**Linux大作业**

**------------------------------------------------------------171491322 谷贺**

**连连看游戏代码：**

**Mainwindow.cpp**

#include <resolv.h>

#include <stdlib.h>

#include <netinet/in.h>

 #include <arpa/inet.h>

 #include <unistd.h>

 #include <QScrollBar>

 #include "mainwindow.h"

 #include "ui\_mainwindow.h"

#include"udp.h"

#include<QImage>

 #include <QTime>

#include "llk.h"

 #include<iostream>

 using namespace std;

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

 {     ui->setupUi(this);     connect(this, SIGNAL(recvMessage(QString,QString)),this,SLOT(onRecvMessage(QString,QString)));

  //this->setCentralWidget(link);

  //link->setMaximumSize(300,300);

   ui->editChatRec->setReadOnly(true);

// link = new llk();

   size=6;

isLinkE=false;

sendMessage（“192.168.3.255”，8838，“o192.168.3.117”，true）；}

MainWindow：：~MainWindow（）{delete ui;//删除用户界面}

void MainWindow:：changeEvent（QEvent\*e）{

QMainWindow:：changeEvent(e)；

switch（e->type（））{

case QEvent:：LanguageChange:ui->retranslateUi（this）；

break；

default: //默认值

break；}

}

void MainWindow:：on\_btnSend\_clicked（）{

QString string=ui->lineEdit->text（）；

if（ui->radiosingle->isChecked（）==true）

 {   sendMessage(string,8838,'c' + ui->editChat->text(),true);     }

   else     {

 string = "192.168.3.255";

        sendMessage("192.168.3.255",8838,'c' + ui->editChat->text(),true);     }

Qstring STR = trutf8 ("own pair") + string + trutf8 ("send") + UI - > editchat - > text();

ui->editChatRec->setPlainText(ui->editChatRec->toPlainText()+" \n" +str);

     ui->editChat->clear();

   QScrollBar \*bar = ui->editChatRec->verticalScrollBar();

     bar->setSliderPosition(bar->maximum()); }

Ui::MainWindow\* MainWindow::getUi() { return ui; }

void MainWindow::onRecvMessage(QString msg, QString ipAdd) {

//      QString time = QTime::currentTime().toString();

Qstring string = time + "- n" + qstring (trutf8 ("received from")) + ipadd + trutf8 (": message of) +" \ n "+ msg.right ( msg.length ()-1);

switch( msg.at (0).toAscii())

 { Case 'o': / / 如果接受的是上线消息

ui->listWidget->addItem( msg.right ( msg.length ()-1));

 break;

Default: / / 如果接受的是普通消息

ui->editChatRec->setPlainText(ui->editChatRec->toPlainText() +"\n" + string);

       break;

 }

QScrollBar \*bar = ui->editChatRec->verticalScrollBar();

 bar->setSliderPosition(bar->maximum());  }

void MainWindow::on\_ action\_ begin\_ triggered() {

   /\* link=new llk();

  link->setsize(6);

 link->arrayInit(6);

 link->Create(6);

 link->show();

  isLinkE = true;\*/

//size=4；

if（isLinkE）{

//link=new llk（）；

//link->setsize（8）；

//link->arrayInit（8）；

//link->Create（8）；

link->close（）；

islinkE=false；

}

else{

link=new llk（）；

link->setsize（size）；

link->arrayInit（size）；

link->Create（size）；

link->show（）；

islinkE=true；}

if（isLink==false）{

link=new llk（）；

link->setsize（size）；

link->arrayInit（size）；

link->Create（size）；

link->show（）；

isLink=true；

}

}

void MainWindow:：on\_action\_close\_trigged（）{

this->close（）；

link->close（）；

}

void MainWindow:：on\_action\_pause\_trigged（）{

if（IsLinkE）

link->setEnabled（false）；}

void MainWindow:：on\_action\_return\_trigged（）{

link->setEnabled（true）；}

void MainWindow:：on\_action\_easy\_trigged（）{

size=6；

if（isLinkE）{

//link=new llk（）；

//link->setsize（8）；

//link->数组索引（8）；

//link->Create（8）；

link->close（）；

isLinkE=false；}

else{

link=new llk（）；

link->setsize（6）；

link->arrayInit（6）

；link->Create（6）；

link->show（）；

islinkE=true；}  
if（isLinkE==false）{

link=new llk（）；

link->setsize（6）；

link->arrayInit（6）；

link->Create（6）；

link->show（）；

islinkE=true；}}

void MainWindow:：on\_action\_soso\_trigged（）{

size=8；

/\*llk\*lin；

lin=new llk（）

//this->setCentralWidget（lin）；

//lin->setGeometry（30,30500500）；lin->setsize（8）；lin->arrayInit（8）;

