任务5.2：销子 功能

2023年7月7日 星期五

14:24

**开始游戏 功能**

地图展示完全后，接下来的工作就是接受用户的鼠标输入，进行销子判断功能，下面将详细介绍。

**（2）销子功能实现**

销子功能的主要流程：

首先，从CGameDlg界面上通过鼠标选取2个图片，确定两个图片的所在的坐标(用V1，V2表示);

然后，通过调用CGameControl的Link()函数去判断两个图片是否相邻；

最后，该Link()函数通过调用CGameLogic的IsLink(int\*\* pGameMap, Vertex V1, Vertex V2)来判断两个点是否相等，而通过Clear(int\*\* pGameMap, Vertex V1, Vertex V2)来去掉两个点，完成具体销子操作。

这里还是采用自底向上的设计方法，首先完成CGameLogic的类的实现，然后完成CGameControl的类的实现，最后去完善CGameDlg类的实现。

（2.1）类CGameLogic的设计

为了记录，两个图片间的相连路径问题，引入stack<Vertex> verList来保存，而类CGameLogic主要完成以下5大功能：判断是否相连，消除相连的图片，判断地图是否为空和图片相连路径的获取和清空操作。

类CGameLogic的具体定义如下：

计算机生成了可选文字:
LISIngnamespacestd-
[ElclassCGameLogic
public
1]It**GameMap《
1]ItrOWS
1]Itcols
IntpicNL皿．
private.
//地信息
//地行
//地列
//地包含的片的个数
stack<Vert.ex>verList．
p巛b112
三==一仁三一涩过
IntGetElement(IntnROW，Intncol)
、℃IdInitMap(IntnRows，intnCols,IntnPicNum)-
增加的属性
相当于对Gam巳Map进行初始化
b001IsLink(VertexVI,VertexV2)
、℃IdClear(VertexVI，VertexV2)
、℃IdClearVerList0
stack<Vertex>GetVerList0
isB1ank()，
b001
private.
b001LinkInRowVertexVI,VertexV2
b001LinklnCol(VertexVI,VertexV2)
增加的方法
b001LinkOneCorner（VertexVI,
b001LinkTwoCorner（VertexVI,
b001LinkY(IntROWI,
IntRow2，
b001Linki（intROW，
IntC011，
VertexV2)
为实现ISLink而增加的子函数
VertexV2)
IntCol)
IntC012），

（2.1.1）判断两个图片相连

首先，判断两个图片是否相连，有以下四种情况需要讨论：

（1）V1和V2在同一行上相连

V1和V2在同一行上则两个点row坐标一致，相当于判断V1(row,col1)和V2(row,col2)两个点是否相连。

首先，设计一个函数 LinkX() 判断在Row上，从Col1到Col2是否相连。

计算机生成了可选文字:
//爿断在R“行上，从c。11到c。12是否才目遴
《三b001CGameLogic：Linki（intROW,
IntC011，
If0
IntC012）
三
If（C011>C012）
两个列上选择c。11为较小的列数
Inttemp=C011
C011=C012，
C012
temp：
for(inti=C011+讼i<=C012
三盲一：二是#
C012）returntrue《
If(GameMap(Row)]！=BLANK)
false
returnfalse

然后，设计函数LinkInRow()，判断点V1和V2是否相连。该函数通过设置具体的row和col值后，调用LinkX()实现

计算机生成了可选文字:
//爿断VI和V2在同一个R“上是否遴
《三b001CGameLogic：LinklnRow（VertexVI
1]ItRowVI.工ow，
VertexV2〕
IntC011
。c01．
1]ItC012=V2.c01．
returnLinki（Row，
C011，
C012），

（2）V1和V2在同一列上相连

V1和V2在同一列上则两个点col坐标一致，相当于判断V1(row1,col)和V2(row2,col)两个点是否相连。和相面的解决思路一致，首先，设计一个函数 LinkY() 判断在Col上，从Row1到Row2是否相连。

