINE_H
#include<iostream>
#include<time.h>
#include<stdlib.h>
#include <QMessageBox>

class Mine{ //扫雷的类
public:
 int map[9][9]; //初级（初级为9*9个方块10个雷） //雷表示-1
 int w_map[9][9]; //显示出来的地图

 Mine();
 ~Mine();

 void newMine(); //产生新的雷
 void calcNum(); //计算雷的数量
 void left(int x, int y); //左击

 void right(int x, int y); //右击

 void chording(int x, int y); //双击

};



#endif // MINE_H
```

main.cpp

```
#include "mainwindow.h"

#include <QApplication>



int main(int argc, char *argv[])

{

 QApplication a(argc, argv);

 MainWindow w;
 w.show();
 return a.exec();
}
```

mainwindow.cpp

```

#include "mainwindow.h"
#include "ui_mainwindow.h"



MainWindow::MainWindow(QWidget *parent)
 : QMainWindow(parent)
 , ui(new Ui::MainWindow)
{

 mine = new Mine; //生成地雷
 //Qt按钮设置：https://baijiahao.baidu.com/s?
```

```
id=1691291899564946968&wfr=spider&for=pc

 ui->setupUi(this);

 int size = 30;



 //注意是[y][x]对应[j][i],这里就直接[y][x]对应[i][j]了

 for(int i = 0 ; i < 9 ; i++ ){

 for(int j = 0 ; j < 9 ; j++ ){



 MainButton[i][j] = new QPushButton("*",this);

 MainButton[i][j]->resize(size,size);

 //MainButton[i][j]->setEnabled(false);

 MainButton[i][j]->move(150+size*(i),150+size*(j)); //将按钮b1
移动到点（50，50）的位置

 // MainButton[i][j]->setParent("1",this);

 MainButton[i][j]->show();

 //给connect传参数的方法:
https://blog.csdn.net/qq_38188725/article/details/80686793
 //解决鼠标左键无法响应鼠标事件，和connect无法响应

 connect(MainButton[i][j], &QPushButton::clicked,this,

 [=]{ButtonClick(j,i);

 });




 }

 }






 //解决鼠标单击还是双击的


 ClickedState = 0; //初始化不点击

 timer = new QTimer(this);
```

```
connect(timer, &QTimer::timeout, this,[ = ]{

timerSingleClick();

});




myTimer = new QTimer(this);


//connect(myTimer,SIGNAL(timeout()), this, SLOT(this->showtime()));




//QPainter painter(this);

//QPixmap pixmap("C:\\Users\\czh\\Desktop\\399370.jpg");

// painter.drawPixmap(45,45,128,128,pixmap);
```

```
 //myTimer->start(100);

}

MainWindow::~MainWindow()
{
 delete ui;
}

void MainWindow::showtime(){
 ui->textEdit_2->setText("str");
 static int minutes=0, seconds=0;
 seconds++;
 if( seconds >= 60 ){
```

```
 seconds = 0;

 minutes++;

 }

 QString str;

 str = QString("%1:%2").arg(minutes).arg(seconds);



 ui->textEdit_2->setText("str");



}









void MainWindow::on_pushButton_clicked()
```

```
{
 myTimer->start(100);
 //启动定时器
 //时间间隔为100ms
 //每隔100ms，定时器myTimer自动触发timeout()
 //如果定时器没有激活，才启动,防止多次点击start按钮出现错误
 if (myTimer->isActive() == false)
 {

 }
}


void MainWindow::mousePressEvent(QMouseEvent *event){
 timer->start(300);

 //鼠标事件参考链接：
```

```
https://www.cnblogs.com/aiguona/p/9999507.html

 qDebug("鼠标点击");

 // 如果是鼠标左键按下

 if(event->button() == Qt::LeftButton)

 {

 qDebug("鼠标左键点击");



 for( int j = 0 ; j < 9 ; j ++ ){

 for( int i = 0 ; i < 9 ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){

 qDebug("%d,%d",i,j);

 showButton(i,j,1);

 }



 }

 }
```

```
}
// 如果是鼠标右键按下
else if(event->button() == Qt::RightButton)
{
qDebug("鼠标右键点击");
timer->stop();
for( int j = 0 ; j < 9 ; j ++ ){
for( int i = 0 ; i < 9 ; i ++ ){
if(this->MainButton[j][i]->geometry().contains(this->mapFromGlobal(QCursor::pos()))){
qDebug("%d,%d",i,j);
showButton(i,j,2);
}

}
}
```

```
 }
}
//鼠标双击事件
/*
void MainWindow::mouseDoubleClickEvent(QMouseEvent *event)
{
 //QString str="("+QString::number(e->x())+","+QString::number(e->y())+")";
 qDebug("鼠标双击");