lin->Create（8）；lin->setVisible（true）；lin->close（）；\*/if（isLinkE）{

//link=new llk（）；// link->setsize（8）；//link->arrayInit（8）；// link->Create（8）；link->close（）；islinkE=false；}

else{

link=new llk（）；

link->setsize（8）；link->arrayInit（8）；link->Create（8）；link->show（）；isLinkE=true；}

if（isLink==false）{

link=new llk（）；link->setsize（8）；link->arrayInit（8）；link->Create（8）；link->show（）；islinkE=true；}}

void MainWindow:：on\_action\_diffcult\_trigged（）{

size=10；/\*llk\*lin；lin=new llk（）

//this->setCentralWidget（lin）；//lin->setGeometry（30,30500500）；

lin->size（8）；lin->arrayInit（8）  
lin->Create（8）；lin->setVisible（true）；lin->close（）；\*/if（isLink）{

//link=new llk（）；//link->setsize（8）；//link->arrayInit（8）；//link->Create（8）；link->close（）；islinkE=false；}

else{

link=new llk（）；link->setsize（10）；link->arrayInit（10）；link->Create（10）；link->show（）；islinkE=true；}

if（isLinkE==false）{

link=new llk（）；link->setsize（10）；link->arrayInit（10）；link->Cre

**mybutton.cpp**

#include“mybutton.h”

MyButton:：MyButton（int value，QWidget\*parent）{

this->value=value；this->setParent（parent）

QObject:：connect（this，SIGNAL（clicked（）），this，SLOT（mySlot（））；}

void MyButton:：mySlot（）{

emit myClicked（this->value）

}

**Main.cpp**

#include<QtGui/QApplication>

#include“mainwindow.h

#include<QtGui/QApplication>

#include<errno.h>

#include<sys/socket.h>

#include<resolv.h>

#include<stdlib.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<iostream>

#include<string>

Using namespace std;

const unsigned int MAXBUF=1024;

void\*thd\_处理程序（void\*window）{

MainWindow \*w=（MainWindow \*）window;

Int sockfd;

Struct sockaddr\_in my\_addr；

socklen\_t addrlen；

char buffer[MAXBUF]；

unsigned int myport=8838

printf（“服务器已启动！\n“）

//为udp创建套接字

if（（sockfd=socket（AF\_INET，SOCK\_DGRAM，0））<0）

//注意attention

{perror（“套接字创建失败”）；exit（errno）；

}

//initialize sockaddr\_in bzero（&myaddr，sizeof（my\_addr））；

my\_addr.sin\_family=my\_addr.sin\_port=htons（myport）；my\_addr.sin\_addr.s\_addr=INADDR\_ANY

//bind

if（bind（sockfd，（struct sockaddr\*）&my\_addr，sizeof（struct sockaddr））==-1{

perror（“绑定失败”）；退出（errno）；}

printf（“绑定成功\n”）

addrlen=sizeof（struct sockaddr）；while（true）{

bzero（buffer，MAXBUF）

int len=recvfrom（sockfd，buffer，MAXBUF，0，（sockaddr\*）&my\_ addr，&addrlen）；

if（len<0）{

perror（“recvfrom”）；exit（errno）；}

emit w->recvMessage（QObject:：trUtf8（buffer）,inet\_ntoa（my\_addr：sin\_addr));      }

//close connection

close（sockfd）；}

int main（int argc，char\*argv[]）{

pthread\_t ntid； QApplication a（argc，argv）；MainWindow w;

pthread\_create（&ntid，NULL，thd\_handler，&w）；w.show（）；return a.exec（）；}

**udp.cpp**

#include“udp.h”