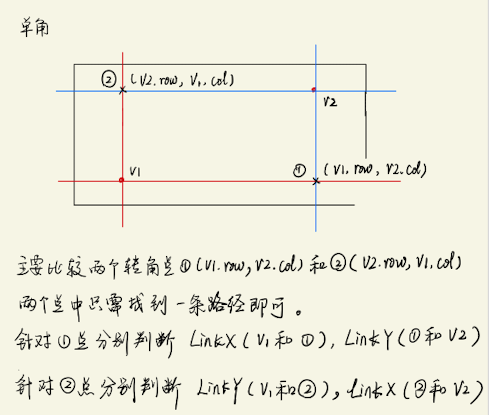
计算机生成了可选文字:
//爿断在c。1列上，从R“1到R“2是否遴
《到b001CGameLogic：LinkY（intROWI，intRow2
If0
Col)
int
三
If(Rowl>Row2)
两个行上选择R“1为较小的行数
1]Ittemp=ROWI
Rowl=Row2
Row2=temp：
for(inti=Rowl+讼i<=Row2
三盲一=二是#
Row2)returntrue
If(GameMapCi)(Col]！=BLANK)
false
returnfalse

然后，设计函数LinkInCol()，判断点V1和V2是否相连。该函数通过设置具体的row和col值后，调用LinkY()实现

计算机生成了可选文字:
//爿断VI和V2在同一个c。1上是否才目遴
b001CGameLogic．:LinklnCol(VertexVI,
思路和LinkInRow0完全一致
1]ItRowl=VI.
IntRow2=V2.row-
1]ItC01=VI.c01，
returnLinkY(Rowl,Row2，Col)
VertexV2)

（3）V1和V2通过一个拐角相连

V1和V2通过一个拐角相连时，有两种情况：下拐角(V1.row,V2.col)和上拐角(V2.row,V1.col)。两个拐角找到一条路径即可。

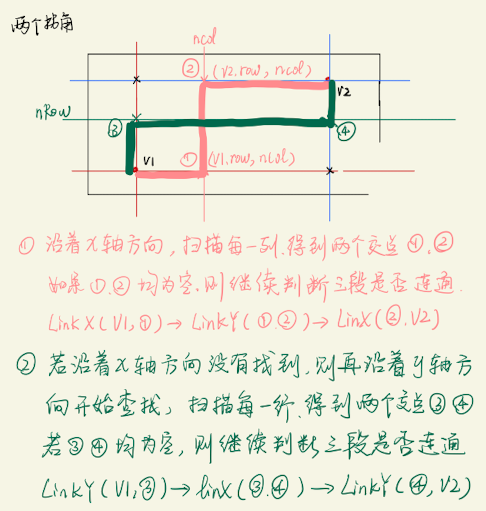


设计函数LinkOneCorner()来判断V1和V2通过一个拐角相连

计算机生成了可选文字:
//爿断VI和V2在经过一个拐斧遴
《三b001CGameLogic．：LinkOneCorner(VertexVI,VertexV2)
If(GameMap0过row)02co到==BLJdiK)
If(LinkX(VI.row，
·201，
verL1st.ush〔的
If(GameMap02．row)01co到
If(LinkX02row，
c01，
VertexV=[
·rOW，
verL1st.push〔的
re．
returnfalse
·20D&&LinkY(VI.row，
col，BLÆiK}
V2.1℃w，
厍于保存途径的拐点，
厍于后《会制路径
BLANK)
c01）&&LinkY(VI.row，
col，BLÆiK}
V2.1℃w，
V2.cal))
。cal))

