学号 WA2224013 专业 机器人工程 姓名 郭义月

实验日期 **2024.4.12**  教师签字 成绩

实验报告

【实验名称】 图形界面编程

【实验目的】

用Matlab编写连连看游戏程序

【实验原理】

Matlab GUI编程的相关原理

【源码】

主函数

clc;close all;clear;

fig = uifigure('Name','连连看');

grid = uigridlayout(fig,'ColumnWidth',repmat({'1x'},1,10),...

'RowHeight',repmat({'1x'},1,10),...

'ColumnSpacing',0,'RowSpacing',0, ...

'Padding',[0 0 0 0]);

global M;

M = zeros(1,12);

M(2:11,:) = repmat([0,ones(1,10),0],10,1);

M(12,:) = zeros(1,12);

%定义全局变量M，当路径上无障碍时置0

%M的初始状态是，10\*10的全1矩阵，外边一圈0

for x = 1:10

for y = 1:10

h(x,y) = uibutton(grid,'state','Text',num2str(randi([0,9])),'FontSize',30);

%产生10\*10个切换按钮作为窗格

end

end

for x = 1:10

for y = 1:10

h(x,y).ValueChangedFcn = @(~,~)WhenPushed(h,x,y);

%为每个状态按钮添加回调函数

end

end

function WhenPushed(h,x,y) %回调函数

global M %定义全局变量，这样才可以在函数中更改M的值

if sum([h.Value]) == 2 %h.Value选中时置1，通过求和是否等于2判断是否已经选中两个

x1 = [];y1 = []; %初始值定义放在if里面，这样才能保证，每次都是最新的选中的坐标

for i = 1:10

for j = 1:10

if(h(i,j).Value == 1)

x1 = [x1,i];

y1 = [y1,j];

%把选中的坐标放在x1，y1中，x1，y1是1行2列的矩阵

end

end

end

if IsRemoved(h,x1,y1) %当函数判断选中的两个按钮可以消除时

M(x1(1)+1,y1(1)+1) = 0;

M(x1(2)+1,y1(2)+1) = 0; %M矩阵中响应的坐标置0

h(x1(1),y1(1)).Visible = 'off';

h(x1(2),y1(2)).Visible = 'off'; %消去的按钮设置为不可见

h(x1(1),y1(1)).Value = 0;

h(x1(2),y1(2)).Value = 0; %取消对选中按钮的选中，保证下次仍然可以用sum>2来判断是否选中两个

else

h(x1(1),y1(1)).Value = 0;

h(x1(2),y1(2)).Value = 0; %当不能消除时，取消对按钮的选中

end

end

end

函数1：判断选中的两个按钮是否可以消除

function [s] = IsRemoved(h,x,y)

global M;

if h(x(1),y(1)).Text ~= h(x(2),y(2)).Text

s = 0; %如果选中的两个按钮的Text不相等，则不能消除，返回0

elseif vertical(h,x,y)

s = 1; %按照ppt上面的提示进行垂直判断，能消除返回1

elseif horizontal(h,x,y)

s = 1; %按照ppt上面的提示进行水平判断，能消除返回1

else

s = 0; %如果不能消除，返回0

end

函数2：垂直判断

function [t] = vertical(~,x,y)

global M;

for i = 1:13

if i == 13 %如果i能到达13，表示没有提前跳出循环，即不能垂直判断不能消除

t = 0; %不能被消除，返回0

break

end

u = 1; %定义变量u，当u被置0表示不能消除

if i <= x(1)+1

if sum(M(i:x(1),y(1)+1)) ~= 0

u = 0;

continue

end

else

if sum(M(x(1)+2:i,y(1)+1)) ~= 0

u = 0;

continue

end

end

%以上是可以连通的第一条线，找到(i,y(1)+1)

if y(1) < y(2)

if sum(M(i,y(1)+2:y(2))) ~= 0

u = 0;

continue

end

else

if sum(M(i,y(2)+2:y(1))) ~= 0

u = 0;

continue

end

end

%以上是找到连通的第二条线

if i <= x(2)+1

if sum(M(i:x(2),y(2)+1)) ~= 0

u = 0;

continue

end

else

if sum(M(x(2)+2:i,y(2)+1)) ~= 0

u = 0;

continue

end

end

%以上是找到连通的第三条线

if u == 1

t = 1;

break; %如果这三条线都能找到，说明选中的两个按钮能连通，t返回1

end

end

end

函数3：水平判断 （和垂直判断类似，相关注释已经在垂直判断中给出）

function [t] = horizontal(~,x,y)

global M;

for i = 1:13

if i == 13

t = 0;

break

end

u = 1;

if i <= y(1)+1

if sum(M(x(1)+1,i:y(1))) ~= 0

u = 0;

continue

end

else

if sum(M(x(1)+1,y(1)+2:i)) ~= 0

u = 0;

continue

end

end

if x(1) < x(2)

if sum(M(x(1)+2:x(2),i)) ~= 0

u = 0;

continue

end

else

if sum(M(x(2)+2:x(1),i)) ~= 0

u = 0;

continue

end

end

if i <= y(2)+1

if sum(M(x(2)+1,i:y(2))) ~= 0

u = 0;

continue

end

else

if sum(M(x(2)+1,y(2)+2:i)) ~= 0

u = 0;

continue

end

end

if u == 1

t = 1;

break;

end

end

end

【运行结果截图】

|  |  |
| --- | --- |
|  |  |

【实验总结】

目前本程序没有发现Bug；

生成state类型的按钮非常有助于编写程序，第一次做的时候没有注意到PPT上面的提示，生成普通的按钮的话如何确定已经选中了两个按钮非常复杂，这是由于每个按钮回调函数里面只能改变按钮自己的参数，想要实现计数（确定是否已经选中了两个）就需要设置很多全局变量，而state类型的按钮只需要判断其Value是否为1就可以判断是否选中该按钮，对整个按钮矩阵的Value矩阵求和，就可以知道是否选中两个按钮；

在判断两个选中的按钮是否可以消除时，由于本人水平实在有限，所以用了简单的循环加if判断，对于水平和垂直上面的12个左边，依次判断是否可以用三条线连接，代码中变量flag和u都是布尔变量的作用，方便找到路线之后及时跳出循环；

在写水平垂直判断时，如果一次写不完美可以先把一些简单的情况单独if出来，先让程序中的一部分按钮可以消去，这样有利于之后代码的调试。当每种情况都考虑到位之后，可以把一些重复的部分合并简化；

生成M初始值矩阵的时候使用如下三行代码，充分利用零矩阵，全1矩阵还有矩阵之间的运算可以简化生成初始M的代码；

M = zeros(1,12);

M(2:11,:) = repmat([0,ones(1,10),0],10,1);

M(12,:) = zeros(1,12);

由于使用了过多的循环与if判断，代码整体不够简洁，没有充分发挥matlab矩阵编程的优势，但是逻辑简单易懂，暴力简单。