#include<QString>

#include<errno.h>

#include <sys/socket.h>

#include<resolv.h>

#include<stdlib.h>

#include<netinet/in.h>

#include <arpa/inet.h>

#include <unistd.h>

void sendMessage(QString ipAddress,unsigned int port,QString msg,bool isBroad) { 

 // const unsigned int MAXBUF = 1024;     int sockfd;     int addr\_ len;     int len;

struct sockaddr\_ in dest;     //char buffer[MAXBUF];

char addr[16] ;

 strcpy(addr, ipAddress.toAscii ());

//create a socket for udp

 if((sockfd = socket(AF\_ INET,SOCK\_ DGRAM,0)) < 0) //attention     {

        perror("socket creation failure");

 exit(errno);     }

   if(isBroad)     {

        int broad=1;

 setsockopt(sockfd,SOL\_ SOCKET,SO\_ BROADCAST,&broad,sizeof(broad));

strcpy(addr,"192.168.3.255");      }

    //initialize sockaddr\_ in

  bzero(&dest,sizeof(dest));

dest.sin\_ family = AF\_ INET;      dest.sin\_ port = htons( port);

    if(inet\_ aton(addr,(struct in\_ addr\*)& dest.sin\_ addr)==0)     {

        perror("ip address format is error");         exit(errno);     }

    //send msg to server

 addr\_ len = sizeof(dest);

// bzero(buffer,MAXBUF);

    len = sendto(sockfd,  msg.toLocal8Bit (),  msg.toLocal8Bit ().length(), 0                  ,(sockaddr\*)&dest, addr\_ len);

 if (len < 0)     {

        printf("\n\rsend error.\n\r");         exit(errno);     }

    //close connection     ::close(sockfd);  }

**llk.cpp**

#include "llk.h"

 #include<iostream>

#include"judge.h"

 llk::llk()

{     isFirstClick = true;     llk::createBtnRerank();

    size=6;

llk::createCurrentProgressBar(size);

    timer = new QTimer();

 currentTime=0;

    QObject::connect(timer,SIGNAL(timeout()),this,SLOT(timeProcess()));  }

void llk::xxx(int x) {

     clear(x);

   // std::cout<<x<<std::endl;

 // mybutton[x]->setVisible(false);

  //  Widget::arrayInit(6);

 // progressBar[1]->setValue(20); }

void llk::addPic(QString path,int i) {

    labelPic[i] = new QLabel(this);

    labelPic[i]->setGeometry(i\*100+5,30,100,120);

 QPixmap pixmap(path);

labelPic[i]->setPixmap（path）；labelPic[i]->setVisible（false）；progressBar[i]=new QProgressBar（）

progressBar[i]->setGeometry（i\*100+10120100,20）；

progressBar[i]->setParent（this）；progressBar[i]->setValue（10）；}

void llk:：Create（int size）{

createTimeProcessBar（size\*size）；

timer->start（1000）；

char ss[26][20]={“begin”，“icon/icon1.png”，“icon/icon2.png”，“icon/icon3.png”，“icon/icon4.png”，“icon/icon5.png”  
“/icon/icon6.png”  
“/icon/icon7.png”，“../icon/icon8.png”，”../icon/icon9.png“，”../icon/icon10.png“

“/icon/icon11.gif”、“.icon/icon12.gif”、“/icon/icon13.gif”、“.icon/icon14.gif”、“/icon/icon15.gif”、“/icon/icon16.gif”、“/icon/icon17.gif”、“/icon/icon19.gif”、“/icon/icon19.gif”、“.icon/icon20.gif”、“/icon/icon21.gif”、“.icon/icon22.gif”，“/icon/icon23.gif”，“../icon/icon24.gif”，“../icon/icon25.gif”}

createCurrentProgressBar（size）；

currentIndex=size\*size;

     progressBarCurrent->setValue(0);

   for(unsigned int i=0;i<size\*size;i++)     {

        mybutton[i] = new MyButton(i,this);

        mybutton[i]->setGeometry(20+i%size\*60,150+i/size\*60,60,60);

        int x,y;

   x=i/size+1;

        if((i+1)%size)         {

            y=(i+1)%size;         }

    else

 y=size;

 int m=array[x][y];