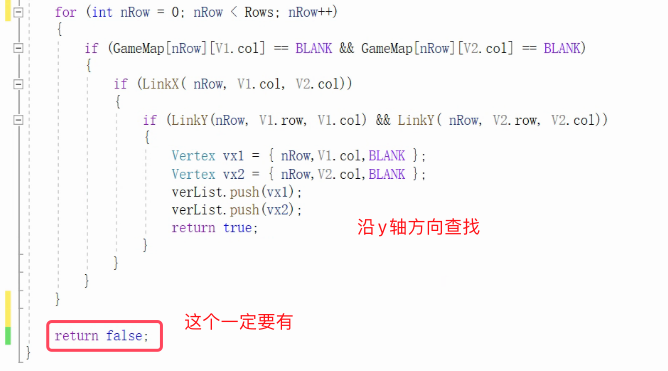
（4）V1和V2通过两个拐角相连

V1和V2通过两个拐角相连时，也有两种情况：沿着x轴方向找到某一列nCol，或者沿着y轴方向找到莫一行nRow，得到两个拐点v3，v4



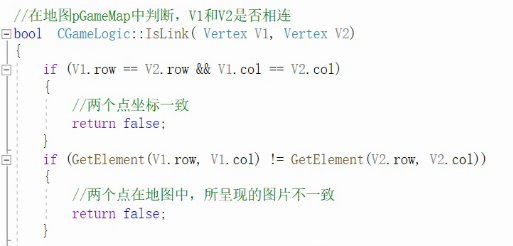
设计函数LinkTwoCorner()来判断v1和v2通过两个拐点相连

计算机生成了可选文字:
//爿断VI和V2在经过两个拐斧才目遴
到b001CGameLogic．：LinkTwoCorner（VertexVIVertexV2)
for(intnC01=0《nC01〈C01s；nC01+刁
If(GameMapCVI.row)CnC01]==BLJdiK&&GameMap021℃w]CnC01)
BLANK)
If(LinkY（VI.row，V2.row,
nC01))
if(LinkX（VI.row，VI.201，
ncol)&&Linkx（V2
·rOW，
Vertexvxl={VI
nC01，BLANK}
{V2
nC01，BLANK}
Vertexvx2
verList.push(vxl)
verList.push(vx2)
沿x轴方向查找
V220L
ncol)）



然后，对IsLink()函数的设计

IsLink函数判断v1和v2是否相连，并且保存相连的路径，具体的设计如下：





计算机生成了可选文字:
If01
If
V2cal){
201
(LinkInC01(VI，(2)）
在一列内相连
verList.push02）
elseif(VI.c01
/边界情况，弟0列，则增加外边框的点作为路径
verL1st.push(lVI
·rOW
verL1st.push（[V2.rowV2
verL1st.push02）
elseif(VI.c01
Cols
/边界情况，弟c。Is一1列，0堰能0过哐扪点乍00过
verL1st.push(lVI
c01+1，0}）
·rOW，
verL1st.push(IV2
2col+1，0}）
·rOW，
verL1st.push02）

计算机生成了可选文字:
If(LinkOneCorner（VI
verL1st.push02）
If(LinkTwoCorner（VI
verL1st.push02）
returnfalse
一个拐角相连
两个拐角相连

（2.1.2）两个相连图片的消除

设计clear函数来去掉v1和v2两个图片，其主要方法就是将v1和v2设置为空

计算机生成了可选文字:
//在扌卓，VI和V2两个片
三、℃IdCGameLogic．:Clear(VertexVI,VertexV2)
Game-MapCVI.工ow〕01co到=BLANK
GameMap02．工ow〕02．co到=BLANK

（2.1.3）判断地图是否为空

遍历地图中的所有元素，判断是否全部为空

计算机生成了可选文字:
//爿断所有的片是否已经全部除
三b001CGameLogic：isB1ank0
三
for(inti
i〈Rows，i++)
for(intj
j〈C01s，」++）
If(this->GameMapCi)0]！=BANK)
returnfalse

（2.1.4）获取和清空相连的图片路径

主要对属性verList进行操作，获取操作是深拷贝一个备份出来，而清空操作则是通过swap方法来达到快速清空的操作。

计算机生成了可选文字:
//將堆栈rLtä5E,Qcopy%copyList
[E]stack<Vertex>CGameLogic．GetVerList0
stack〈Vertex)(verL1st)
stack<Vertex>copyList
returncopyList．
//清窒堆栈操作，区为c艹本身投有供清窒操作，因此编写函数完成清窒操作
、℃IdCGameLogic．ClearVerList0
利用swap操作速度比p。p快2/3
stack<Vertex>0。swap(verL1st)

（2.2）类CGameControl的设计

类CCameControl的设计包含几个部分：选择点的设置，判断两个连接点是否连接和游戏是否胜利，具体定义如下所示:

计算机生成了可选文字:
—classCGameControl
/选择的弟一个点
selFirst
Vertex
/选择的弟二个点
selSecond，
Vertex
//买觐具体的逻辑
CGameLogicmGameLogic，
public
CGameContr01()，//带参数的构造函数
、℃IdSetFirstPoint(IntPROW，Intncol)
、℃IdSetSecondPoint(intnRowintnCOl)
IntGetElement(IntnROW，
Intncol)
b001StartGame(Introws,
b001Link(stack<Vertex>&verList)
b001i到巧且0
增加选择点的设置
增加属性的设置方氵去
//获她中的据
Intcols,IntpicNum)//开始渫戏，供CGamlg类调拜
//爿断selFirst和selSecond是否相遴，路径放在verList中
//爿断渫戏是否结束

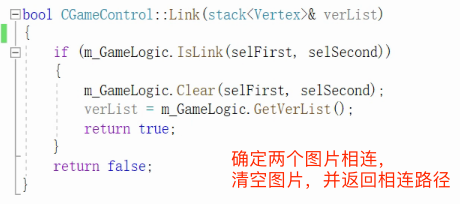
（2.2.1）选择点的设置

首先，定义两个属性，来表示选择的两个顶点 First和Second，并且给出两个选择点的设置方法。

计算机生成了可选文字:
IdCGameControl．
selF1rst.row
selF1rst.201
IdCGameControl．
selSecond.row
selSecond.c01
SetFirstPoint(IntnROW,
nRow
nCol
SetSecondPoint(intnRow
nRow
nCol
Intncol)
Intncol)

（2.2.2）判断两个连接点是否相连

通过设计Link函数来判断两个连接点selFirst和selSecond是否相连，返回相连状态，并且返回相连的路径列表。



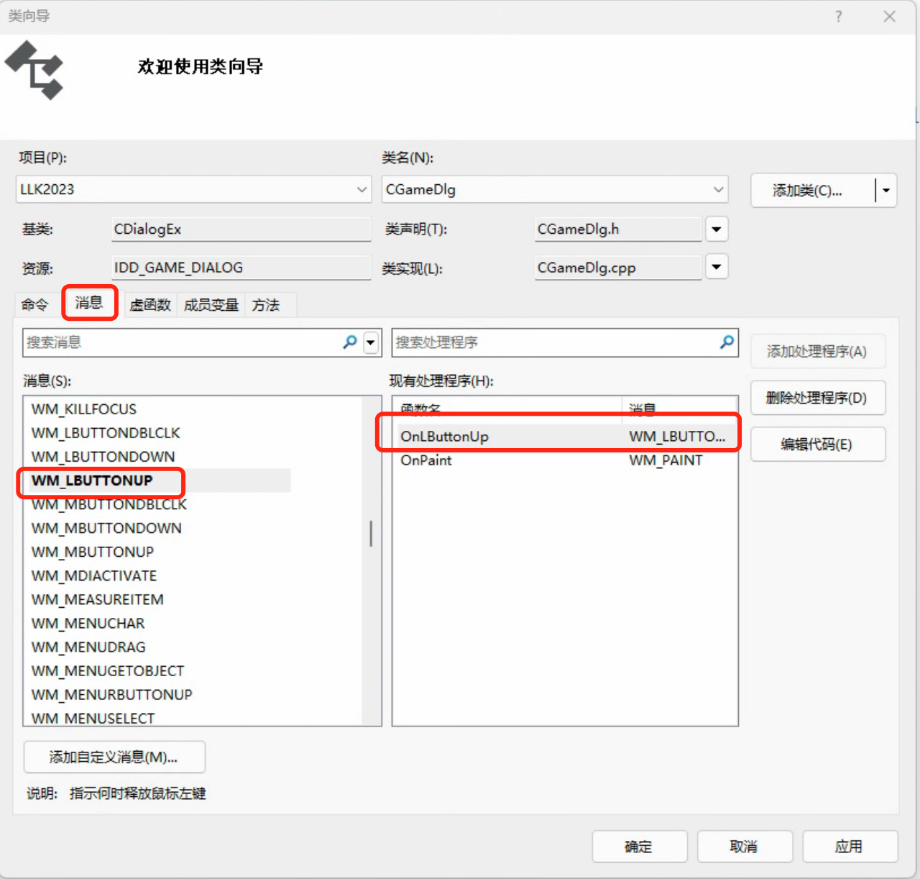
（2.2.3）判断游戏是否胜利

游戏胜利的标志是地图为空，因此可以通过调用CGameLogic中的IsBlank来实现该功能

计算机生成了可选文字:
到b001CGameControl：isWin0
returnmGameLogic.isBlank()