}
*/
void MainWindow::mouseDoubleClickEvent(QMouseEvent *event){
 // 如果是鼠标左键按下（双击）
 if(event->button() == Qt::LeftButton){
 qDebug("鼠标左键双击");
 for( int j = 0 ; j < 9 ; j ++ ){

 for( int i = 0 ; i < 9 ; i ++ ){

 if(this->MainButton[j][i]->geometry().contains(this-
>mapFromGlobal(QCursor::pos()))){

 qDebug("%d,%d",i,j);

 showButton(i,j,3);

 }



 }

 }

 }

}



void MainWindow::showButton(int i, int j, int model){

 //注意这里的i和j对应x,y

 //☠▧

```

```
qDebug("%d,%d",i,j);

if( model == 1 ){ //左键点击

qDebug("1111");

mine->left(i,j);

this->MainButton[j][i]->setText(QString::number(mine->map[j][i]));

// this->MainButton[j][i]->setText("");

}else if( model == 2 ){ //右键点击

qDebug("2222");

mine->right(i,j);

this->MainButton[j][i]->setText("۞");

}else if( model == 3 ){

qDebug("3333");

mine->chording(i,j); //双击

this->MainButton[j][i]->setText("۞");

}
```

```
}


void MainWindow::ButtonClick(int i,int j){
 ClickedState ++;
 retainx = i ,retainy = j;
 if(ClickedState == 1)
 timer->start(430); //430ms为自己设置

 if(ClickedState == 2)
 {
 ClickedState = 0; //初始化
 timer->stop(); //停止定时器继续计时
 //按钮双击响应代码放这里
 //QMessageBox::warning(this,tr("警告"),tr("我是双击！！！"),tr("确定"));
 showButton(i,j,3);
 }



}



//https://blog.csdn.net/wdmzslh/article/details/53301495

void MainWindow::timerSingleClick()

{

 ClickedState = 0; //初始化

 timer->stop(); //定时器停止计时



 //按钮单击响应代码放这里

 //QMessageBox::warning(this,tr("警告"),tr("单击"),tr("确定"));

 showButton(retainx,retainy,1);

}
```

Mine.cpp

```cpp
//注意是map[y][x]

#include "Mine.h"

#include "ui_mainwindow.h"




void Mine::newMine() { //产生新的雷

 srand((unsigned)time(NULL));

 for (int i = 0; i < 10; i++) { //随机产生10个雷

 int x = rand() % 10; //产生0到9的数字

 int y = rand() % 10;

 if (map[y][x] != -1) {

 map[y][x] = -1; //埋下地雷

 }

 else { //如果有雷则重新开始

 i--; //重新开始

 }
 }

}



void Mine::calcNum() { //计算雷的数量(实际用map[j][i]更佳，不容易混)

 //根据展开方格的数字去推断其相邻九宫格内未展开方格下面是否是地雷

 //填入数字

 for (int i = 0; i < 9; i++) {

 for (int j = 0; j < 9; j++) {

 if (map[i][j] != -1) { //不是地雷的话就计算周围一圈

 int tmp = 0;

 //周围一圈

 if (j - 1 >= 0 && i + 1 < 9 && map[i + 1][j - 1] == -1) tmp++;

 if (i + 1 < 9 && map[i + 1][j] == -1) tmp++;

 if (i + 1 < 9 && map[i + 1][j + 1] == -1) tmp++;
 if (j - 1 >= 0 && map[i][j - 1] == -1) tmp++;

 if (j + 1 < 9 && map[i][j + 1] == -1) tmp++;



 //周围一圈

 if (j - 1 >= 0 && i - 1 >= 0 && map[i - 1][j - 1] == -1) tmp++;

 if (map[i - 1][j] == -1) tmp++;

 if (j + 1 < 9 && map[i - 1][j + 1] == -1) tmp++;

 map[i][j] = tmp;



 }



 }

 }




}
```

```
void Mine::left(int x, int y) {
 //左键用于打开安全的格子，推进游戏进度;
 if (map[y][x] == -1) {
 //信息框（https://www.cnblogs.com/zhoug2020/p/10094855.html，注意加上头文件）
 QMessageBox::information(NULL, "游戏结束", "很遗憾，你已经踩到雷了，游戏结束！");
 //cout << "你已经踩雷，游戏失败" << endl;
 }
 else {
 if (map[y][x] == 0) {
 w_map[y][x] = -3; //-3表示为空
 }
 else {
 w_map[y][x] = map[y][x];
 }
 }



}



void Mine::right(int x, int y) {

 //右键用于标记地雷，以辅助判断，或为接下来的双击做准备；(-2表示标记)

 w_map[y][x] = -2;