       // std::cout<<x<<","<<y<<std::endl;

       mybutton[i]->setIcon(QIcon(ss[m]));

     mybutton[i]->setIconSize(QSize(50,50));

        QObject::connect(mybutton[i],SIGNAL(myClicked(int)),this,SLOT(xxx(int )));         //mybutton[i]->setIcon(QIcon(s));      } }

void llk::arrayInit(int size) {

    array[size+1][0]=0;

    array[size+1][size+1]=0;

array[0][size+1]=0;

   array[0][0]=0;

   for(int i=1;i<=size;i++)     {

        array[0][i]=0;

        array[size+1][i]=0;

     array[i][0]=0;

  array[i][size+1]=0;

        for(int j=1;j<=size;j++)         {

            array[i][j]=((i-1)\*size+j-1)/4+1;

         }      }

   for(int i=1;i<=size;i++)     {

        for(int j=1;j<=size;j++)         {

            int x1,x2,y1,y2,tmp1;

           // srand(unsigned(time(NULL)));

          x1=rand()%size+1;

  x2=rand()%size+1;

     y1=rand()%size+1;

      y2=rand()%size+1;

             tmp1=array[x1][y1];

           array[x1][y1]=array[x2][y2];

     array[x2][y2]=tmp1;

x1=firstIndex/size+1;

      x2=index/size+1;

        if((firstIndex+1)%size)         {

          y1=(firstIndex+1)%size;         }

      else

   y1=size;

   if((index+1)%size)         {

            y2=(index+1)%size;         }

     else

 y2=size;

        std::cout<<x1<<","<<y1<<std::endl;

        std::cout<<x2<<","<<y2<<std::endl;

     std::cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<std::endl;

        if(judge(x1,y1,x2,y2,array,size))         {

            mybutton[firstIndex]->setVisible(false);

         // free(mybutton[firstIndex]);

      //mybutton[firstIndex]=NULL;

      mybutton[index]->setVisible(false);

            array[x1][y1]=0;

   array[x2][y2]=0;

  currentIndex-=2;

            progressBarCurrent->setValue(size\*size-currentIndex);

        }

        isFirstClick=true;          } }

void llk::setsize(int size1) {

    if(size1==8||size1==6||size1==10||size1==4)     {

        size=size1;     } }

void llk::mybuttonDelt() {

    for(int i=0;i<size\*size;i++)     {

        free(mybutton[i]);     } }

void llk::createBtnRerank()  {

     btnReRank = new QPushButton();

     btnReRank->setParent(this);

   btnReRank->setText("reRank");

}

void llk::createTimeProcessBar(int sumtime) {

    progressBarTime = new QProgressBar();

    progressBarTime->setGeometry(20,80,60\*size,30);

   progressBarTime->setParent(this);

  progressBarTime->setMaximum(sumtime);    }

void llk::timeProcess() {

    if(currentTime>=size\*size)

     this->close();

 currentTime++;

    progressBarTime->setValue(currentTime); }

Judge.cpp  #include"judge.h" #include<iostream>

int judge(int x1,int y1,int x2,int y2,int array[16][16],int size) {

        if(x1==x2&&y1==y2)       {

            return 0;       }

        if((array[x1][y1]==array[x2][y2]))

        {

if(judge1(x1,y1,x2,y2,array,size)||judge2(x1,y1,x2,y2,array,size)||judge3(x1,y1,x2,y2,array,size))

                {

                        array[x1][y1]=0;

                  array[x2][y2]=0;

                       return 1;

      }

    }

        return 0;

 }

int judge1(int x1,int y1,int x2,int y2,int array[16][16],int size) {

        //if(array[x1][y1]==array[x2][y2])

   //{

  int tmp;

        if(x1!=x2&&y1!=y2)

         return 0;

    if(y1>y2)         {

                tmp=y1;

         y1=y2;

      y2=tmp;         }

     if(x1==x2)         {

                if(y1>y2)

{

tmp=y1；y1=y2；y2=tmp；}

对于（int i=y1+1；i<y2；i++）{

if（array[x1][i]）{

return0；}}}if（x1>x2）{

tmp=x1；x1=x2；x2=tmp；}if（y1==y2）{

if（x1>x2）{

tmp=x1；x1=x2；x2=tmp；}

if（int j=x1+1；j<x2；j++）{

if（array[j][y1]）

{

 return 0;