（2.3）类CGameDlg的设计

类CGameDlg主要用来和用户进行交互，对用户的输入鼠标点击事件进行响应，因此我们在CGameDlg上，增加鼠标的左键点击事件，可以通过打开类向导来添加，如下图所示。



为了将鼠标的点击位置和地图进行关联，定义了地图左上角坐标，地图所在矩形，以及图片大小等成员属性，同时也增加了画框DrawTipFrame()和画连接路径DrawTipLine()等函数，全新的CGameDlg定义如下：

计算机生成了可选文字:
—classCGameDlg：publlcCDialogEx
DECLAREDYNAMIC(CGameDlg)
public
//析准造函数
CGameDlg(CWnd*pParent
nullptr
virtual、CGameDlg0，
E#ifdefAFXDESIGNTIME
ni-皿[IDD=IDD_GAIE_DIALOG}，
莘一一三
#endif
private：
CPOintmGameRegionTop．
CSizemsizeElem-
CRectmGameRegion，
protected：
过二飞三二兰=二
二0三一0
二飞三二三三一过三
virtual、℃idDODataExchange(CDataExchange*PDX)
//DDX/DDV0；三
HICONmhlcon;
内存DC
CDCmdcMem;
CDCmdcBG
CDCmdcEle
/掩码
CDCmdcMask，
背到片路径
staticCStringBGPath《
元片路径
staticCStringElementPath
揎码片路径
staticCStringMaskPath《
创建一个CGameContr01对象
CGameControlmGameC
b001playing,
b001flrstSelect
DECLAREMESSAGEYAP
/渫戏是否进行中
//true小
preparedfor
public
virtual
afx
afx
afx
afx
afx
BOOL
VOId
VOId
VOId
VOId
VOId
OnlnitDialog0，
OnPaint0，
OnC1ickedBtnStart0
OnC1ickedBtnStop0
OnC1ickedBtnReset0
OnC1ickedBtnPrompt0
三=一三=

计算机生成了可选文字:
public
VOld
private.
VOld
VOld
VOld
VOld
VOld
VOld
VOld
p巛b112
SetGameMode(Intmode)
InitMode(int
mode)
InitBackground()O(d)-
SetButton(boolstart，boolstop，b001prompt,
b001reset)
InitElement(CStringElementPath，CStringMaskPath)
UpdateMap（、℃Id），
DrawTipFrame(intnROW，intnCOl)
DrawTipLine(stack<Vertex>verList)
增加的方法
afx_msgVOIdOnLButtonUp(UINTnFlags，CPOintpoint)
/巛标左键松开事件

(2.3.1) 对话框构造函数修改

在对话框构造函数中，对一些成员属性进行了初始化设置

计算机生成了可选文字:
ECGameDlgpParent
CDia10gEx(IDDGAMEDIALOG，
mGameRegionTop.y=VADTOP;
mGameRegionTop.x=MAPLEFT，
SiCX
sizeEIem.CY
/*=nulIptr*/)
pParent)
//50
//20
//40
//40
m
m
p
=PICWIDTH;
=PICHEIGHT，
mGameRegion.top=IAPTOP，
mGameRegionleft=MAPLEFT，
mGameRegionright=NAPLEFT+PICWIDTH*Cols;
mGameRegion.bottom_豐TOP+PICHEIGHT*Rows
aylng=false-
f．rstSelect
false-
GameMode