}



void Mine::chording(int x, int y) {

 // 双击在一个数字周围的地雷标记完时， 相当于对数字周围未打开的方块均进行一次左键单击操作：

 //标记的雷数一定要等于X，否则双击是打不开的。

 //把周围没标记的都打开（8个）

 if (x + 1 < 9 && w_map[y][x + 1] != -2) left(x + 1, y);

 if (x + 1 < 9 && y - 1 >= 0 && w_map[y-1][x + 1] != -2) left(x + 1, y-1);
 if (x + 1 < 9 && y + 1 < 9 && w_map[y+1][x + 1] != -2) left(x
+ 1, y+1);





 if ( y - 1 >= 0 && w_map[y - 1][x] != -2) left(x, y - 1);

 if ( y + 1 < 9 && w_map[y + 1][x] != -2) left(x, y + 1);



 if (x - 1 >= 0 && w_map[y][x - 1] != -2) left(x + 1, y);

 if (x - 1 >= 0 && y - 1 >= 0 && w_map[y - 1][x - 1] != -2)
left(x - 1, y - 1);

 if (x - 1 >= 0 && y + 1 < 9 && w_map[y + 1][x - 1] != -2)
left(x - 1, y + 1);

}





Mine::Mine(){

 newMine();
 calcNum();
}
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
 <property name="geometry">
 <rect>
 <x>0</x>
 <y>0</y>
 <width>581</width>
 <height>501</height>
 </rect>
 </property>
```

```
<property name="windowTitle">
<string>扫雷 ——by: orall</string>
</property>
<property name="layoutDirection">
<enum>Qt::LeftToRight</enum>
</property>
<widget class="QWidget" name="centralwidget">
<widget class="QTextEdit" name="textEdit">
<property name="enabled">
<bool>false</bool>
</property>
<property name="geometry">
<rect>
<x>100</x>
<y>90</y>
<width>81</width>
```

```
 <height>31</height>

 </rect>

 </property>

 <property name="layoutDirection">

 <enum>Qt::RightToLeft</enum>

 </property>

 <property name="html">

 <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot; &quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;

&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot; /&gt;&lt;style type=&quot;text/css&quot;&gt;

p, li { white-space: pre-wrap; }

&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'SimSun'; font-size:9pt; font-weight:400; font-style:normal;&quot;&gt;

&lt;p style=&quot; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;&quot;&gt;10&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;
</string>
```

```xml
</property>

</widget>

<widget class="QTextEdit" name="textEdit_2">

<property name="enabled">

<bool>false</bool>

</property>

<property name="geometry">

<rect>

<x>370</x>

<y>90</y>

<width>91</width>

<height>31</height>

</rect>

</property>

<property name="focusPolicy">

<enum>Qt::StrongFocus</enum>
```

```
 </property>

 <property name="layoutDirection">

 <enum>Qt::RightToLeft</enum>

 </property>

 <property name="html">

 <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML
4.0//EN&quot; &quot;http://www.w3.org/TR/REC-
html40/strict.dtd&quot;&gt;

&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot;
content=&quot;1&quot; /&gt;&lt;style
type=&quot;text/css&quot;&gt;

p, li { white-space: pre-wrap; }

&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-
family:'SimSun'; font-size:9pt; font-weight:400; font-
style:normal;&quot;&gt;

&lt;p style=&quot; margin-top:0px; margin-bottom:0px; margin-
left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;00:00&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;
</string>

 </property>
```

```
</widget>
<widget class="QPushButton" name="pushButton">
<property name="geometry">
<rect>
<x>200</x>
<y>30</y>
<width>181</width>
<height>41</height>
</rect>
</property>
<property name="text">
<string>开始游戏</string>
</property>
</widget>
</widget>
<widget class="QMenuBar" name="menubar">
```

```xml
<property name="geometry">
<rect>
<x>0</x>
<y>0</y>
<width>581</width>
<height>26</height>
</rect>
</property>
<widget class="QMenu" name="menu">
<property name="title">
<string>扫雷</string>
</property>
<addaction name="actionStart"/>
</widget>
<widget class="QMenu" name="menu_2">
<property name="title">
```

```
 <string>帮助</string>
 </property>
 </widget>
 <addaction name="menu"/>
 <addaction name="menu_2"/>
 </widget>
 <widget class="QStatusBar" name="statusbar"/>
 <action name="actionStart">
 <property name="text">
 <string>Start</string>
 </property>
 </action>
 </widget>
 <resources/>
 <connections/>
</ui>
```

1.生成9*9个按钮（2021.9.24）

定义代码，声明在头文件里面

```
//Qt按钮设置：https://baijiahao.baidu.com/s?id=1691291899564946968&wfr=spider&for=pc

ui->setupUi(this);

int size = 30;

//注意是[y][x]对应[j][i]

for(int i = 0 ; i < 9 ; i++ ){

for(int j = 0 ; j < 9 ; j++ ){

MainButton[i][j] = new QPushButton(this);

MainButton[i][j]->resize(size,size);

MainButton[i][j]->move(150+size*(i),150+size*(j)); //将按钮b1移动到点（50，50）的位置

// MainButton[i][j]->setParent("1",this);

MainButton[i][j]->show();
}

}
```