  }

      }

    }

 int judge2(int x1,int y1,int x2,int y2,int array[16][16],int size) {

        int flag=0;

if(judge1(x1,y2,x1,y1,array,size)&&judge1(x1,y2,x2,y2,array,size)&&array[x1][y2]==0)                 {

                        //flag++;

   //std::cout<<x1<<","<<i<<std::endl;

   return 1;                 }

if(judge1(x2,y1,x1,y1,array,size)&&judge1(x2,y1,x2,y2,array,size)&&array[x2][y1]==0)                 {

                        //flag++;

  //std::cout<<j<<","<<y1<<std::endl;

          return 1;                 }

        return 0; }

int judge3(int x1,int y1,int x2,int y2,int array[16][16],int size) {

        for(int i=0;i<=size+1;i++)         {

if(judge2(x1,i,x2,y2,array,size)&&judge1(x1,i,x1,y1,array,size)&&array[x1][i]==0)                 { //std::cout<<x1<<","<<i<<std::endl;

   return 1;                 }         }

        for(int j=0;j<=size+1;j++)         {

if(judge2(j,y1,x2,y2,array,size)&&judge1(j,y1,x1,y1,array,size)&&array[j][y1]==0)                 {

  //  std::cout<<j<<","<<y1<<std::endl;

   return 1;                 }

        }

 return 0; }

**Header file source code:**

**Judge. H**

#ifndef JUDGE\_ H

#define JUDGE\_ H

int judge(int x1,int y1,int x2,int y2,int array[16][16],int size);

 int judge1(int x1,int y1,int x2,int y2,int array[16][16],int size);

int judge2(int x1,int y1,int x2,int y2,int array[16][16],int size);

int judge3(int x1,int y1,int x2,int y2,int array[16][16],int size);

#endif // JUDGE\_ H

Llk.h

 #ifndef LLK\_ H

#define LLK\_ H

#include<mybutton.h>

 #include <QWidget>

 #include<QPicture>

#include <QWidget>

 #include<QObject>

 #include<QProgressBar>

#include<QLabel>

 #include<QTimer>

class llk : public QWidget {

    Q\_ OBJECT public:

  llk();

    void setsize(int size1);

    void addPic(QString path,int i);

  void Create(int size);

void clear(int index);

  void mybuttonDelt();

  void createBtnRerank();

 void newGame();

    void createCurrentProgressBar(int size);

  void createTimeProcessBar(int sumtime);

  int currentTime;

  private:

    MyButton \* mybutton[150];

  QLabel \* labelPic[4];

    QProgressBar  \* progressBar[4];

  int array[16][16];

 bool isFirstClick;

   int firstIndex;

    QPushButton \* btnReRank;

  QProgressBar \*progressBarCurrent;

  QProgressBar \*progressBarTime;

   int currentIndex;

    QTimer \*timer;

     int size;

public slots:

  void xxx(int x);

 void arrayInit(int size);

    void Rerank();

  void timeProcess();  };

#endif // LLK\_ H

**Mainwindow.h**

#ifndef JUDGE\_ H

 #define JUDGE\_ H

int judge(int x1,int y1,int x2,int y2,int array[16][16],int size);

int judge1(int x1,int y1,int x2,int y2,int array[16][16],int size);

 int judge2(int x1,int y1,int x2,int y2,int array[16][16],int size);

 int judge3(int x1,int y1,int x2,int y2,int array[16][16],int size);

#endif // JUDGE\_ H

**Mybutton.h**

#ifndef MYBUTTON\_ H

#define MYBUTTON\_ H

#include <QPushButton>

class MyButton : public QPushButton {

    Q\_ OBJECT public:

    MyButton(int value,QWidget \* parent = 0);

 private:

 int value;  signals:

    void myClicked(int value);

public slots:

 void mySlot(); };

#endif // MYBUTTON\_ H

**Udp.h**

#ifndef UDP\_ H

#define UDP\_ H

#include<QString>

void sendMessage(QString ipAddress,unsigned int port,QString msg,bool isBroad);

#endif // UDP\_H