(2.3.2) 画框函数的设计

画框函数的重点是根据当前获得的图片坐标，准确计算图片的四个边框位置（left,top,right,bottom）

然后用调用FrameRect函数，进行边框绘制

计算机生成了可选文字:
、℃IdCGameDlg：DrawTipFrame(intnRow
CCIientDCdc(thls)，
CBrushbrush(RGB（233，43，43）），
CRectrtTipFrame，
intnCOl)
rtTipFrame.left=mGameReg10nTop.x+nCOl*msizeElemCX《
冰mSIZeEIem.CY，
rtTipFrametop=mGameRegionTop.y+nRow
rtTipFrame.right=rtTipFrame.left+msizeElem.CX；
rtTipFrame.bottom_rtTipFrame.top+msizeElem.2芏；
dc.FrameRect(rtTipFrame,&brush)

(2.3.3) 画线函数的设计

画线函数的重点是根据传入的图片连接路径列表，来画线，具体的流程是：

首先获取第一个点，移动到这个点位置，然后依次读取下一个点，从刚刚那个点到这个点开始画直线，依次进行，直到连接路径列表中所有的点都画完为止。

计算机生成了可选文字:
—voidCGameDlg:DrawTipLine(stack<Vertex>verList)
三
CClientDCdc(this)
CPenpenLine(PS_SOLID,2，RGB(),250））。
CPen*poldP引1dc.Selectobject(&penLine)，
VertexVTop，
CP01ntCP》
if(IverList.empty0）
、'Top=verList.top0
CP．X_mGameRegionTop.X+VTop.C01*msizeElem.Ci+msizeElem.CX
CP．y=mGameRegionTop.+vTop.row*msizeElem.CY+msizeElem.CY
dc.MoveTocp
verL1st.pop
while(IverList.empty0）
vTop=verList.top0
2；
2；
CP·X_mGameRegionTop.+¥TO鼽C01*msizeElem.Ci+msizeElem.CX
CP·y=mGameRegionTop.+¥TO鼽row*msizeElem.CY+msizeElem.CY
dc.LineToc
verList.pop0《
dc.SelectObject(pOldPen)》
2；
2；

(2.3.4) 鼠标左键松开事件函数的设置

计算机生成了可选文字:
—voidCGameDlg:OnLButtonUp(UINTnFlags,
if(:playingreturn
CPOintpoint)
if(point.y(mGameRegion.t00》》point.》0卫GameResion.Bottom
》point.X)mGameResion.right
point.X<mGameRegion.left
三
三
//如果没有在地的区域门，则不菅
returnCDialogEx.OnLButtonUp(nFlags,
//计亘当前点的坐标
intnROW(point.y_mGameRegionTop.y)
intnCol=(point.x_mGameRegionTop.幻
目nCol冫Cols
if(nRow冫Rows
returnCDialogEx:OnLButtonUp(nFlags,
point)
msizeEIem.CY，
／msizeElem.cx；
point)
if(mGameC.GetElement(nRow，ncol)
BLANK)
//如果这个区域没有片，则返亘
returnCDialogEx.OnLButtonUp(nFlags,point)

计算机生成了可选文字:
if(firstSelect)[
//如果罡此次0壬罡第一《
DrawTipFrame(nRow，ncol)“
。则画边榧。卉进行
mGameC.SetFirstPoint(nRow,ncol)
《//如果此次0壬罡第二个，
。怎画边榧。卉进行
DrawTipFrame(nRow，ncol)“
mGameC.SetSecondPoint01R“，ncol)
//同时判斫两个0罡否相连
stack<Vertex>verL1st
巧001bSuc=mGameC.Link(verList)
汾(bSuc)[
//如果相连，则画连接线路，
DrawTipLine(verLi(t)《
重新加道背景，卉更新新道
mdcMem.BitB1t，0，800，0，&卫dcBG，
UpdateMap0，
Sleep（2）
In、ralidateRect(FALSE)
汾(bsucmGameC.iswin0）
MessageBox(TEXT（“
/置按钮
SetButton(TRUE，FALSE,FALSE,FALSE)
0，
0，
SRCCOPY)《
playing=false．
!firstSelect;一次和第
firstSe1ect
CDialogEx：OnLButtonUp(nFlags,point)，
一帙卢壬切换标志

(2.4) 运行测试

运行程序，点击开始游戏按钮，界面上展示地图，点击合适的图片能够进行图片是否连通的判断，并且进行消除操作。最后所有图片消除后，能够提示游戏成功。

计算机生成了可选文字:
。23